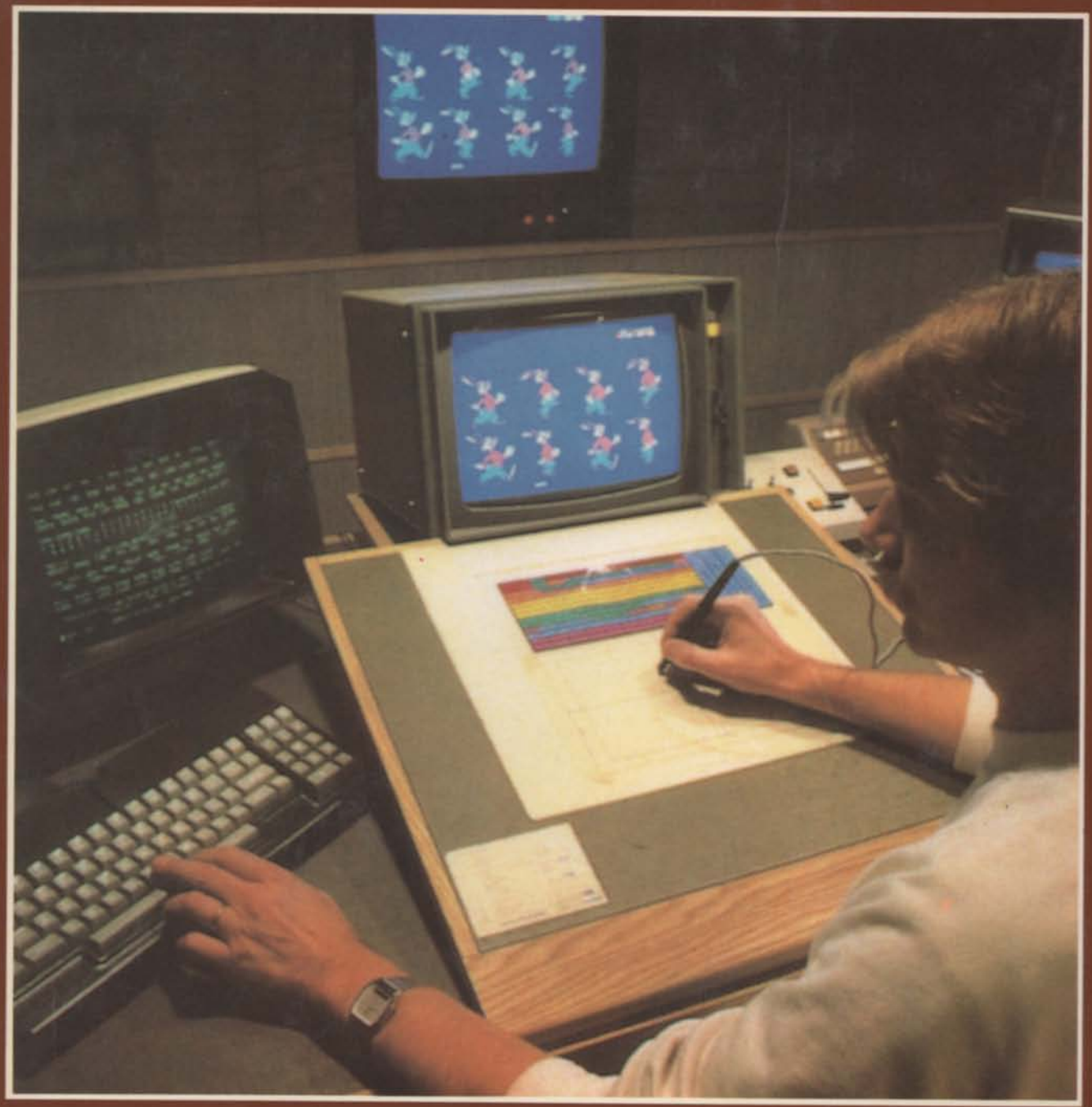
IL MIO COMPUTER

CORSO PRATICO DI INFORMATICA: MICRO, MINI E PERSONAL COMPUTER



ISTITUTO GEOGRAFICO DE AGOSTINI

IL MIO COMPUTER

CORSO PRATICO DI INFORMATICA: MICRO, MINI E PERSONAL COMPUTER

volume secondo

ISTITUTO GEOGRAFICO DE AGOSTINI

Direzione: Achille Boroli - Adolfo Boroli
Direzione editoriale: Mario Nilo; *settore fascicoli:* Jason Vella

Redazione dell'edizione italiana a cura di:
Edizioni Remo Sandron - Firenze
Redazione dell'edizione inglese:
Richard Pawson (*Editor*); David Tebbutt (*Consultant Editor*); Chris Cooper (*Executive Editor*); David Whelan (*Art Director*); Gareth Jefferson, Roger Ford (*Collaborators*)
Designed and Produced by Bunch Partworks Limited
Traduzione a cura di: Daniel Quinn

Coordinamento grafico: Otello Geddo
Coordinamento fotografico a cura del Centro Iconografico dell'Istituto Geografico De Agostini

Titolo originale dell'opera: The home computer course
© Orbis Publishing Limited, Londra, 1983.
© Istituto Geografico De Agostini S.p.A., Novara, 1984.
Stampato in Italia - IGDA Officine Grafiche, Novara, 1984.

Sommario

241 Maghi del Flipper
243 I missili Cruise
244 Le liste concatenate
246 Effetti speciali
248 Che tempo farà, domani?
250 Il Sord M5
253 Il controllo di parità
254 Scegliere la struttura adatta
258 Le tavolette grafiche
260 Gottfried Leibniz
261 Il Lisa e le "icone"
264 Una finestra sul mondo
266 I giochi... postali
267 Il comportamento dei modelli
269 La generazione di caratteri
270 Il Tandy Color Computer
272 L'inserimento di nuovi nominativi
276 Consigli sul suono... e sulla grafica
278 I display a cristalli liquidi
280 Soluzioni degli esercizi di riepilogo
281 Un servo obbediente
284 I suoni del Vic... le luci del Dragon
286 I vari tipi di "sort"
288 Quattro bit... per un labirinto!
290 Il Memotech MTX 512
292 Scegliamo il menu adatto!
296 Uomini e topi
298 Il codice Hamming
300 Norbert Wiener
301 La comunicazione di massa
304 Memorie di oggi, memorie di domani
306 I "Bulletin Board"
308 Un redattore ineccepibile
309 Lo Sharp MZ-711
312 Semplici suoni... Immagini primarie
314 Il braccio e la mente
316 Come usare i file
320 Lyons: pasticcini ed informatica
321 Il computer ergonomico
324 Tracce e settori
326 Oltre i limiti
328 Una marcia in più
330 Il Tandy MC-10
332 I joystick della nuova generazione
334 Sistemi audio... Programmi luminosi
336 Scambi e ricerche
340 Konrad Zuse
341 Viaggiare grazie ai computer
344 I linguaggi del computer
346 Il computer astronomo
348 Un passo alla volta
349 L'Apple IIe
352 I dischi Winchester
354 Dai moduli al programma intero
358 Qualità musicali... Lampi di luce
360 Leonardo Torres
361 Inventiamo un gioco
364 Le mappe di memoria
366 Un po' d'immaginazione
368 Che accadrebbe se...
370 L'Electron della Acorn
372 Le stampanti "a getto"
374 A prova di suono. Immagini d'autore
376 Un "giro" di prova
380 Powers e Hollerith
381 Fantascienza e realtà
384 I generatori di programmi
386 Un arcobaleno di accessori
388 I chip ULA
389 Tre giochi di simulazione
390 L'Olivetti M-10
392 Suoni spettacolari... Immagini alla ribalta
394 I "sensi" del computer
396 Velocità di ricerca
400 Vannevar Bush
401 Per i più piccoli
404 Gli "spelling checker"
406 Programmi generatori specializzati
408 Suoni incredibili... Luci sfolgoranti
410 L'Osborne-1
413 Un nuovo Sort
414 Il computer stenografo
416 Chi cerca...
420 La Bell Laboratories
421 Il Computer Aided Design
424 La macchina di Turing
426 Idee sonore... Onde di luce
428 Il Computerese
430 Il Commodore PET 4032
432 Elementi... sovversivi
434 I dischi a laser
436 Ritocchi finali
440 Grace Hopper
441 Guerra e pace
444 Gli attrezzi del mestiere
446 Il riconoscimento vocale
448 Il linguaggio macchina
450 Il QL Sinclair
452 Principi del suono... Luci rinfrescanti
454 Dpef Dsbdljoh
456 Una questione di stile
460 Il MARK 1
461 Rischio calcolato
464 Qualche passo in Assembly
466 Il futuro
468 Salti generazionali
470 Il 380Z della Research Machines
472 Suoni ideali... Luci di posizione
474 Quanto vale il BASIC?
478 Tutti gli uomini del... computer

Indice per settore

Panorami



241 Maghi del Flipper
243 I missili Cruise
248 Che tempo farà, domani?
258 Le tavolette grafiche
264 Una finestra sul mondo
266 I giochi... postali
278 I display a cristalli liquidi
281 Un servo obbediente
296 Uomini e topi
301 La comunicazione di massa
314 Il braccio e la mente
321 Il computer ergonomico
332 I joystick della nuova generazione
341 Viaggiare grazie ai computer
346 Il computer astronomo
352 I dischi Winchester
361 Inventiamo un gioco
368 Che accadrebbe se...
372 Le stampanti "a getto"
381 Fantascienza e realtà
394 I "sensi" del computer
401 Per i più piccoli
414 Il computer stenografo
421 Il Computer Aided Design
434 I dischi a laser
441 Guerra e pace
446 Il riconoscimento vocale
466 Il futuro

Software



244 Le liste concatenate
261 Il Lisa e le "icone"
267 Il comportamento dei modelli
286 I vari tipi di "sort"
288 Quattro bit... per un labirinto!
306 I "Bulletin Board"
328 Una marcia in più
344 I linguaggi del computer
366 Un po' d'immaginazione
389 Tre giochi di simulazione
404 Gli "spelling checker"
406 Programmi generatori specializzati
432 Elementi... sovversivi
444 Gli attrezzi del mestiere
454 Dpef Dsbdljoh
461 Rischio calcolato

Suoni e luci



246 Effetti speciali
276 Consigli sul suono... e sulla grafica
284 I suoni del Vic... le luci del Dragon
312 Semplici suoni... Immagini primarie
334 Sistemi audio... Programmi luminosi
358 Qualità musicali... Lampi di luce
374 A prova di suono... Immagini d'autore

392 Suoni spettacolari...
Immagini alla ribalta
408 Suoni incredibili...
Luci sfolgoranti
426 Idee sonore...
Onde di luce
452 Principi del suono...
Luci rinfrescanti
472 Suoni ideali...
Luci di posizione

Hardware

250 Il Sord M5
270 Il Tandy Color Computer
290 Il Memotech MTX 512
304 Memorie di oggi,
memorie di domani
309 Lo Sharp MZ-711
326 Oltre i limiti
330 Il Tandy MC-10
349 L'Apple IIe
370 L'Electron della Acorn
386 Un arcobaleno di accessori
390 L'Olivetti M-10
410 L'Osborne-1
430 Il Commodore PET 4032
450 Il QL Sinclair
470 Il 380Z della Research
Machines

Le chiavi dell'Informatica

253 Il controllo di parità
269 La generazione di
caratteri
298 Il codice Hamming
308 Un redattore ineccepibile
324 Tracce e settori
348 Un passo alla volta
364 Le mappe di memoria
384 I generatori di programmi
388 I chip ULA
413 Un nuovo sort
424 La macchina di Turing
428 Il Computerese
448 Il linguaggio macchina
464 Qualche passo in
Assembly
468 Salti generazionali

Programmazione BASIC

254 Scegliere la struttura
adatta
272 L'inserimento
di nuovi nominativi
280 Soluzioni degli esercizi
di riepilogo
292 Scegliamo il menu adatto!
316 Come usare i file
336 Scambi e ricerche
354 Dai moduli al programma
intero
376 Un "giro" di prova
396 Velocità di ricerca
416 Chi cerca...
436 Ritocchi finali
456 Una questione di stile
474 Quanto vale il BASIC?

I pionieri dell'Informatica

260 Gottfried Leibniz
300 Norbert Wiener
320 Lyons: pasticcini
ed informatica
340 Konrad Zuse
360 Leonardo Torres
380 Powers e Hollerith
400 Vannevar Bush
420 La Bell Laboratories
440 Grace Hopper
460 Il MARK 1
478 Tutti gli uomini
del... computer

Maghi del Flipper

Ultime novità nei computer game: il "fai da te"!

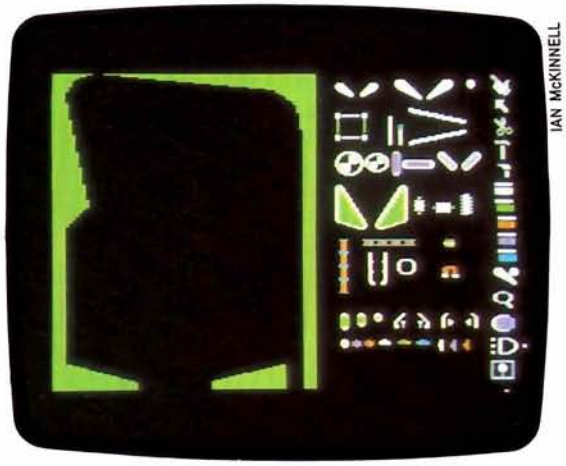
In una industria in continua evoluzione, come quella dell'informatica, è ragionevole aspettarsi un gran numero d'innovazioni, ma è ancora abbastanza raro imbattersi in un prodotto radicalmente diverso, concettualmente e qualitativamente, dagli altri già in vendita. Qualche differenza e originalità offre il programma "PCS", cioè "Pinball Construction Set" (un "flipper" in scatola di montaggio), prodotto dalla Budgeco. Progettato appositamente per l'Apple II, il programma richiede 48 Kbyte di memoria, l'impiego di un'unità a disco e di un joystick. La sua funzione è, apparentemente, molto semplice: permettere la creazione e l'uso di un flipper sullo schermo del computer. Il giocatore può scegliere tra 38 diversi "oggetti" da disporre come meglio crede sul tavolo del flipper. La scelta degli attrezzi avviene mediante un "menu".

Una volta preparato il tavolo da gioco nel modo desiderato (è consentito collocare fino a 128 pezzi sul tavolo, senza limitazione su quante volte si ripete un particolare pezzo), non rimane altro che iniziare la partita. Il gioco è comandato mediante il joystick. Possono parteciparvi fino a quattro giocatori, ma a ciascuno spetta una sola pallina invece di 3, come di solito avviene con i flipper veri e non esiste la pallina in omaggio. Alla fine della partita basta premere il tasto ESCAPE per ritornare al menu. Il giocatore viene incoraggiato a rendere più difficile il gioco secondo il punteggio ottenuto dopo ogni partita.

Sia nella concezione che nel funzionamento del gioco, il sistema PCS è l'esempio di una nuova tendenza nel settore software: quella di offrire all'utente un "dialogo" amichevole durante l'uso dei programmi. Una volta caricato il programma (e per far questo basta inserire il disco e premere il tasto RETURN), quasi tutta l'azione è controllata dal joystick. Il primo "attrezzo" che viene adoperato sullo schermo è una piccola mano, che serve per la selezione degli oggetti da impiegare sul tavolo del flipper (per esempio, il tipo di respingente o di ostacolo). Se viene azionato il tasto del joystick, la mano raccoglie l'oggetto indicato e, manovrando il joystick (sempre con il tasto premuto), lo trascina fino alla posizione desiderata. Per depositare l'oggetto, basta rilasciare il tasto del joystick. Uno degli aspetti più interessanti è il fatto che, oltre alla personalizzazione del tavolo da gioco, è possibile anche modificare il comportamento della pallina nel gioco. Un flipper (l'aletta usata per respingere la pallina), per esempio, ruota sempre di 45 gradi, muovendosi prima verso l'alto e poi verso il basso e ogni respingente, urtato dalla pallina, la respinge imprimendole una spinta ben precisa. La pallina ubbidisce alle leggi di Newton sul moto dei corpi e scende, lungo il tavolo, secondo la legge di gravità.

L'utente, però, può anche modificare alcuni parametri (quali, ad esempio, la forza di gravità o anche il fattore tempo) per rendere imprevedibile il comportamento della pallina e vivacizzare il gioco. Anche questa funzione è controllata mediante joystick.

Nel package sono presenti anche tutte quelle funzioni che ci si aspetta di trovare in un buon programma basato su di una grafica ad alta qualità: si possono tirare righe o deformarle a piacimento; congiungere, graficamente, due o più oggetti del tavolo da gioco;



IAN McKINNELL

Roba da bambini

Nel PCS è incorporata una gestione degli effetti sonori visivi molto realistica. Talvolta è più divertente la fase di creazione del flipper, che non l'effettivo gioco. L'unica funzione non ancora inclusa è il TILT!

Un gioco personalizzato

Il "Pinball Construction Set" è un programma per la costruzione, sullo schermo di un computer, di un "flipper" personalizzato: il tavolo da gioco, inizialmente vuoto, può essere riempito scegliendo, da una vasta gamma di oggetti (cerchi respingenti, bersagli, luci intermittenti, ostacoli, ecc.), quelli desiderati.

Nella colonna di destra ci sono gli "attrezzi" di base, oltre al menu di funzioni, con le quali è possibile modificare le dimensioni, la forma, il colore e il grado di interazione dei pezzi. Uno speciale comando serve per archiviare, su dischetto, la definizione del "nostro" flipper



THE IMAGE M



IAN McKINNELL

colorare gli oggetti usando un'apposita tavolozza; ingrandire piccoli particolari dell'immagine grafica per poterli definire con maggiore dettaglio.

L'aspetto più importante non consiste tanto nelle singole funzioni e nelle capacità del PCS, quanto nella filosofia generale del funzionamento: la "programmazione orientata" (così si chiama questo particolare modo di concepire il software) è stata definita come la scoperta più sensazionale nel campo dell'informatica, dopo l'introduzione dei linguaggi ad alto livello, avvenuta negli anni '50.

La maggior parte degli home computer dispone di memoria e capacità di elaborazione sufficienti per le normali esigenze dell'utente, ma, in futuro, vi saranno enormi sviluppi verso programmi il cui impiego sarà più semplice ed "amichevole" ("user friendly"). Rimane un fatto notevole che, nel programma PCS, questo risultato si sia ottenuto usufruendo di soli 48 Kbyte.

La "programmazione orientata" si presta molto bene in applicazioni ricreative o in programmi basati sulla grafica, mentre è necessaria una buona dose d'inventiva per poterla utilizzare nel campo del software commerciale. In parte, le "lavagne elettroniche" (quali il Visicalc o il Supercalc), benché non si avvalgano della grafica come mezzo principale di comunicazione, si possono considerare "programmi orientati", in quanto ciascun campo (o celletta) può contenere sia un particolare dato sia le informazioni che lo definiscono.

Un altro esempio è costituito dal sistema Lisa della Apple: in questo caso, per selezionare il programma desiderato (rappresentato con un particolare simbolo grafico), viene utilizzato un "topo" (mouse), col quale si manovra un indice sullo schermo del computer. Per esempio, il programma di "word processing" è rappresentato su un foglio di carta da lettere, mentre il programma per la preparazione di grafici è rappresentato su un foglio di carta millimetrata.

Tra le molte funzioni del Lisa quella forse più interessante è costituita dal metodo adoperato per il passaggio di dati da un programma ad un altro. Una delle icone (così vengono definiti i simboli delle funzioni sullo schermo) rappresenta una piccola bacheca. Se si desidera estrarre una porzione di dati dalla lavagna elettronica, per riprodurla sotto forma di un grafico, basta definire la "finestra" della lavagna elettronica, trasferirla nella "bacheca" (nient'altro, in realtà, che una zona temporanea di memoria), da dove viene opportunamente estratta dal programma che stampa i grafici.

Quando abbiamo parlato di videogiochi (vedere a pag. 221), abbiamo visto che ne esistono più tipi, anche molto diversi fra di loro: il PCS costituisce una categoria tutta nuova, permettendo allo stesso utente di "costruirsi" un ambiente di gioco personalizzato. Può darsi che, in un futuro non molto lontano, anche altri tipi di giochi per home computer funzionino secondo questa rivoluzionaria concezione e potremo "personalizzare" i comuni videogiochi, come lo *Space Invaders*, a nostro piacimento. È molto probabile che ciò avvenga in concomitanza con l'introduzione, sul mercato, dei computer della quinta generazione, i cosiddetti computer "pensanti".

Passo per passo

Le quattro foto illustrano le varie fasi della costruzione del tavolo del "flipper". Per prima cosa si dispongono alcuni pezzi "principali", viene poi aggiunto un poligono, in posizione centrale, che viene rimodellato e colorato di arancione. Infine, alcuni degli oggetti vengono collegati assieme "logicamente" (ossia mediante una funzione che simula le porte logiche di tipo AND), in modo da premiare il giocatore che riesca ad attivare una particolare sequenza di oggetti

Pronti, attenti, via!

Una volta costruito il tavolo da gioco, è possibile conservarne le caratteristiche su disco. Siccome tutte le funzioni occorrenti al gioco vengono conservate assieme alle caratteristiche del tavolo da gioco, non è necessario ricorrere ai programmi originali ogni volta che si desideri giocare



CHRIS STEVENS

La filosofia del sistema

Il PCS non è solamente un gioco educativo e divertente, ma costituisce anche un ottimo esempio di "programmazione orientata". Nella programmazione normale, prima viene definita la struttura dei dati e quindi vengono sviluppati i moduli del programma necessari all'elaborazione. Nella "programmazione orientata", invece, i calcoli e le procedure sono inseparabili dai dati. Quando, nel programma PCS, il simbolo di uno degli oggetti viene piazzato sul tavolo da gioco, non vengono fissati soltanto i dati ad esso relativi (in questo caso la forma e le dimensioni dell'oggetto), ma anche, allo stesso tempo, i gruppi di istruzioni necessarie all'attivazione dell'oggetto in questione.

La programmazione orientata è molto indicata in applicazioni quali, ad esempio, le lavagne elettroniche (spreadsheet), dove il campo relativo ad un dato contiene anche la formula per arrivare a tale risultato.

La tendenza attuale, nel software commerciale, è quella di offrire, sullo schermo del computer, una riproduzione di vari oggetti da scrivania: quando l'indice viene posizionato sull'immagine di un foglio di carta da lettere, per esempio, viene attivato il programma di "word processing", mentre quando l'indice viene spostato sul simbolo di un raccoglitore da archivio il computer "capisce" che i dati appena elaborati vanno "archiviati". Questo sistema facilita enormemente lo svolgimento delle operazioni più ricorrenti

I missili Cruise

La tecnologia impiegata in questi sofisticatissimi ordigni

Grazie a sistemi di pilotaggio computerizzato, all'astronauta Armstrong è stato possibile mettere per primo il piede sulla luna.
Naturalmente, per le navicelle interplanetarie è necessario un altissimo grado di precisione ingegneristica, ma senza l'impiego di hardware e software specializzati, non sarebbe mai possibile eseguire i precisi calcoli sul posizionamento in tempo utile per permettere il contatto fra due oggetti così distanti l'uno dall'altro, anche se uno di questi è grande quanto la luna.
Secondo gli attuali requisiti militari, una testata nucleare, dopo aver sorvolato un continente intero, deve poter centrare un bersaglio entro un raggio di 20-30 metri. Ovviamente, tali obiettivi si possono raggiungere soltanto impiegando una tecnologia, per l'elaborazione di dati, molto sofisticata.
Le prime esperienze in campo militare rilevarono che il problema fondamentale era costituito dall'impossibilità di correggere la rotta, una volta effettuato il lancio. Il primo salto di qualità avvenne con il perfezionamento di sistemi di guida automatica, che permettevano di stabilire la distanza e la direzione del missile in relazione ad un punto preciso sulla superficie terrestre (ad esempio, la base di lancio). Purtroppo, anche i più moderni sistemi di questo genere non escludono la possibilità di errori.
Un metodo più attendibile è basato sull'impiego, come punto di riferimento, di satelliti in orbita geo-stazionaria. Il difetto più grande di questi sistemi è che la rotta del missile, e probabilmente anche il bersaglio, grazie al perfezionamento dei sistemi di intercettazione radar, è individuabile anche dal nemico. Per ovviare a questo inconveniente, il missile ideale dovrebbe consistere in un congegno che navighi a bassa quota, dotato di un piccolo radar "trasversale", capace di valutare quale sia la rotta da prendere per colpire il bersaglio. Il "Cruise" funziona proprio così.
Questo missile mantiene aggiornata la sua posizione analizzando i rilevamenti fisici del terreno sopra al quale sta volando. Questo avviene mediante la collimazione di una serie di letture altitudinali, rilevate da un radar altimetrico estremamente preciso, con una cartina in rilievo del terreno, conservato a bordo in una memoria "a bolle".
Questo sistema, progettato dalla McDonnell Douglas, viene chiamato TERCOM (da: TERrain COntour Matching, ossia "confronto sui contorni del terreno"), oppure indicato con la sigla DPW-23. Ciascun missile contiene nella propria memoria a bolle circa 25 diversi "itinerari", con tutte le caratteristiche del terreno sopra al quale il missile si troverebbe a volare. Tuttavia, esistono alcuni svantaggi: il sistema non funziona né sulla superficie dell'acqua, né su zone desertiche, in quanto le caratteristiche del terreno vengono costantemente influenzate dal vento. Inoltre, è ragionevole supporre che lo stesso valga anche per la parte settentrionale dell'Europa, dove, durante i mesi invernali, gran parte del terreno è coperto dalla neve.
Per i missili Cruise non si adopera il sistema di rilevamento durante tutto il volo: esso non viene attivato durante il volo in quota sopra un terreno "amico". Ma, non appena il missile diviene vulnerabile ad un attacco nemico (proveniente da terra o dall'aria), esso si "tuffa" fino ad un'altezza di circa 15 metri da terra e mantiene questa quota lungo tutto il volo sul territorio nemico. A questo punto il missile potrebbe essere fuori rotta, ma i tecnici ritengono che esso sarà sempre sufficientemente vicino ad uno degli itinerari conservati nella memoria per potersi automaticamente riposizionare con precisione.
Quando il missile si avvicina al bersaglio viene attivata l'Unità di Correlazione Terminale che contiene, sempre in una memoria a bolle, un'immagine digitale dettagliata della zona da colpire vista come apparirebbe ad un missile in volo. Da test eseguiti è stato dimostrato che il sistema dovrebbe colpire il bersaglio entro un raggio di soli 18 metri, e questo dopo un volo di circa 2.800 km.



per concessione della AEROSPACE PUBLISHING LTD.



per concessione della NEW SCIENTIST

Il missile autocomandato

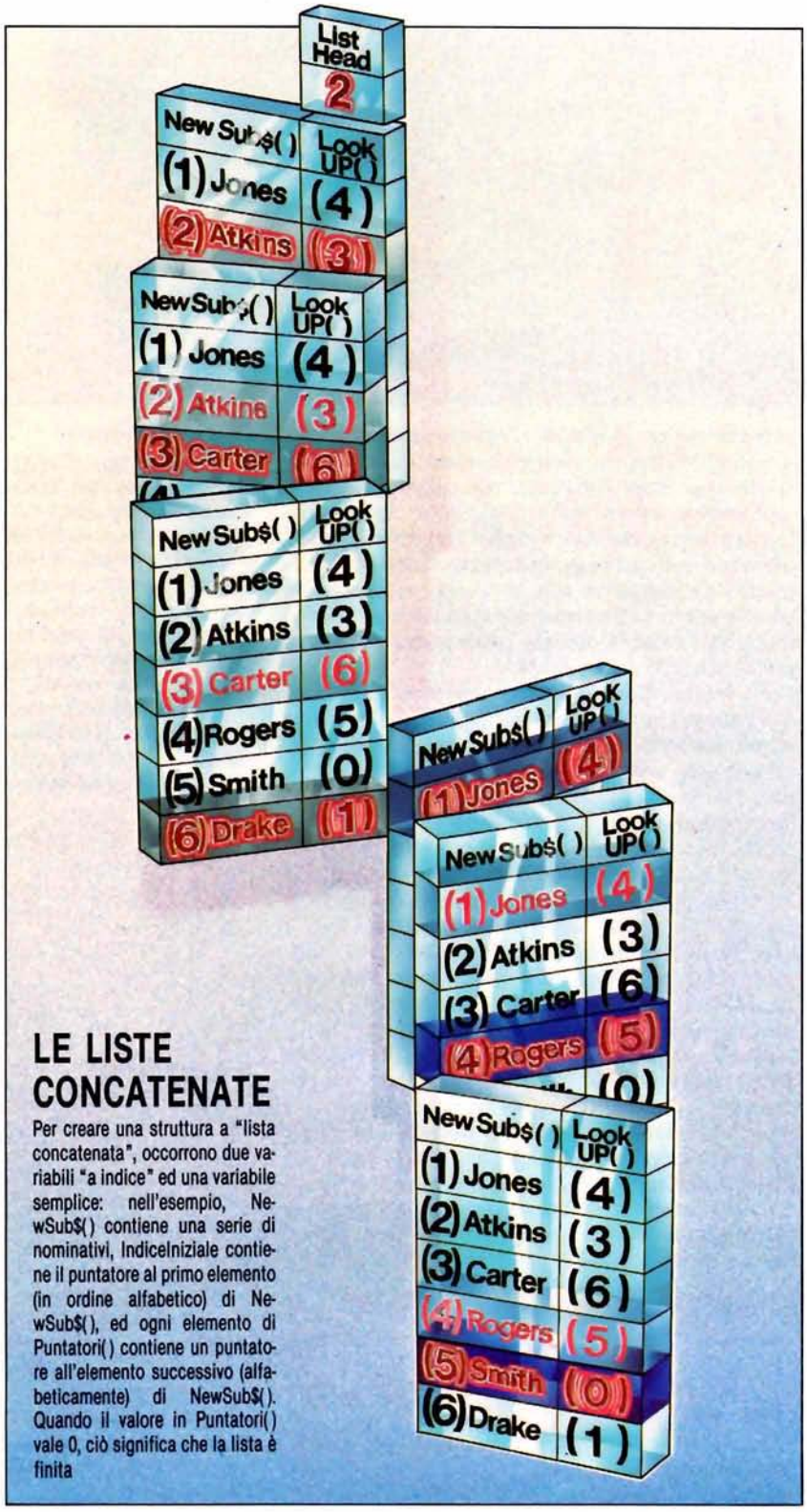
Il General Dynamics "Tomahawk" Ground Launched Cruise Missile (conosciuto come il missile "Cruise") ha una lunghezza di 6,40 metri e pesa meno di 1.200 chili. Viene lanciato da unità portamissili mobili. Inizialmente, esso assomiglia ad un normale missile, ma ben presto spiega le piccole ali di cui è dotato e naviga ad una quota fissa molto bassa, alimentato da un motore turbogetto a doppio flusso estremamente piccolo e compatto

La memoria "a bolle"

Nella memoria a bolle le condizioni logiche 1 e 0 vengono ottenute dalla presenza o meno di microscopici "rigonfiamenti" magnetici (detti, appunto, bolle) creati su piccolissimi chip di granato. Il vantaggio di questo tipo di memoria risiede nell'alto grado di concentrazione (attualmente 1 milione di bit, ossia 28 Kbyte per chip) e nella possibilità di conservazione permanente dei dati, anche quando l'alimentazione viene a mancare. Tuttavia la memoria a bolle opera ad una velocità ridotta rispetto a RAM convenzionali

Le liste concatenate

Vediamo come simulare una struttura inesistente nel linguaggio BASIC



LE LISTE CONCATENATE

Per creare una struttura a "lista concatenata", occorrono due variabili "a indice" ed una variabile semplice: nell'esempio, NewSub$( ) contiene una serie di nominativi, IndiceIniziale contiene il puntatore al primo elemento (in ordine alfabetico) di NewSub$( ), ed ogni elemento di Puntatori( ) contiene un puntatore all'elemento successivo (alfabeticamente) di NewSub$( ). Quando il valore in Puntatori( ) vale 0, ciò significa che la lista è finita

Nella memoria di un computer possono esserci migliaia di dati, conservati sotto forma di correnti elettriche. Per comodità, i segnali vengono spesso considerati a gruppi di otto, costituendo un byte, ma la struttura dei dati dipende dall'unità centrale (il processore) del computer ed in base ad essa il processore è in grado di stabilire con esattezza se un particolare dato memorizzato rappresenta un carattere alfabetico oppure è una parte di un'istruzione o di un numero.

La maggior parte dei linguaggi di programmazione offre una limitata gamma di strutture (alcune di esse, poi, intimamente connesse alla circuiteria, rimangono totalmente estranee al normale impiego da parte dell'utente). Il BASIC, ad esempio, dispone di due soli tipi di strutture: le variabili semplici e quelle con indice (dette anche matrici). Altri linguaggi offrono una maggiore scelta.

In BASIC, sfruttando le variabili ad indice, è possibile simulare anche altri tipi di strutture.

Vediamo, attraverso un semplice esempio, come utilizzare qualche nuova struttura. Supponiamo che un'azienda telefonica conservi una lista di nuovi abbonati, da inserire nel nuovo elenco telefonico. I nominativi, per una più rapida consultazione, dovrebbero risultare in ordine alfabetico, ma le continue aggiunte potrebbero alterare quest'ordine. Mettiamo che al lunedì mattina la lista dei nuovi abbonati, che indichiamo con NuoviAbb$( ), contenga i seguenti nominativi:

NuoviAbb$( )	Indici( )
(1) Giusti	(2)
(2) Aloisi	(3)
(3) Conti	(6)
(4) Rossi	(1)
(5) Serra	(4)
(6) Danti	(5)

La matrice Indici( ) contiene gli indici da usare per la variabile NuoviAbb$( ), volendo accedere in ordine alfabetico ai nominativi in essa contenuti. Pertanto, il primo valore di Indice( ) ci dice che il primo nominativo (in ordine alfabetico) è contenuto in NuoviAbb$(2), mentre il secondo è in NuoviAbb$(3) e così via. Per semplicità, abbiamo riportato soltanto dei cognomi, ma in realtà ad ogni nominativo vengono associati anche il nome, l'indirizzo, ed altre informazioni. Tipicamente occorrono circa 60

caratteri per ciascun nominativo. Spostare blocchi di 60 caratteri all'interno della memoria (operazione necessaria per un riordino alfabetico, ad esempio), richiede molto tempo ed un notevole spreco di memoria. Risulta più pratico lasciare NuoviAbb$( ) inalterata ed utilizzare i valori della variabile Indici( ). L'aggiunta di un nuovo nominativo (Bastogi) porta alla seguente situazione:

NuoviAbb$( )	Indici( )
(1) Giusti	(2)
(2) Aloisi	(7)
(3) Conti	(3)
(4) Rossi	(6)
(5) Serra	(1)
(6) Danti	(4)
(7) Bastogi	(5)

Si noti che l'unica differenza, nei valori di Indici( ), consiste nell'aver inserito opportunamente l'indice relativo al nominativo Bastogi, e nell'aver scalato i successivi valori di una posizione. Per l'inserimento di un nuovo nominativo, la procedura è: "accodare" in NuoviAbb$() il nuovo nominativo, inserire l'indice appena adoperato nella corretta posizione (corrispondente alla corretta sequenza alfabetica) di Indici( ), avendo cura di scalare di un posto i restanti valori in Indici( ). Il processo è senza dubbio più veloce che non operare direttamente sulla matrice NuoviAbb$, ma, se Indici( ) diviene molto ampia, il sistema risulta ancora lento. Ricorrendo ad un ingegnoso espediente, possiamo creare una nuova struttura, che impiega una nuova matrice, Puntatori( ), nella quale depositiamo l'indice relativo alla posizione di ciascun nominativo in NuoviAbb$:

IndiceIniziale(2)

NuoviAbb$( )	Puntatori( )	Indice( )
(1) Giusti	(4)	(2)
(2) Aloisi	(3)	(3)
(3) Conti	(6)	(6)
(4) Rossi	(5)	(1)
(5) Serra	(0)	(4)
(6) Danti	(1)	(5)

La prima differenza consiste nell'impiego di una variabile semplice IndiceIniziale: il suo valore indica il primo nominativo, in ordine alfabetico, di NuoviAbb$( ). Inoltre, viene impiegato uno 0 in Puntatori(5) per indicare che NuoviAbb$(5) è l'ultimo nominativo, sempre in ordine alfabetico.
La successiva differenza consiste nel contenuto di Indici( ) e di Puntatori( ). Brevemente, Indici( ) contiene la posizione "assoluta" del nominativo in NuoviAbb$( ), mentre Puntatori( ) contiene la posizione "relativa". Vediamo di chiarire il concetto: IndiceIniziale "punta" al primo nominativo in ordine alfabetico, ogni valore di Indici( ) punta al nominativo alfabeticamente successivo, mentre Puntatori( ) ci indica soltanto dove trovare il "prossimo" elemento. La struttura che abbiamo creato usando IndiceIniziale, NuoviAbb$( ) e Puntatori( ), si chiama "lista concatenata" (linked list). La lettura di una lista concatenata avviene con un metodo simile a quello impiegato nel gioco della *Caccia al Tesoro*: un'indicazione ci conduce ad un luogo, dove troviamo una nuova indicazione per il luogo successivo, e così via.
Il grande vantaggio che deriva dall'impiego delle strutture a lista concatenata è la flessibilità. Consideriamo il contenuto della lista dopo l'inserimento di un nuovo nominativo (Bastogi, come nell'esempio precedente):

IndiceIniziale(2)

NuoviAbb$( )	Puntatori( )
(1) Giusti	(4)
(2) Aloisi	(7)
(3) Conti	(6)
(4) Rossi	(5)
(5) Serra	(0)
(6) Danti	(1)
(7) Bastogi	(3)

Solo due elementi della matrice Puntatori( ) sono cambiati:

1. Puntatori(2), che precedentemente puntava a NuoviAbb$(3), adesso punta a NuoviAbb$(7), poiché quest'ultimo è il nominativo alfabeticamente successivo
2. Puntatori(7), che prima non esisteva, adesso contiene un puntatore a NuoviAbb$(3), elemento successivo (alfabeticamente) a NuoviAbb$(7).

Il processo di inserimento consiste, dunque, nel trovare il puntatore al nominativo che precede alfabeticamente quello nuovo, sostituirlo con un puntatore al nuovo elemento, ed "accodare" il vecchio puntatore alla matrice Puntatori( ).
Una caratteristica non molto comoda, nelle liste concatenate, è che la ricerca avviene sempre in modo sequenziale: si parte dal primo elemento finché non si trova quello desiderato.
Abbiamo così ottenuto un risultato a dir poco sconcertante: le variabili "ad indice", che per la loro struttura sono tipicamente ad "accesso diretto", sono state trasformate in una "lista concatenata" con accesso sequenziale. In linguaggi quali il LISP ed il PASCAL, la struttura a "lista concatenata" è tra quelle "incorporate" e non è necessario simularla.
Le strutture a liste concatenate, nonostante gli inconvenienti cui abbiamo accennato, sono molto comode nel trattamento di dati "dinamici" (soggetti a continui cambiamenti). Costituiscono validissimi strumenti, inoltre, nelle applicazioni che concernono il riconoscimento del linguaggio parlato.

Effetti speciali

Una nuova sezione dedicata alla gestione della grafica e degli effetti audio sui computer

Grazie alle recenti evoluzioni nel settore dell'elettronica, le prestazioni sonore e grafiche degli home computer si sono enormemente sviluppate. Probabilmente, il fattore determinante di questo sviluppo è stato proprio l'impiego di questi apparecchi nel settore dei giochi: i progettisti hanno dedicato molto tempo ed energie per offrire sofisticate capacità grafiche. Pur se in modo meno appariscente, anche le caratteristiche sonore e musicali si sono evolute di pari passo. L'importanza degli effetti sonori, in un videogioco, è probabilmente collocabile al terzo posto, subito dopo il concetto del gioco e la grafica. Un uso intelligente di effetti sonori e musicali esaltano il divertimento e l'emozione in tutti i videogiochi.
Sfruttando la capacità degli home computer è anche possibile approfondire la conoscenza della musica. Su molti modelli esistono degli speciali comandi BASIC, che, inseriti in brevi programmi, permettono di suonare un motivo musicale, anche abbastanza complesso, impiegando, addirittura, veri e propri accordi. Alcuni modelli permettono di modificare le caratteristiche del suono riprodotto, per renderlo più piacevole all'orecchio, oppure per simulare gli strumenti musicali tradizionali. In tutti i casi, con un'idonea programmazione, la tastiera del computer può essere sfruttata come quella di un pianoforte, permettendo all'utente di suonare in "tempo reale".
Anche i programmatori meno esperti non dovrebbero incontrare difficoltà nello scrivere brevi e semplici programmi, capaci di creare dei suoni musicali abbastanza sofisticati. Desiderando sfruttare al massimo le capacità sonore del proprio computer può risultare utile acquistare dei programmi scritti appositamente per questa applicazione. Comunque, qualunque sia il sistema adoperato dall'utente per creare la musica, è sempre utile comprendere come il computer generi, elabori e controlli l'output degli effetti sonori.

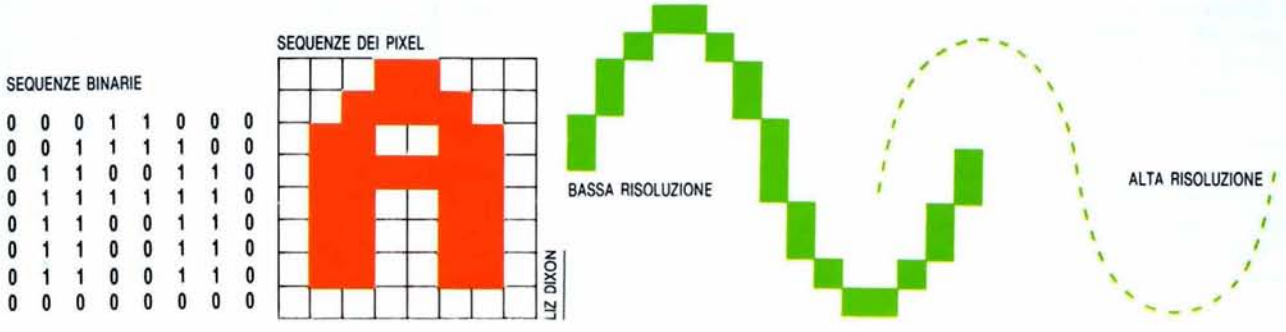
L'oscillatore

L'oscillatore è un circuito elettronico che produce dei segnali ripetitivi. Quando questi segnali vengono amplificati e passati ad un altoparlante, si ottengono dei suoni o delle note di toni diversi. Negli home computer ci

Bassa ed alta risoluzione

Le caratteristiche grafiche dei microcomputer si possono classificare in due categorie: grafica ad alta risoluzione o a bassa risoluzione. Il modo migliore per descrivere la differenza fra le due è di considerare la composizione grafica di un carattere (una lettera, un numero o una forma).
Se si guarda da vicino un carattere riprodotto sullo schermo del monitor si può notare che esso è formato da gruppi di punti, chiamati in inglese "picture elements" (elementi d'immagine), o più comunemente col termine pixel. Nella maggioranza dei computer, i caratteri sono formati da un quadrato di 64 pixel disposti in quadrato di 8 per 8 elementi. Per esempio, la lettera A viene così formata:

Ciascun pixel illuminato sul reticolo è rappresentato, nella memoria, da un 1 e quelli non illuminati da uno 0. Un byte è composto da 8 bit, pertanto ciascuna riga del reticolo viene conservata in una singola locazione di memoria, e per un carattere intero occorrono 8 locazioni.
Le immagini grafiche sono a volte composte da "blocchi" della misura di un reticolo intero oppure da metà o da un quarto di esso. Le immagini composte da questi "mattoni" vengono denominate "a bassa risoluzione". Nella maggioranza degli attuali home computer è possibile creare delle immagini indirizzando ogni singolo pixel (grafica ad alta risoluzione).
Un'indicazione sulla diversa qualità dell'immagine è illustrata qui sotto:



sono fino a quattro di questi oscillatori: ovviamente, più ce ne sono, più note si possono generare contemporaneamente.
Tre sono i parametri che descrivono i suoni generati: la frequenza, l'inviluppo (in cui è incluso anche il volume) e la forma d'onda. L'inviluppo e la forma d'onda verranno trattati più avanti nel corso.

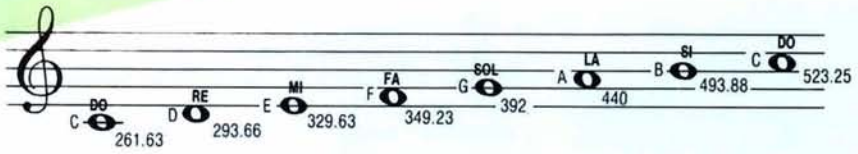
La frequenza

Questa è la caratteristica più importante da controllare, poiché è proprio la frequenza che determina la nota generata. Il termine frequenza indica il numero di volte che un segnale si ripete nell'intervallo di un secondo e viene misurata in Hertz (o cicli al secondo). I suoni percepibili dall'orecchio umano hanno una frequenza compresa, grosso modo, tra 20 Hz e 10.000 Hz. Anche se le frequenze al di sotto di 20 Hz non sono percepibili, le possiamo egualmente impiegare per modificare le caratteristiche di un suono percepibile. Tale tecnica si chiama "modulazione" ed, attualmente, è possibile applicarla (per quanto riguarda la gamma degli home computer) solamente al Commodore 64.

Comunque, non occorre una conoscenza approfondita delle varie frequenze, basta sapere come si ottengono le note musicali. Con alcuni computer il compito è molto facile: esistono comandi BASIC che calcolano automaticamente la frequenza di ciascun suono e non rimane altro che specificare l'intensità del suono oppure indicare la nota desiderata, secondo la notazione anglosassone (in lettere): A, A#, B, ecc. (corrispondenti a LA, LA#, SI, ecc.). In altri computer il sistema è più complicato, ed occorre usare una tabella di conversione, dove sono indicate le frequenze corrispondenti alla nota desiderata, assieme ai valori che devono essere depositati (con l'impiego dell'istruzione POKE), in una particolare locazione di memoria. La tabella riportata qui sotto illustra la conversione delle note corrispondenti alla scala di do centrale e dovrebbe risultare utile a chi volesse cimentarsi nella generazione di musica in linguaggio macchina.

Dalla nota musicale alla frequenza

Per calcolare la frequenza di ciascun suono della scala musicale, basta semplicemente moltiplicare la frequenza della nota distante un semitono inferiore per 1,0594631. Se si esegue la moltiplicazione 12 volte si vedrà che la frequenza originale viene raddoppiata. Ci sono 12 semitoni in un'ottava musicale (l'ottava è la distanza tra due note con lo stesso nome), pertanto se raddoppiamo la frequenza il suono sale di un'ottava. La tabella illustra la conversione da simboli musicali (la scala di do centrale) in frequenze audio



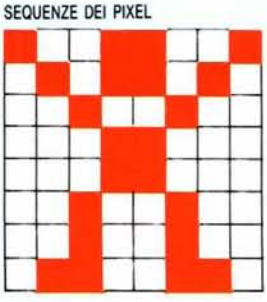
I caratteri definiti dall'utente

Per la creazione di immagini grafiche d'effetto, occorre disporre di caratteri diversi dalla normale serie alfanumerica. Sul Vic-20 e sul Commodore 64 è disponibile una serie speciale di caratteri grafici selezionabili direttamente da tastiera, ma a volte ciò non basta: in quasi tutti gli home computer esiste la possibilità di creare dei caratteri "personalizzati".
In genere, per definire un nuovo carattere occorre depositare una serie di valori binari in apposite zone di memoria. Il nuovo carattere può essere usato mediante l'istruzione PRINT del BASIC.
L'illustrazione qui sotto mostra un esempio di un carattere "personalizzato" e il corrispondente codice in binario:

SEQUENZE BINARIE

128	64	32	16	8	4	2	1
1	0	0	1	1	0	0	1
0	1	0	1	1	0	1	0
0	0	1	0	0	1	0	0
0	0	0	1	1	0	0	0
0	0	0	1	1	0	0	0
0	0	1	0	0	1	0	0
0	0	1	0	0	1	0	0
0	1	1	0	0	1	1	0

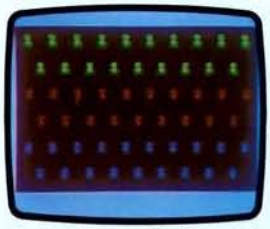
SEQUENZE DEI PIXEL



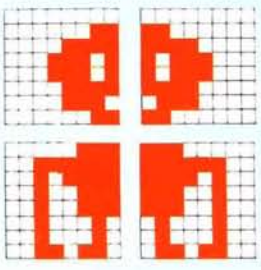
Il metodo impiegato per la composizione di nuovi caratteri varia molto da modello a modello. Per esempio, sullo Spectrum è predisposto un comando USR e tutto quello che occorre è fornire l'appropriata sequenza di valori in binario. Nel Commodore 64, invece, l'utente deve per prima cosa trasferire la serie completa di caratteri dalla ROM alla RAM e successivamente usare l'istruzione POKE per depositare i nuovi valori nelle locazioni di memoria. Comunque, esistono in commercio diversi programmi che aiutano nella formazione di nuovi caratteri o forme.
Per creare immagini di dimensioni maggiori di un singolo carattere si possono raggruppare assieme due o più caratteri "personalizzati". Nelle illustrazioni qui a fianco si può vedere come siano stati costruiti dei piccoli "extraterrestri", impiegando 4 caratteri personalizzati. Il programma, impiegato su di un Commodore 64, visualizza (mediante delle PRINT) i gruppi di caratteri in tre colori diversi. I caratteri sono stati creati usando una breve routine, che verrà presentata più avanti, grazie alla quale i simboli ⊞, ◫, □ e ' ' vengono sostituiti mediante delle POKE.
Anche quando viene usata la grafica "a sprite" (vedere a pag. 152), esiste spesso una limitazione sul numero di sprite visualizzabili sullo schermo, e risulta utile saper creare nuove forme per migliorare la qualità delle immagini riprodotte.

Gli extraterrestri

L'immagine dell'extraterrestre, riportata nelle illustrazioni, è stata ottenuta con l'impiego di 4 caratteri definiti dal programmatore. Questo metodo è utile nei casi in cui non esistano degli "sprite" già "preconfezionati" per l'immagine da generare



IAN McKINNELL



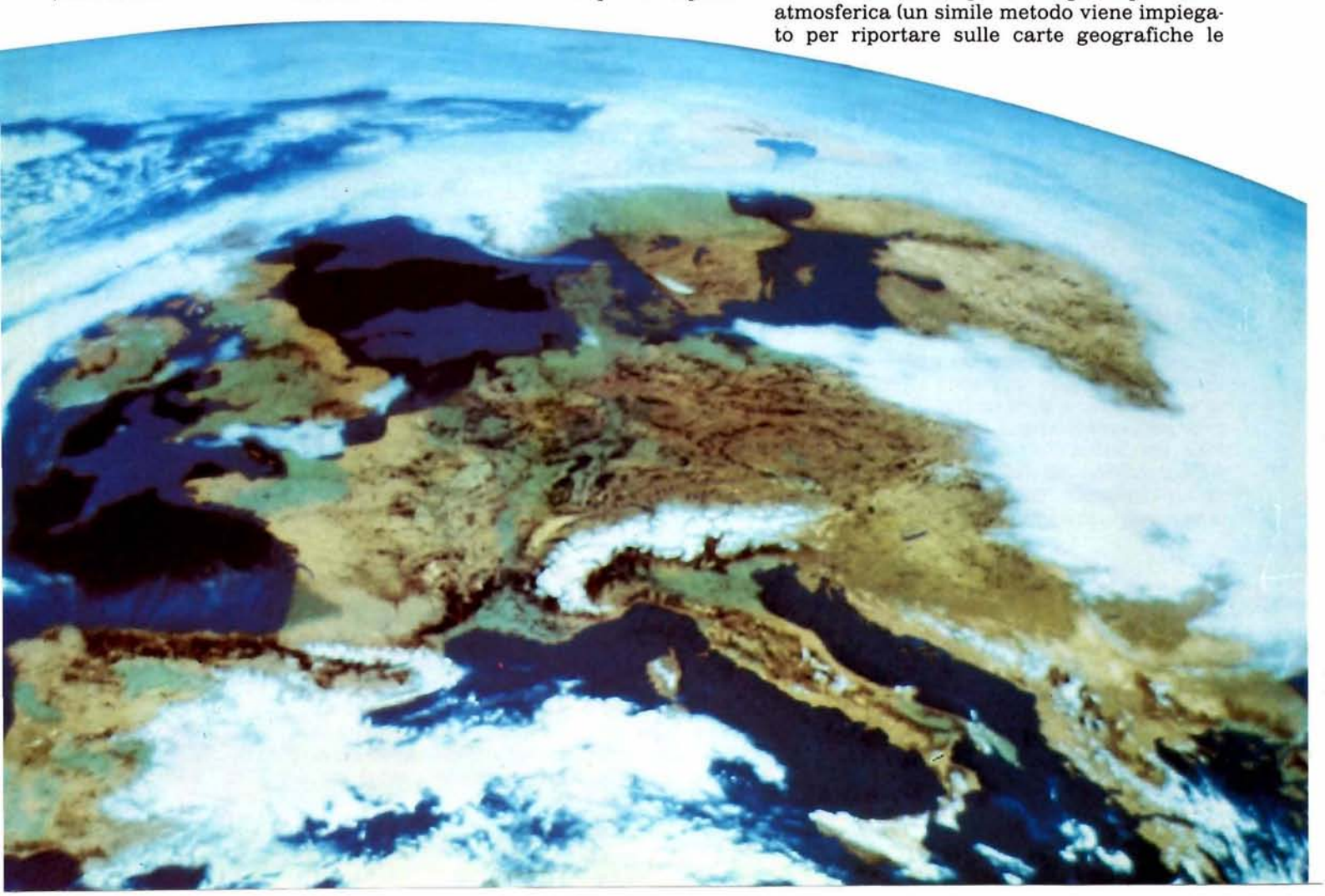
Che tempo farà, domani?

Impieghi del computer nelle previsioni meteorologiche

I risultati di molte delle applicazioni più complesse dell'elaborazione di dati fanno parte ormai della nostra vita quotidiana, anche se non ce ne accorgiamo. Una delle applicazioni più complesse, nella quale l'informatica svolge un ruolo determinante, è quella delle previsioni meteorologiche. Considerando quanto sia complessa la materia, è davvero sorprendente che gli esperti siano così spesso in grado di fornire indicazioni precise sul tempo.

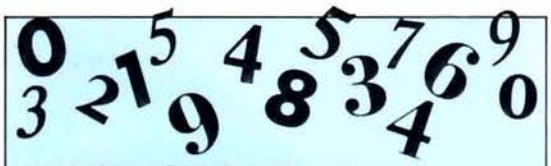
I fattori climatici sono estremamente complessi. Per quanto riguarda l'Europa, le condizioni meteorologiche sono molto influenzate dalla vicinanza del Polo Nord e dall'Oceano Atlantico. La Gran Bretagna, per esempio, essendo affacciata sull'Atlantico, è dominata dagli effetti climatici che si creano sulla superficie oceanica a causa dell'"effetto Coriolis", ossia delle conseguenze della rotazione terrestre. Dobbiamo infatti ricordarci che, all'Equatore, un oggetto posto sulla superficie terrestre viaggia a più di 1.600 Km all'ora! Questo movimento rotatorio, combinato ai normali venti che soffiano dai poli all'equatore, crea forti correnti d'aria provenienti, nel nostro emisfero, dal versante occidentale. Questo costante flusso di aria umida (che sale o scende a seconda delle variazioni locali di temperatura) determina le condizioni del tempo predominanti nel nord Europa. I meteorologi europei basano i loro calcoli su osservazioni rilevate, principalmente, da stazioni per la raccolta di dati situate in posizioni strategiche o su mezzi in movimento sull'Atlantico (navi, boe, palloni meteorologici, aerei). In base alle rilevazioni di queste postazioni ed alle esperienze verificate in passato è possibile fare delle previsioni abbastanza precise su ciò che accadrà quando le perturbazioni si avvicinano alla massa terrestre.

Prima del lancio nello spazio del satellite meteorologico Meteosat 1, avvenuto nel 1976, l'unico metodo che i meteorologi avevano a disposizione, per le previsioni, era quello di riportare manualmente le informazioni provenienti dalle varie stazioni meteorologiche su una cartina geografica, ottenendo un grafico isobarico. Le isobare sono linee immaginarie tracciate tra punti di eguale pressione atmosferica (un simile metodo viene impiegato per riportare sulle carte geografiche le



Immagini dallo spazio

Il satellite Meteosat 2, lanciato nello spazio nel giugno 1981, viaggia in un'orbita geostazionaria (ossia si trova in una posizione fissa rispetto alla Terra) a circa 35.880 km di altezza, sopra l'equatore. Esso raccoglie dati provenienti da numerose stazioni meteorologiche sparse sulla superficie terrestre



MANGIATORI DI NUMERI

Una delle materie nelle quali i maxicomputer adoperati nella ricerca scientifica sono veramente "ferrati" è l'elaborazione di dati puramente numerici, tanto comuni nelle applicazioni scientifiche pure (come per esempio la fisica nucleare) quanto in quelle applicate (come la meteorologia). Teoricamente, sarebbe possibile eseguire calcoli di questo genere anche su home computer, ma il tempo necessario sarebbe immenso, non solo per il numero elevato dei termini nelle equazioni, ma anche per la grandezza dei numeri, che possono avere fino a 30 cifre decimali. Per questo genere di applicazione è dunque essenziale poter disporre di computer molto veloci con moltissima memoria

varie altitudini rispetto al livello del mare). Dalle cartine così ottenute, è possibile dedurre la velocità e la direzione delle correnti d'aria calde e fredde (e relativi cicloni e anticicloni) onde arrivare a quelle che si potrebbero definire "ragionevoli supposizioni" sulle condizioni meteorologiche a venire. Oggi gli uffici meteorologici, comunque, non utilizzano soltanto i grafici isobarici: grazie alla vastissima quantità di dati, conservati nel "database" meteorologico di enormi computer, è possibile produrre anche cartine relative alle temperature medie, alla pioggia, alle ore di sole giornaliere, ecc. Essenzialmente, la preparazione delle mappe particolareggiate sulle condizioni attuali e prossime venture, si avvale ancora delle cartine isobariche, ma a queste si aggiungono anche delle immagini provenienti dal Meteosat 1. Le immagini trasmesse dal satellite, sotto forma di segnali analogici che vengono digitalizzati ed elaborati dal computer, servono a costruire delle speciali carte geografiche, artificialmente ma opportunamente colorate dal computer, per dar maggior risalto ai fattori climatici. Le immagini, trasmesse ad intervalli regolari di circa 4 minuti, riproducono un quadro "vivente" dell'andamento meteorologico, offrendo ai meteorologi la possibilità di osservare le varie situazioni atmosferiche mentre esse si vanno formando.

Il Meteosat 2, che dal giugno del 1981 ha sostituito il precedente satellite meteorologico, viaggia in un'orbita geostazionaria a circa 35.880 km d'altezza, sopra l'equatore. Raccoglie dati provenienti da un elevato numero di stazioni meteorologiche situate in diverse parti del globo, ritrasmettendole a chiunque, in possesso delle necessarie attrezzature, sia abbonato al servizio.

In teoria sarebbe possibile analizzare ed interpretare questi dati, anche se non in "tempo reale", con un home computer, memorizzando su disco i dati ricevuti dal satellite. Tuttavia, poiché i segnali sono di tipo analogico, la ricezione e la conversione di essi, anche se possibile, non è alla portata di tutti; sarebbe infatti necessario installare una speciale antenna parabolica perfettamente allineata con il satellite, oltre a sofisticate apparecchiature di ricezione.



I computer impiegati dagli uffici meteorologici non si limitano all'elaborazione dei segnali provenienti dallo spazio: collegati ad altri grossi centri di elaborazione, situati in più parti del mondo, essi conservano dei modelli meteorologici mondiali, estraendo da questi una grandissima quantità di statistiche, che vanno a formare l'archivio di informazioni "storiche", in base alle quali vengono tracciate le tendenze del tempo locali e mondiali. Non vengono utilizzati solo i dati barometrici, ma anche dettagli sulla velocità e la direzione del vento, sulla pioggia e la temperatura, e non soltanto al livello del mare o della terra, ma anche a diverse altitudini.

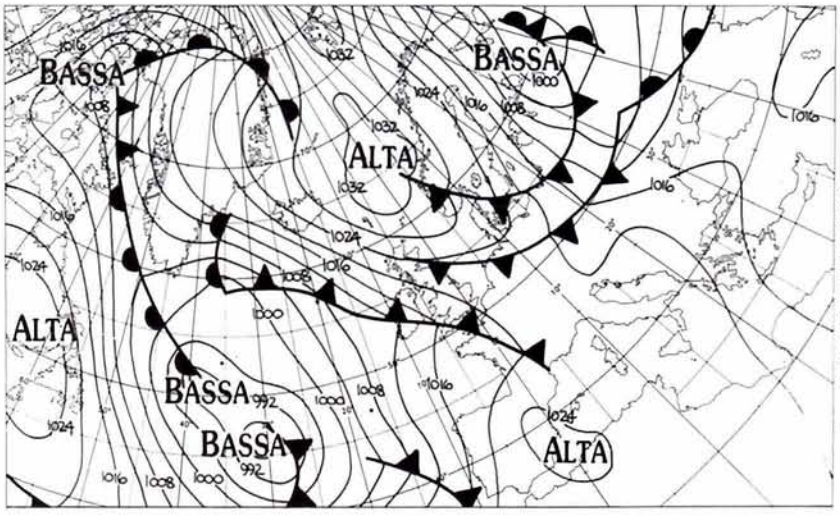
Questi dati sono considerati essenziali per molte attività (per l'agricoltura, per molte industrie e per l'economia e l'ecologia di continenti interi), in quanto è possibile formulare delle previsioni sui cambiamenti, talvolta radicali, nel clima. Grazie a queste enormi "banche dati" è stato possibile rilevare la progressiva distruzione della foresta amazzonica e la crescita delle calotte polari, che potrebbero preannunciare l'inizio di una nuova era glaciale.

Le stazioni terrestri

Le antenne paraboliche per la ricezione dei segnali inviati dai satelliti variano molto in dimensioni ed in complessità: quella illustrata nella foto serve sia per la trasmissione che per la ricezione e non è limitata ai segnali provenienti da satelliti geostazionari. Il sistema di controllo elettronico di cui è dotata permette di seguire accuratamente un satellite in orbita

Le cartine isobariche

Le cartine meteorologiche, che spesso vediamo sui giornali ed alla televisione, sono relative alla pressione barometrica. Le linee concentriche delimitano le aree aventi uguale pressione atmosferica. Nel nostro emisfero, i venti soffiano in direzione antioraria intorno ad un campo di pressione bassa e in direzione oraria intorno ad un campo di pressione alta e la velocità del vento è proporzionale alla distanza tra le isobare



per concessione del METEOROLOGICAL OFFICE

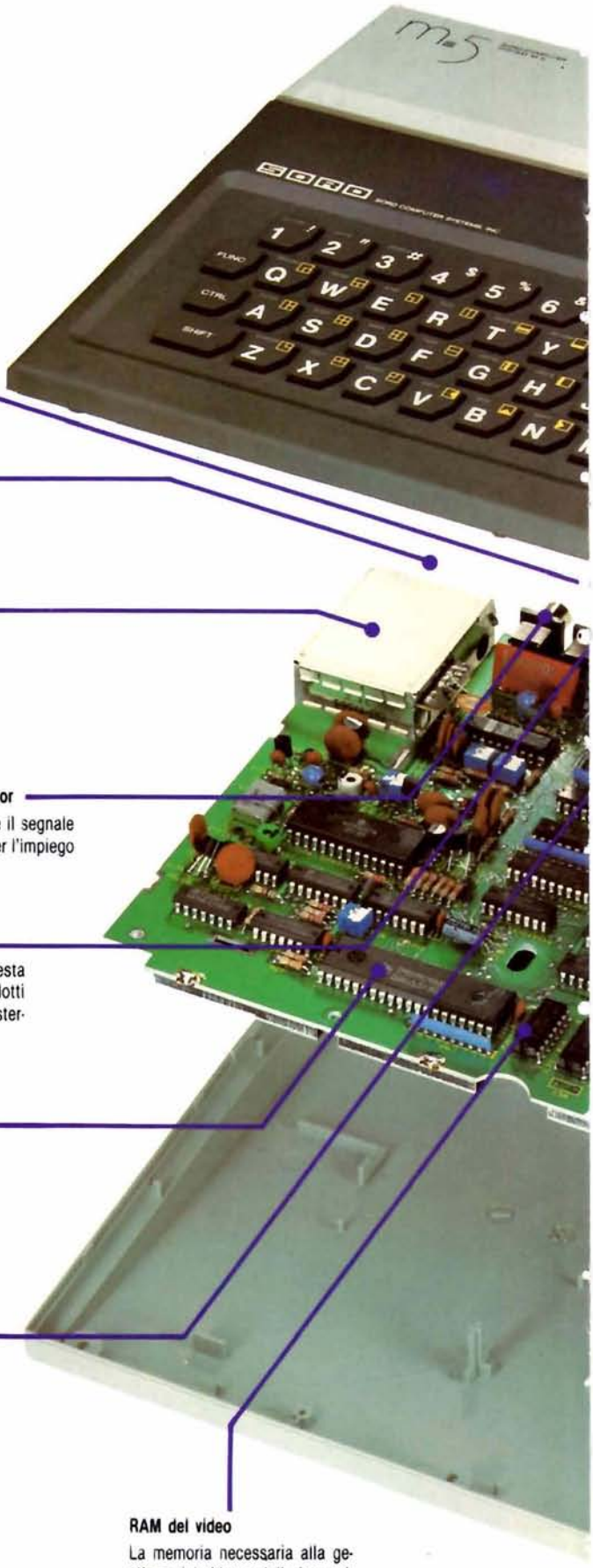
Il Sord M5

Anatomia di un microcomputer a diffusione europea: tecnologie a confronto

Inizialmente, la maggior parte degli home computer veniva progettata negli Stati Uniti. Successivamente, anche la Gran Bretagna è riuscita ad accaparrarsi una buona fetta di questo mercato ed ora anche il Giappone si sta lanciando all'attacco. Il Sord M5 non è il primo microcomputer nipponico, ma certamente il primo ad aver creato un notevole impatto nel settore.
Di costruzione solida e compatta, questo apparecchio ha circa le stesse dimensioni dello Spectrum Sinclair, ma è sensibilmente più pesante e dà una maggiore impressione di robustezza. Le somiglianze non si fermano qui: anche l'M5 è dotato di una CPU Z80A, le istruzioni BASIC possono essere immesse con la semplice pressione di singoli tasti e la conservazione dei programmi (o dati) avviene su nastri a cassetta. Internamente, tuttavia, le differenze sono piuttosto notevoli. Il Sord M5 possiede un'interfaccia Centronics incorporata, una maggiore estensione di memoria RAM (4 Kbyte espandibili a 36) e speciali chip per la grafica e per la generazione di effetti audio.
Per la grafica vengono impiegati gli integrati TI 9918 o TI 9928 (a seconda delle nazioni



La cartuccia ROM
Caratteristica apprezzabile del Sord M5 è la possibilità di utilizzare più linguaggi di programmazione sostituendo la cartuccia ROM. Esistono tre versioni di BASIC: il BASIC-I (per principianti), il BASIC-G (orientato alla grafica) ed il BASIC-F (orientato ad applicazioni scientifiche). Inoltre, è disponibile anche un package, il FALC, che raggruppa in sé più funzioni: lavagna elettronica, gestione di dati e di immagini grafiche. Il FALC consente lo sviluppo di sofisticate procedure, anche nel campo commerciale



Connettore per stampante
Il modello M5 dispone di una interfaccia parallela Centronics incorporata, che permette di utilizzare una vasta gamma di stampanti

Connettore per video TV
Da qui escono i segnali diretti al televisore

Modulatore RF
Il modulatore RF provvede a convertire i segnali prodotti dal chip del video, affinché possano essere utilizzati con un normale televisore

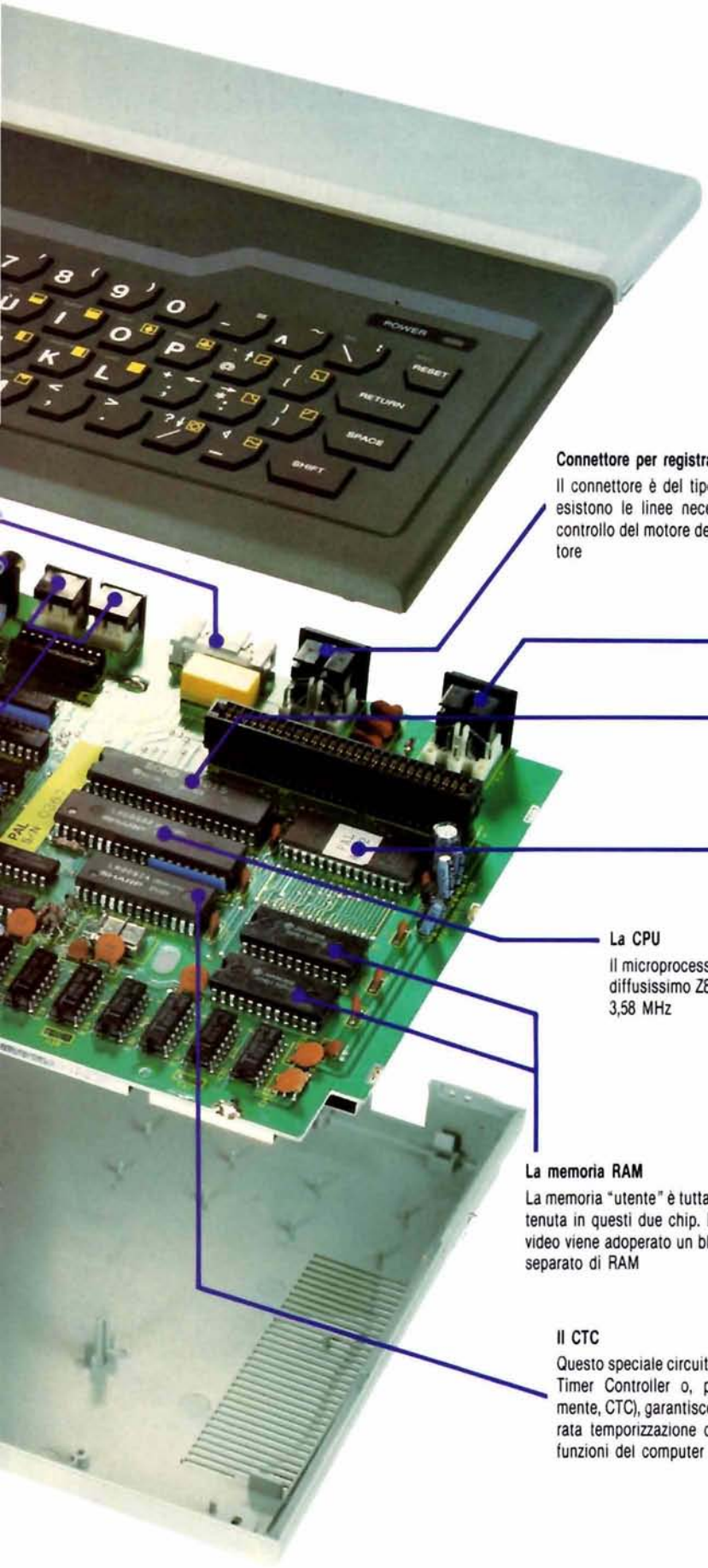
Connettore per monitor
Da questa presa esce il segnale di video composito per l'impiego di un monitor

Uscita audio
I segnali disponibili da questa presa possono essere riprodotti da un amplificatore audio esterno

Il chip del video
Il circuito integrato impiegato per il controllo della visualizzazione è prodotto dalla Texas Instruments e può gestire fino a 32 "sprite" indipendenti

Connettore per joypad
A questa presa si possono collegare due comandi a distanza, simili ai joystick

RAM del video
La memoria necessaria alla gestione del video e delle immagini, 16 Kbyte di RAM, costituisce un blocco separato dalla RAM dei programmi





I joypad
La risposta Sord ai joystick: un disco orizzontale invia al computer segnali per il posizionamento del cursore o di altri "oggetti" sullo schermo. Il funzionamento basato su "interruzioni" della CPU, risulta velocissimo

Connettore per registratore
Il connettore è del tipo DIN, ed esistono le linee necessarie al controllo del motore del registratore

Presa d'alimentazione

Chip "personalizzato"
La Sord impiega un circuito integrato, progettato appositamente, per ridurre al minimo i componenti dell'apparecchio. Questi speciali circuiti vengono chiamati "Custom chip", ossia chip personalizzati

La memoria ROM
Contiene i programmi di gestione interna (per il video, la tastiera ed il registratore)

La CPU
Il microprocessore impiegato è il diffusissimo Z80A, funzionante a 3,58 MHz

La memoria RAM
La memoria "utente" è tutta contenuta in questi due chip. Per il video viene adoperato un blocco separato di RAM

Il CTC
Questo speciale circuito (il Clock Timer Controller o, più brevemente, CTC), garantisce un'accurata temporizzazione di tutte le funzioni del computer

nelle quali è venduto), che offrono una risoluzione di 192 × 256 punti e fino a 16 diversi colori. I "modi" di funzionamento grafico sono quattro ed in tre di essi si possono impiegare fino a 32 "sprite" indipendenti tra loro per quanto riguarda la mobilità sullo schermo. Il set di caratteri visualizzabili comprende sia le maiuscole che le minuscole, oltre ai normali simboli di punteggiatura ed ai numeri. Inoltre, potendo ridefinire a piacimento ciascun carattere, le possibilità sono pressoché illimitate.
È grazie ai chip impiegati per la grafica, gli stessi adoperati nel TI 99/4A (vedere a pag. 189), che le prestazioni del Sord M5 sono così apprezzabili, nonostante la limitata memoria RAM a disposizione. Infatti, per la memoria video non viene impiegata la RAM principale e quest'ultima è totalmente disponibile per i programmi.
Un argomento molto dibattuto, nel settore degli home computer, è il cosiddetto "standard MSX", proposto da un gruppo di ditte

SORD M5

DIMENSIONI

mm. 185 x 70 x 55

PESO

1.000 grammi

CPU

Z80A

CLOCK

3,55 MHz

MEMORIA

8 Kbyte di ROM (espandibile a 16 Kbyte). 20 Kbyte di RAM (di cui 16 Kbyte dedicati alla gestione video), espandibile a 32 Kbyte

VIDEO

I colori disponibili sono 16, utilizzabili anche su diversi "piani", come del resto la grafica "a sprite". Esistono 4 diversi modi di funzionamento grafico

INTERFACCE

Connettore per TV, per registratore a cassette, per cartucce ROM, interfaccia parallela (Centronics) ed uscita audio

LINGUAGGI FORNITI

BASIC-I su cartuccia (per principianti)

ALTRI LINGUAGGI DISPONIBILI

BASIC-G (orientato alla grafica), BASIC-F (per applicazioni scientifiche), FALC (programma applicativo misto)

ACCESSORI COMPRESI

Alimentatore, cavi di connessione per TV e per registratore a cassette, un nastro con 2 giochi, cartuccia del BASIC-I

TASTIERA

55 tasti multifunzione

DOCUMENTAZIONE

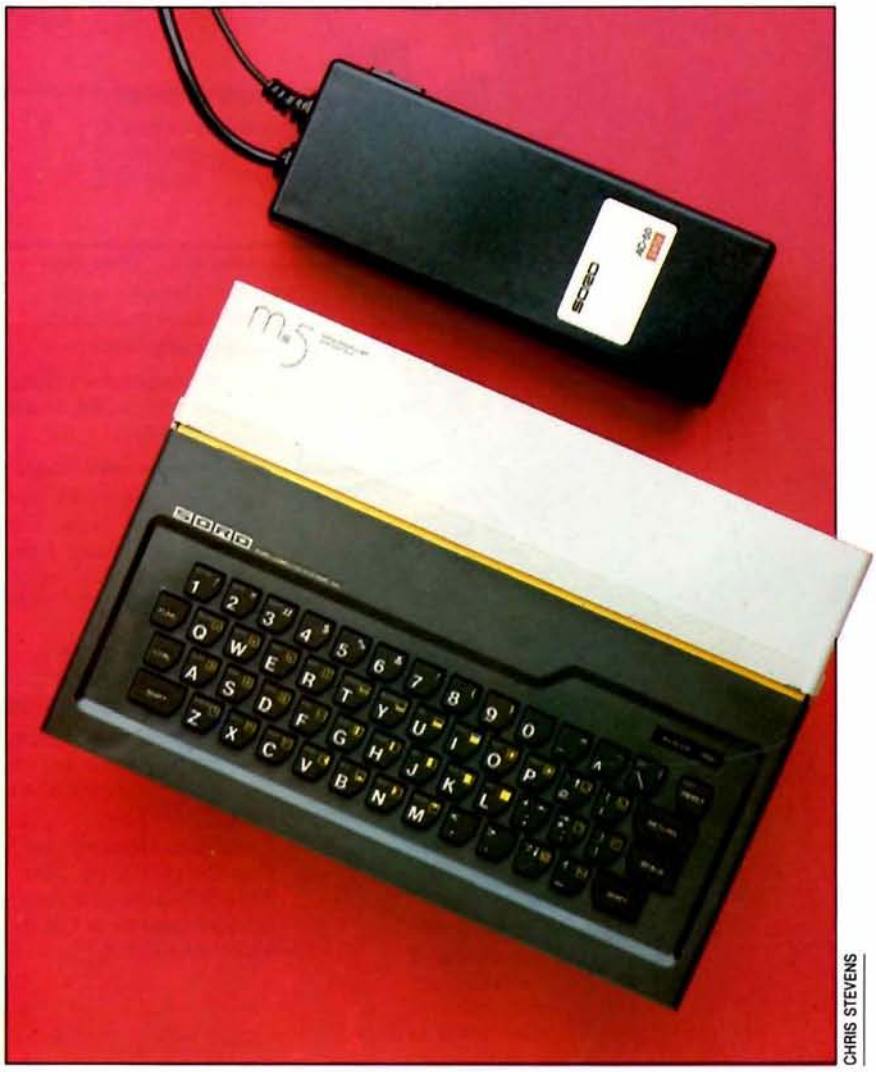
La "Guida per Utenti" originale è composta da una ventina di pagine nelle quali viene spiegato come installare l'apparecchio, come utilizzare i due giochi forniti su nastro. Una pagina è dedicata all'individuazione di eventuali funzionamenti difettosi. Il BASIC non è molto documentato, né vengono fornite molte informazioni su impieghi alternativi delle interfacce per joypad

giapponesi, tra le quali anche la Sord. La proposta prevede che i fabbricanti di home computer si attengano a delle convenzioni ben precise per quanto concerne sia l'hardware che il software (in particolare la versione di BASIC). Ovviamente, lo scopo è quello di consentire il più ampio scambio possibile di software anche tra apparecchi di diverse marche. Per quanto concerne la grafica, il Sord M5 rispetta questi standard.

Invece, per quanto riguarda la generazione degli effetti audio, il Sord (come il Micro BBC) impiega il chip TI 76489 e non l'AY-3-8910 della General Instruments, come suggerito dallo standard MSX. In effetti, il TI 76489 offre maggiori possibilità di controllo sulla qualità dei suoni generati, che non il chip della General Instruments.

Sono disponibili ben tre diverse versioni di BASIC, una discreta quantità di programmi di "utilità", un certo numero di giochi ed altre applicazioni, tutte su cartucce ROM. Poiché le cartucce hanno una capacità di 16 Kbyte, è probabile che vengano sviluppati, in un prossimo futuro, molti buoni programmi applicativi.

Le prestazioni che si ottengono dal Sord M5 giustificano ampiamente il suo prezzo, sensibilmente superiore alla media.



CHRIS STEVENS

La tastiera del Sord M5

La tastiera, benché costruita con tasti di gomma simili a quelli dello Spectrum Sinclair, è adatta anche a chi è abituato a normali tastiere da macchina da scrivere. I tasti, 55 in tutto, sono multifunzione, ossia assumono un diverso significato a seconda del modo grafico selezionato

Il controllo di parità

Come si può controllare l'integrità di dati trasmessi a lunga distanza?

Rispetto agli strumenti analogici, i computer digitali hanno un grosso vantaggio: eventuali errori o imperfezioni che possono verificarsi in tutti i circuiti elettrici, non vengono accumulati durante il percorso del segnale attraverso i vari stadi del circuito (vedere a pag. 239). Tuttavia, trasmettendo serie di dati da un luogo ad un altro, sia per mezzo di interfacce seriali e cavi elettrici, sia lungo le linee telefoniche, può accadere che il rumore elettrico di fondo sia abbastanza elevato da far "scattare" un bit, modificandolo da 1 in 0, o viceversa. Qualora non esista un sistema di verifica, il computer ricevente, in genere, non si accorge di questo tipo di errore ed accetta il dato per buono. Vediamo cosa accade se viene distorto un bit del codice ASCII, corrispondente alla lettera Q:

[l]1010001 (codice ASCII trasmesso, corrispondente alla lettera Q)
[l]1000001 (codice ASCII ricevuto, corrispondente alla lettera A)

Un errore di trasmissione di questo tipo può portare in certe applicazioni, addirittura ad una catastrofe. Tuttavia, come abbiamo già spiegato parlando del codice ASCII, i valori usati vanno da 0 a 127, richiedendo l'impiego di soli 7 bit sugli 8 di un byte. Il bit "più significativo" (convenzionalmente quello più a sinistra) può essere impiegato per rilevare eventuali errori ed in tal caso lo si chiama "bit di parità". A seconda del tipo di controllo, si parla di "parità pari" (even parity) o di "parità dispari" (odd parity). Nel controllo di "parità pari", il bit più significativo del byte viene posto a 0 solo se il numero di "1", nel codice ASCII, è pari, altrimenti esso vale 1. Per esempio, aggiungiamo il bit di parità ai codici ASCII per le lettere A e Q. Nel controllo a parità pari si ha:

[0] 1000001 (Lettera A con parità pari nel codice ASCII)
[1] 1010001 (Lettera Q con parità pari nel codice ASCII)

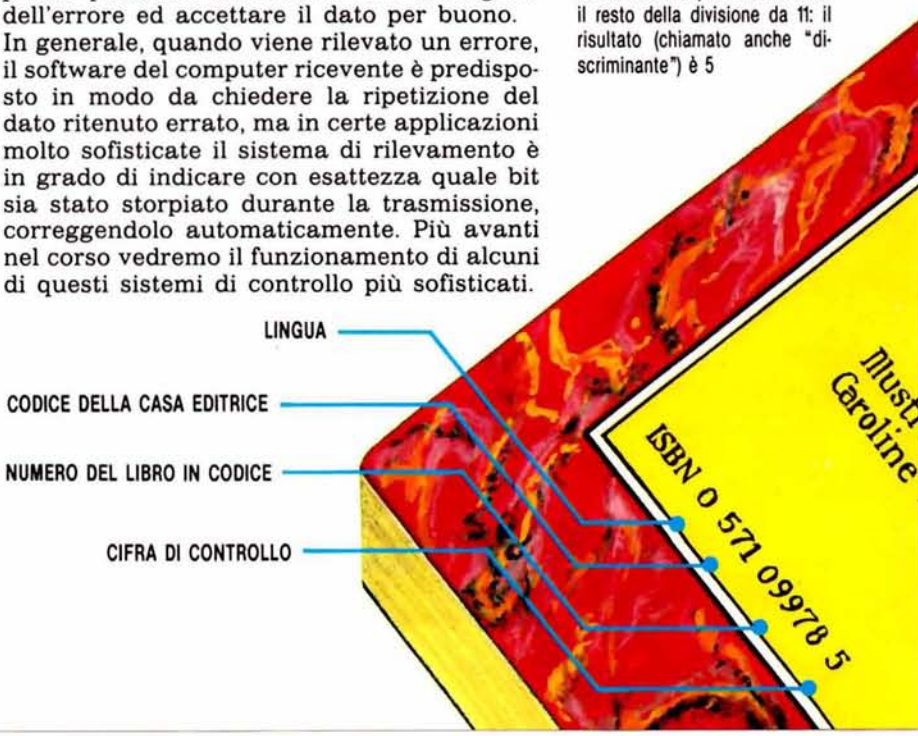
Nel primo caso ci sono due bit "1": al bit di parità viene quindi assegnato il valore 0, in modo che il totale degli "1" sia pari. Nel secondo caso, invece, sono presenti tre "1" e pertanto il bit di parità viene posto ad 1: il totale di bit "1" risulta così pari.
Vediamo adesso cosa succederebbe se il quarto bit nella lettera Q venisse distorto durante la trasmissione:

[1] 1000001 (Lettera Q corrotta)

Il controllo di parità (effettuato con l'impiego di apposito hardware o software) rileva che il numero di bit ad 1, nell'intero byte, è dispari e che pertanto deve esserci un errore (in questo caso il quarto bit è sbagliato). A questo punto, il computer ricevente sarà in grado di scegliere se ignorare semplicemente il dato oppure se chiedere al computer trasmittente di ripetere l'invio del dato.
Dall'esempio si può notare che, anche se vi fosse un errore nello stesso bit di parità, esso verrebbe comunque rilevato.
Questo sistema di controllo, che viene normalmente impiegato durante il trasferimento di dati da un computer ad un altro, è molto semplice, ma al tempo stesso efficace.
Esaminando i codici alfanumerici impiegati nella maggior parte dei piccoli computer, si noterà che il settimo bit del codice viene spesso impiegato, ma per tutt'altro scopo: infatti, poiché è raro che si verifichino errori di trasferimento nella circuiteria interna, il bit di parità viene usato per estendere la gamma di caratteri visualizzabili (in genere i codici, in base decimale, da 128 a 256 individuano particolari simboli grafici).
Il controllo di parità viene normalmente usato solo per la trasmissione di dati a lunga distanza o per il trasferimento di dati alle memorie di massa (nastri o dischetti), suscettibili di errori del tipo esaminato.
Questo metodo di controllo, essendo molto elementare, presenta alcune limitazioni: quando viene rilevato un errore, non è possibile stabilire né quanti né quali bit siano sbagliati e, quindi, il computer ricevente non può apportare nessuna correzione. Inoltre, se i bit "storpiati" sono due (o più), il controllo di parità potrebbe addirittura non accorgersi dell'errore ed accettare il dato per buono.
In generale, quando viene rilevato un errore, il software del computer ricevente è predisposto in modo da chiedere la ripetizione del dato ritenuto errato, ma in certe applicazioni molto sofisticate il sistema di rilevamento è in grado di indicare con esattezza quale bit sia stato storpiato durante la trasmissione, correggendolo automaticamente. Più avanti nel corso vedremo il funzionamento di alcuni di questi sistemi di controllo più sofisticati.

Il carattere di controllo

L'ultimo numero riportato nel codice ISBN serve da controllo ed è l'equivalente del bit di parità in un computer. Esso viene ottenuto moltiplicando il primo numero (in questo caso 0) per 10, il secondo numero per 9 e così via e poi sommando i risultati di queste moltiplicazioni. Si divide il totale ottenuto per 11 e si sottrae il resto della divisione da 11: il risultato (chiamato anche "discriminante") è 5



Scegliere la struttura adatta

Continua lo sviluppo della nostra "rubrica computerizzata"

Nella lezione precedente, abbiamo sviluppato una prima parte del programma, impiegando uno pseudolinguaggio di programmazione, che consentisse una facile trasposizione in istruzioni BASIC. Rivediamo brevemente il problema, e poi illustreremo alcune possibili soluzioni. La nostra definizione del problema era:

INPUT
Un nominativo (in qualsiasi formato)
OUTPUT
1. un nome
2. un cognome

Al primo livello di raffinamento, avevamo individuato sei diversi passi (successivamente ridotti a cinque, con l'eliminazione del sesto):

1. Lettura del nominativo (* * LEGGI * *)
2. Conversione in maiuscole (* * CONVERTI * *)
3. Ricerca dello spazio separatore (* * SPAZIO * *)
4. Lettura del cognome (* * LEGGICOGNOME * *)
5. Lettura del nome (* * LEGGINOME * *)
6. Scarto dei caratteri non alfabetici dal cognome

Tratteremo queste cinque azioni in altrettante subroutine, assegnando loro i nomi riportati tra parentesi. Poiché la maggior parte dei BASIC non consente l'assegnamento di nomi alle subroutine, nella versione definitiva del programma dovremo sostituire i nomi con i numeri di linea corrispondenti alle subroutine ed impiegare le istruzioni GOSUB e RETURN. Nei linguaggi dove è permesso identificare le subroutine con dei nomi, come nel PASCAL, le subroutine vengono chiamate anche "procedure".
Similmente accade per i nomi delle variabili: nella versione definitiva andranno sostituiti con combinazioni di due caratteri: così STRINGATEMP$ diverrà, magari XV$. Durante tutta la fase di sviluppo in pseudolinguaggio, però, conviene adoperare nomi di variabili tali da evitare qualsiasi confusione.
Avevamo già sviluppato, fino alla codifica in BASIC, il secondo passo (la conversione in maiuscole): adesso vediamo come procedere con il resto.

2° Raffinamento

```
3. (Ricerca dello spazio separatore)
   INIZIO
   CICLO finché rimangono caratteri in NOMINATI-
   VO$
     SE carattere = " "
        ALLORA memorizza la posizione in una
        variabile
        ALTRIMENTI passa oltre
     FINE SE
   FINE CICLO
   FINE
```

3° Raffinamento

```
3. (Ricerca dello spazio separatore)
   INIZIO
   LEGGI NOMINATIVO$
   CICLO (finché rimangono caratteri)
     FOR L = 1 fino alla lunghezza di NOMINATIVO$
     LEGGI un carattere di NOMINATIVO$
     SE carattere = " "
        ALLORA LET CONTATORE = posizione del
        carattere
        ALTRIMENTI passa oltre
     FINE SE
   FINE CICLO
   FINE
```

Adesso siamo in grado di codificare la subroutine in BASIC:

```
10 INPUT "IMMETTERE IL NOMINATIVO"; NOMINATIVO$
20 FOR L = 1 TO LEN(NOMINATIVO$)
30 LET CARATT$ = MID$(NOMINATIVO$,L,1)
40 IF CARATT$ = " " THEN CONTATORE = L
50 NEXT L
60 PRINT "POSIZIONE DELL'ULTIMO SPAZIO:";CONTATORE
70 END
```

Si noti che le linee 10 e 60 servono soltanto in fase di verifica (nella versione definitiva vengono omesse) e che nella linea 70 la END andrà modificata in RETURN.
Adesso vediamo il passo 4:

2° Raffinamento

```
4. (Lettura cognome)
   INIZIO
   Assegna i caratteri dopo lo spazio separa-
   tore a COGNOME$
   FINE
```

3° Raffinamento

```
4. (Lettura cognome)
   INIZIO
     LEGGI NOMINATIVO$
     Individua il separatore (ossia chiama la
     subroutine SPAZIO)
     CICLO finché ci sono caratteri dopo lo
          spazio
            LEGGI un carattere ed assegnalo
                a COGNOME$
     FINE CICLO
   FINE
```

Prima di trasporre in BASIC questo passo, è bene soffermarci un momento su qualche particolare. Questa porzione di programma contiene una chiamata alla subroutine SPAZIO e, pertanto, non è possibile effettuare delle prove finché non abbiamo codificato anche questa seconda subroutine. In genere, prima di passare alla codifica nel linguaggio di programmazione, occorre aver completato la stesura del programma in pseudolinguaggio. Naturalmente, possiamo sempre introdurre delle variabili "di comodo", che simulino la presenza delle parti di programma non ancora codificate. Per esempio, mancando le subroutine LEGGI e SPAZIO, possiamo procedere alla verifica del modulo 4 in questo modo:

```
10 LET NOMINATIVO$ = "LUIGI BANCHI"
20 LET CONTATORE = 6
30 FOR L = CONTATORE TO LEN(NOMINATIVO$)
40 LET COGNOME$ = COGNOME$ + MID$(NOMINATIVO$,L,1)
50 NEXT L
60 PRINT "IL COGNOME È ";COGNOME$
70 END
```

Vediamo adesso come sviluppare il passo 5, ricordandoci che il nome è composto da tutti i caratteri precedenti lo spazio separatore, dopo aver eliminato eventuali segni non alfabetici:

2° Raffinamento

```
5. (Lettura nome)
   INIZIO
   CICLO finché i caratteri di NOMINATIVO$ sono
         precedenti allo spazio separatore
         Esamina un carattere
         SE il carattere non è una lettera
            ALLORA passa al successivo
            ALTRIMENTI aggiungilo a NOME$
         FINE SE
   FINE CICLO
   FINE
```

3° Raffinamento

```
5. (Lettura nome)
   INIZIO
   CICLO finché non si raggiunge il valore
         di CONTATORE
     LET STRINGATEMP$ = carattere di posizione
         L in NOMINATIVO$
     SE STRINGATEMP$ non è una lettera
        ALLORA passa al successivo carattere
        ALTRIMENTI NOME$ = NOME$ + STRINGATEMP$
     FINE SE
   FINE
```

Per meglio confrontare la codifica in BASIC con la struttura appena vista, omettiamo i numeri di linea:

Codifica

```
5. (Lettura nome)
   REM INIZIO
   REM CICLO
        FOR L = 1 TO CONTATORE – 1
             LET STRINGATEMP$ = MID$(NOMINATIVO$,L,1)
             LET CODICE = ASC(STRINGATEMP$)
              IF CODICE > 64 THEN NOME$ = NOME$
                + CHR$(CODICE)
             REM FINE SE
        NEXT L : REM FINE CICLO
   REM END
```

L'effettiva codifica sarà:

```
10 FOR L = 1 TO CONTATORE – 1
20 LET STRINGATEMP$ = MID$(NOMINATIVO$,L,1)
30 LET CODICE = ASC(STRINGATEMP$)
40 IF CODICE > 64 THEN NOME$ = NOME$ + CHR$(CODICE)
50 NEXT L
60 END
```

In questa forma, però, non è possibile effettuare nessuna verifica del modulo: non è prevista l'immissione di NOMINATIVO$, né del valore di CONTATORE, né la visualizzazione di alcun risultato.
Ovviamente, nel programma finale, questi valori proverranno da altre subroutine ed il risultato (NOME$) verrà utilizzato altrove nel programma. Questo scambio di dati fra le varie parti di un programma è chiamato "passaggio di parametri" e deve essere progettato con estrema cura. La variabile CONTATORE, per esempio, viene impiegata in più parti del programma: se il suo contenuto venisse modificato dall'esecuzione di più subroutine, potremmo incorrere in errori tali da compromettere il funzionamento dell'intero programma. Consideriamo due ipotetiche subroutine di un programma:

Frammento della subroutine X

```
FOR L = 1 TO LEN(PAROLA$)
LET CARATT$ = MID$(PAROLA$,L,1)
IF CARATT$ = "." THEN GOTO 1550
NEXT L
```

Frammento della subroutine Y

```
FOR Q = 1 TO LIMITE
LET A(L) = P(Q)
NEXT Q
```

In questo secondo frammento, l'indice della matrice P(Q) varia da 1 a LIMITE: ma che valore verrà trasferito in A(L) se l'indice L è stato modificato dal frammento precedente, nel quale il valore di L dipende dalla posizione del carattere "." in PAROLA$? In questo particolare esempio, viene eseguito un salto (GOTO) dall'interno di un ciclo FOR-NEXT; questa è una pratica altamente sconsigliata, anche se talvolta necessaria.
Per evitare confusione nell'impiego delle va-

riabili, è buona norma, fin dal primo livello di stesura in pseudolinguaggio, mantenere una lista dei nomi impiegati per le variabili, meglio ancora se corredata da una breve annotazione sul probabile contenuto di ciascuna di esse. Alcuni linguaggi (ma non il BASIC) permettono di distinguere le variabili in **globali** o **locali**: quelle **globali** si riferiscono a tutte le parti del programma, mentre quelle **locali** soltanto a porzioni di esso. Per esempio, gli indici impiegati nei cicli FOR-NEXT vengono generalmente definiti **locali** ed, in genere, occorre "inizializzare" tali variabili (ossia assegnare loro un valore iniziale). Benché il BASIC provveda automaticamente all'azzeramento delle variabili quando inizia l'esecuzione di un programma, conviene sempre prevedere una subroutine di inizializzazione delle variabili.
Avendo, a questo punto, definito la struttura generale del programma e visto come codificarne alcune delle parti, dobbiamo dedicarci alla struttura dei dati da elaborare, ossia alla struttura dei "record". Per far ciò dobbiamo adoperare dei termini molto ricorrenti nell'informatica: file, record, e campo (field). Il termine "file" indica l'archivio nel quale vengono raccolti tutti i dati (nel nostro caso tutti i nomi, indirizzi e numeri di telefono) ed è composto da una certa quantità di "record". Ogni record si riferisce ad un particolare nominativo ed è composto, a sua volta, da più "campi" (nel nostro caso, ogni record contiene un campo per il nominativo, uno per l'indirizzo, ecc.)
Il contenuto di un record della nostra "rubrica computerizzata" potrebbe essere:

Pietro Esmeraldi
Via Cavour 56 Bologna
051-455444

Possiamo considerare questo record suddiviso in tre campi: il nominativo, composto da lettere dell'alfabeto ed, eventualmente da eventuali apostrofi (per esempio Michele D'Anna); l'indirizzo, composto da lettere e da numeri; il numero telefonico, composto da numeri (eventualmente anche da un trattino di separazione tra prefisso e numero).
Nella stesura di un programma è molto importante, fin da principio, definire con precisione la struttura dei dati, poiché essa influenza notevolmente la struttura del programma e la scelta delle istruzioni più adatte da impiegare. Per esempio, potremo decidere di utilizzare un'unica stringa di caratteri per ciascun record. Il problema, con questo tipo di struttura, è che risulta difficile individuare con esattezza la posizione di ogni elemento. Supponiamo, infatti, che un record, costituito da un'unica stringa, contenga:

Enrico Maria Cappelletti
Via del Colle 56a
SCANDICCI
055 250924

Durante una ricerca del numero telefonico del Sig. Cappelletti, è sufficiente considerare gli ultimi 10 caratteri della stringa, per estrarre il numero di telefono? E se il numero fosse più lungo oppure contenesse anche il prefisso indicativo della nazione?
Conviene dunque considerare il numero telefonico come un campo a sé stante.
Un possibile metodo per accedere ai vari campi di un record potrebbe consistere nel creare un campo aggiuntivo, che serva da indice. Individuato il nominativo, potremmo risalire all'indice da usare per estrarre il resto delle informazioni. Per esempio:

ENRICO MARIA CAPPELLETTI	Campo del NOME
VIA DEL COLLE 56A	Campo dell'INDIRIZZO
SCANDICCI	Campo della CITTÀ
055 250924	Campo del TELEFONO
015	Campo dell'INDICE

Supponiamo, adesso, di utilizzare una variabile indicizzata per ogni campo del record: per trovare il numero di telefono di una persona, possiamo scorrere la variabile che contiene il nominativo fino a trovare quello da noi cercato e, utilizzando il medesimo indice della variabile, accedere così all'elemento (il quindicesimo nell'esempio) della variabile TELEFONO.
Nella prossima lezione esamineremo le tecniche associate alla ricerca di dati nelle liste.

Esercizio

* Si supponga che il formato dei record, per la nostra rubrica computerizzata, sia:

Campo del NOMINATIVO
Campo dell'INDIRIZZO
Campo della CITTÀ
Campo della NAZIONE
Campo del NUMERO TELEFONICO

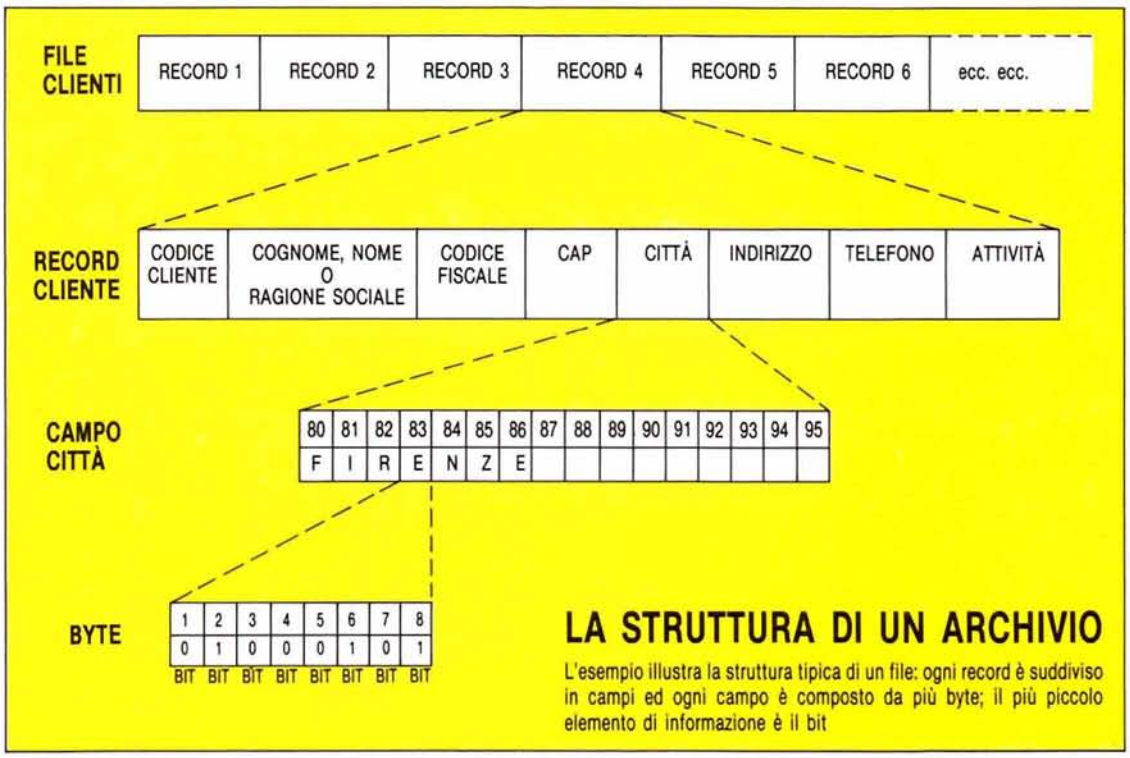
Si supponga, inoltre, che una delle opzioni del menu principale sia:

5. INSERIMENTO DI UN NUOVO NOMINATIVO

Selezionando l'opzione 5, il programma presenterà una serie di domande, necessarie al nuovo inserimento. Il risultato di questa procedura dovrebbe essere l'assegnamento degli appropriati valori alle seguenti matrici (variabili indicizzate):

1. Un valore nella matrice dei nominativi
2. Un valore nella matrice degli indirizzi
3. Un valore nella matrice delle città
4. Un valore nella matrice della nazione
5. Un valore nella matrice dei numeri telefonici

L'esercizio consiste nello sviluppo di questa parte del programma, in pseudolinguaggio di programmazione, mettendo in pratica gli insegnamenti finora esposti.



LA STRUTTURA DI UN ARCHIVIO

L'esempio illustra la struttura tipica di un file: ogni record è suddiviso in campi ed ogni campo è composto da più byte; il più piccolo elemento di informazione è il bit

Basic & Basic

SPECTRUM

Passo 3

```
10 INPUT "IMMETTERE UN NOMINATIVO";F$
15 LET CONTATORE = 0
20 FOR L = 1 TO LEN F$
30 LET C$ = F$(L)
40 IF C$ = " " THEN LET CONTATORE = L
50 NEXT L
60 PRINT "POSIZIONE DELL'ULTIMO SPA-
   ZIO";CONTATORE
70 STOP
```

Nello sviluppo del programma per la rubrica computerizzata, verranno largamente impiegate le funzioni MID$, LEFT$ e RIGHT$. Poiché lo Spectrum non le possiede, occorre effettuare le seguenti sostituzioni:

```
LEFT$(F$,N)    → F$( TO N)
RIGHT$(F$,N)   → F$(LEN(F$) — N + 1 TO )
MID$(F$,P,N)   → F$(P TO P + N — 1)
MID$(F$,P,1)   → F$(P)
```

Si noti che nel BASIC dello Spectrum i nomi di variabili alfanumeriche sono limitati ad un sola lettera dell'alfabeto, seguita dal simbolo "$".

Passo 4

```
5 LET S$ = ""
10 LET F$ = "LUIGI BANCHI"
20 LET CONTAT = 4
30 FOR L = CONTAT + 1 TO LEN F$
40 LET S$ = S$ + F$(L)
50 NEXT L
60 PRINT "IL COGNOME È:";S$
70 STOP
```

Passo 5

```
5 LET C$ = ""
10 FOR L = 1 TO CONTAT — 1
20 LET T$ = F$(L)
30 LET CARAT = CODE T$
40 IF CARAT > 64 THEN LET C$ = C$ +
   CHR$ CARAT
50 NEXT L
60 STOP
```

Poiché i nomi delle variabili alfanumeriche devono essere di una sola lettera, nella versione per lo Spectrum alla nostra variabile NOMINATIVO$ corrisponde F$, mentre a NOME$ corrisponde C$.

Frammento della subroutine X

```
FOR L = 1 TO LEN P$
LET C$ = W$(L)
IF C$ = "." THEN GOTO 1550
NEXT L
```

Frammento della subroutine Y

```
FOR Q = 1 TO LIMITE
LET A(L) = P(Q)
NEXT Q
```

VARIABILI

Solo il Micro BBC, tra gli home computer più diffusi, permette l'impiego di nomi completi (come COGNOME$) per le variabili alfanumeriche. Soltanto nello Spectrum le variabili alfanumeriche possono avere anche nomi completi. Il Dragon 32, il Vic-20 ed il Commodore 64 permettono l'impiego di nomi completi, ma sono significativi soltanto i primi due caratteri del nome, cosicché NOMINATIVO$ è un nome valido, ma è come se impiegassimo soltanto NO$.

Nell'Oric-1 i nomi delle variabili non possono essere più lunghi di due caratteri (il primo una lettera, il secondo un numero), mentre nel Lynx si può usare una sola lettera, ma, a seconda che essa sia maiuscola o minuscola, identifica variabili diverse.

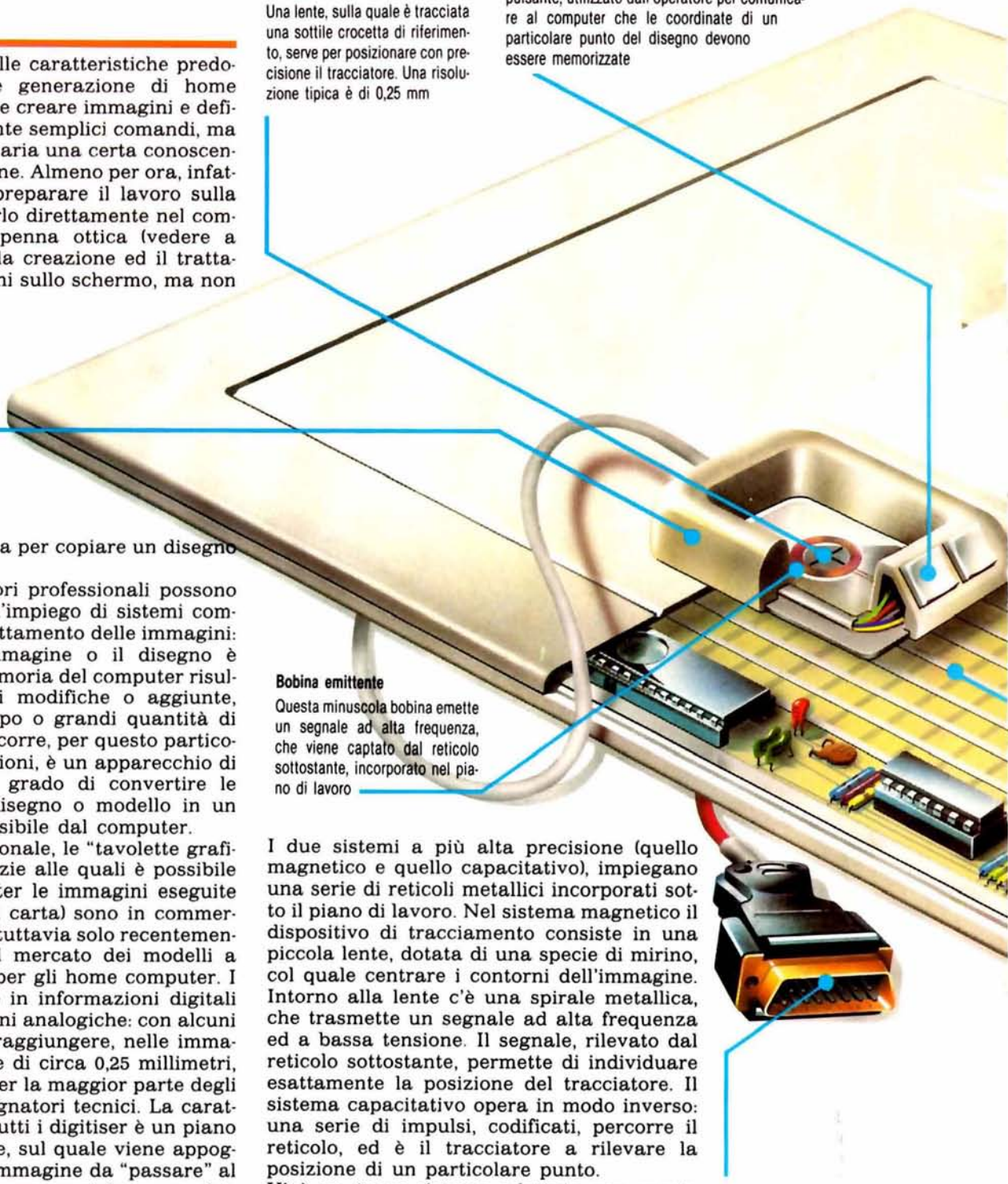
Le tavolette grafiche

Memorizzare disegni ed immagini può essere molto semplice, con i "digitiser"

La grafica è una delle caratteristiche predominanti nell'attuale generazione di home computer. È possibile creare immagini e definirne i colori mediante semplici comandi, ma per fare ciò è necessaria una certa conoscenza di programmazione. Almeno per ora, infatti, non è possibile preparare il lavoro sulla carta per poi passarlo direttamente nel computer. L'uso della penna ottica (vedere a pagina 156) facilita la creazione ed il trattamento delle immagini sullo schermo, ma non può essere adoperata per copiare un disegno eseguito su carta.

Numerosissimi settori professionali possono trarre beneficio dall'impiego di sistemi computerizzati per il trattamento delle immagini: una volta che l'immagine o il disegno è conservato nella memoria del computer risulta facile apportarvi modifiche o aggiunte, senza sprecare tempo o grandi quantità di carta. Quello che occorre, per questo particolare tipo di applicazioni, è un apparecchio di immissione dati in grado di convertire le linee e curve del disegno o modello in un linguaggio comprensibile dal computer.

Nell'ambito professionale, le "tavolette grafiche" o digitiser (grazie alle quali è possibile trasferire al computer le immagini eseguite precedentemente su carta) sono in commercio da molto tempo, tuttavia solo recentemente sono apparsi sul mercato dei modelli a basso costo, adatti per gli home computer. I digitiser convertono in informazioni digitali le forme e le immagini analogiche: con alcuni modelli è possibile raggiungere, nelle immagini, una risoluzione di circa 0,25 millimetri, più che sufficiente per la maggior parte degli ingegneri e dei disegnatori tecnici. La caratteristica comune a tutti i digitiser è un piano di lavoro orizzontale, sul quale viene appoggiato il disegno o l'immagine da "passare" al computer; mediante una speciale penna elettronica o un "mirino" graduato, se ne seguono i contorni ed i circuiti interni al digitiser provvedono a convertire la posizione di ciascun punto in coppie di coordinate, che vengono memorizzate dal computer.



Il "mirino"
Una lente, sulla quale è tracciata una sottile crocetta di riferimento, serve per posizionare con precisione il tracciatore. Una risoluzione tipica è di 0,25 mm

Pulsanti di comando
Generalmente, sul tracciatore è presente almeno un pulsante, utilizzato dall'operatore per comunicare al computer che le coordinate di un particolare punto del disegno devono essere memorizzate

Il tracciatore
Chiamato anche "cursore", questo strumento viene spostato manualmente sui contorni del disegno da "digitalizzare"

Bobina emittente
Questa minuscola bobina emette un segnale ad alta frequenza, che viene captato dal reticolo sottostante, incorporato nel piano di lavoro

I due sistemi a più alta precisione (quello magnetico e quello capacitativo), impiegano una serie di reticoli metallici incorporati sotto il piano di lavoro. Nel sistema magnetico il dispositivo di tracciamento consiste in una piccola lente, dotata di una specie di mirino, col quale centrare i contorni dell'immagine. Intorno alla lente c'è una spirale metallica, che trasmette un segnale ad alta frequenza ed a bassa tensione. Il segnale, rilevato dal reticolo sottostante, permette di individuare esattamente la posizione del tracciatore. Il sistema capacitativo opera in modo inverso: una serie di impulsi, codificati, percorre il reticolo, ed è il tracciatore a rilevare la posizione di un particolare punto.

Vi è un terzo sistema adottato per questo apparecchio: quello acustico. Il tracciatore, in questo caso, possiede una certa carica elettrostatica, che si scarica a contatto col piano di lavoro, generando una piccola scintilla: il tempo necessario all'onda acustica

Connettore
I digitiser vengono collegati al computer utilizzando normali porte seriali o parallele

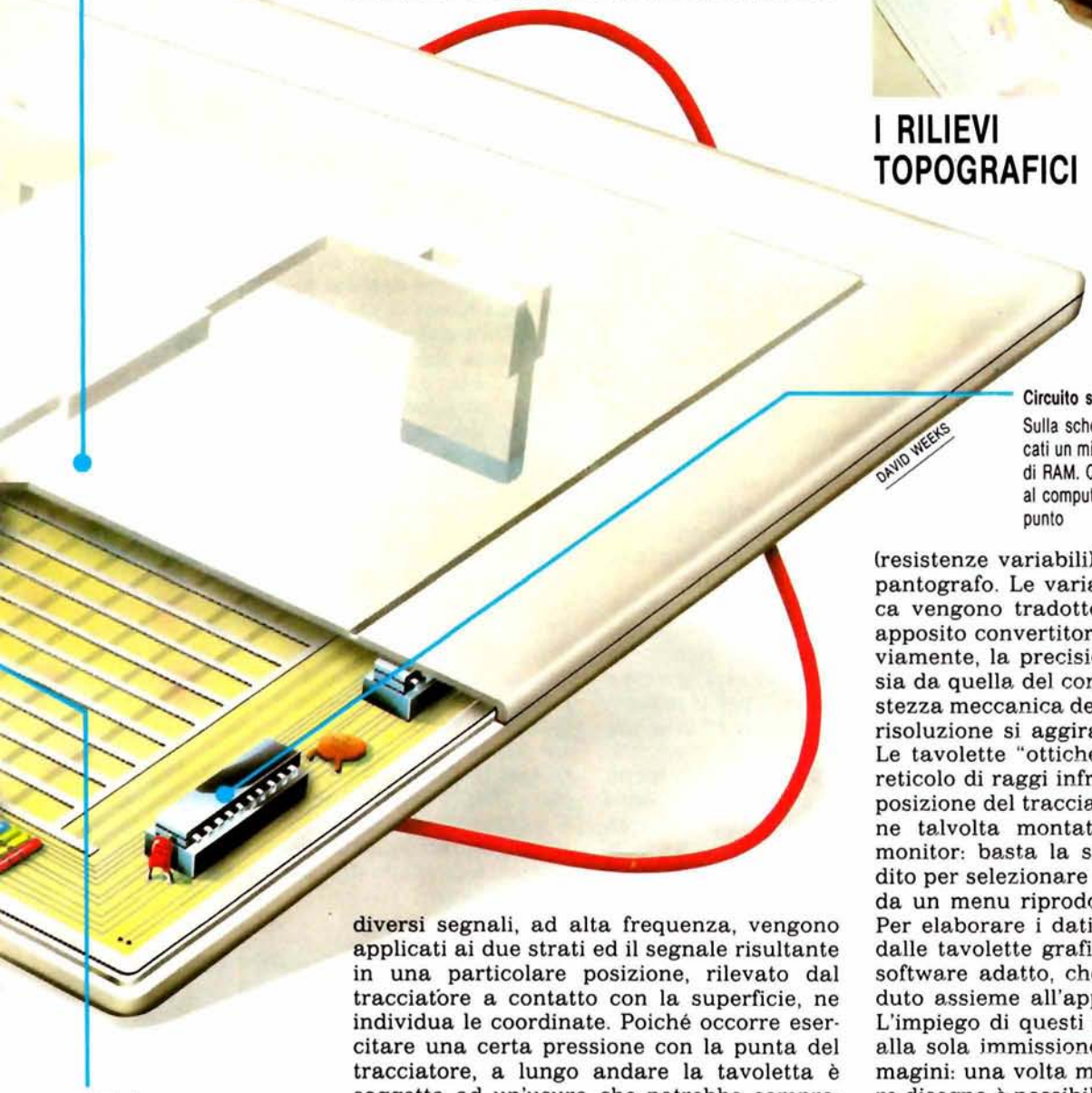


SIMON LEWIS

I RILIEVI TOPOGRAFICI

Una delle applicazioni più diffuse dei digitiser è senz'altro nelle rilevazioni topografiche. Anziché immettere manualmente interminabili serie di coordinate, basta seguire i contorni di un'immagine per memorizzarne la forma e le dimensioni. Oltre a una maggior precisione e semplicità d'impiego, simili sistemi consentono sensibili risparmi di tempo

per raggiungere due captatori microfonici permette di individuare con esattezza la posizione del tracciatore. Con questo sistema è possibile passare al computer immagini tridimensionali, sfruttando un terzo segnale.
Per quello che riguarda applicazioni più generiche e meno professionali, esistono delle tavolette grafiche "a pressione". Il principio di funzionamento è simile a quelli precedenti: il disegno è collocato su di un piano di lavoro ed i suoi contorni "ripassati" con una speciale penna. Sotto al piano di lavoro ci sono due fogli di materiale conduttore, tra i quali è interposto uno strato isolante reticolare. Due



DAVID WEEKS

Piano di lavoro

Il foglio sul quale è riprodotto il disegno o l'immagine da "digitalizzare" viene adagiato e fissato sul piano di lavoro. In alcuni casi, il foglio viene fissato impiegando una carica elettrostatica ed è estremamente importante che esso non subisca alcuno spostamento durante l'immissione delle coordinate

Circuito stampante

Sulla scheda del circuito stampato sono collocati un microprocessore e alcuni chip di ROM e di RAM. Questi componenti servono per fornire al computer le coppie di coordinate di ciascun punto

Reticolo

Sotto al piano di lavoro è incorporato uno speciale reticolo che serve per rilevare la posizione del tracciatore. Non è necessario che le dimensioni di ciascun elemento del reticolo siano molto ridotte, poiché il software provvede a calcolare, per estrapolazione, anche minimi spostamenti della bobina posta sul tracciatore

diversi segnali, ad alta frequenza, vengono applicati ai due strati ed il segnale risultante in una particolare posizione, rilevato dal tracciatore a contatto con la superficie, ne individua le coordinate. Poiché occorre esercitare una certa pressione con la punta del tracciatore, a lungo andare la tavoletta è soggetta ad un'usura che potrebbe comprometterne il corretto funzionamento, anche se ciò accade raramente. Il grado di precisione offerto dalle tavolette grafiche a pressione è adeguato alla risoluzione grafica ottenibile sulla maggior parte degli home computer.
Il tipo più semplice ed economico di digitiser è quello "a pantografo", basato sullo stesso principio del tradizionale accessorio da disegno. Il sistema di rilevazione delle coordinate si basa sull'impiego di due potenziometri (resistenze variabili) montate sugli snodi del pantografo. Le variazioni di tensione elettrica vengono tradotte in valori binari da un apposito convertitore analogico/digitale. Ovviamente, la precisione dei risultati dipende sia da quella del convertitore, sia dalla robustezza meccanica dell'insieme (tipicamente la risoluzione si aggira sul 5%).
Le tavolette "ottiche", invece, impiegano un reticolo di raggi infrarossi per individuare la posizione del tracciatore. Questo sistema viene talvolta montato sullo schermo di un monitor: basta la semplice pressione di un dito per selezionare una particolare funzione da un menu riprodotto sullo schermo.
Per elaborare i dati generati dai digitiser, e dalle tavolette grafiche in genere, occorre il software adatto, che viene solitamente venduto assieme all'apparecchio.
L'impiego di questi apparecchi non si limita alla sola immissione dei dati relativi ad immagini: una volta memorizzato un particolare disegno è possibile richiamarlo, modificarlo o colorarlo a piacimento. In alcuni casi una porzione del piano di lavoro è utilizzata per selezionare le principali operazioni grafiche (come ad esempio la scelta dei colori) o, addirittura, per richiamare e memorizzare le immagini. Una tipica applicazione delle tavolette grafiche ad alta risoluzione è nella produzione dei cartoni animati (vedere a pag. 181), dove vengono impiegate come principale forma di immissione dei dati.

Gottfried Leibniz

Gli scienziati di oggi riscoprono il lavoro di un loro collega del '600



per concessione del SCIENCE MUSEUM

1646
Nasce il 1 luglio a Lipsia

1661
Si iscrive all'Università di Lipsia, dove ottiene un diploma a soli 17 anni

1666
Avvocato e diplomatico, pubblica la sua "Dissertazione sull'arte combinatoria"

1672
Si trova a Parigi. Formula il "principio di ragione sufficiente"

1673
Presenta la sua calcolatrice meccanica alla Società Reale inglese

1675
Inventa il calcolo infinitesimale, indipendentemente da Newton

1676
Considera la fisica dinamica sotto l'aspetto di energia cinetica

1678
Diviene bibliotecario e consigliere del Duca di Hannover

1679
Sviluppa l'aritmetica binaria

1689
Visita diverse città italiane (Venezia, Padova, Firenze, Roma e Napoli)

1700
Su sua iniziativa, viene fondata l'Accademia delle Scienze di Berlino

1714
È chiamato a stabilire il diritto di successione al trono d'Inghilterra, alla morte della Regina Anna

1716
Muore ad Hannover, il 14 novembre

Si può dire che Gottfried Wilhelm Leibniz sia stato uno dei massimi scienziati del suo tempo. Nato a Lipsia nel 1646, morì 70 anni dopo ad Hannover. Durante il corso della sua vita egli inventò il calcolo infinitesimale, si dedicò alla fisica dinamica, contribuì allo sviluppo della geologia, della teologia, della storia e della filosofia. Molte delle sue felici intuizioni sono state utilizzate, a distanza di secoli, per la creazione dei computer.

Poco dopo aver compiuto 20 anni, forse amareggiato dal fatto che l'Università di Lipsia si era rifiutata di conferirgli una laurea in Giurisprudenza (per via della sua giovane età), Leibniz iniziò a viaggiare per il mondo. Non godendo di alcuna rendita familiare, per potersi dedicare alla ricerca scientifica dovette lavorare in campi spesso totalmente estranei alla scienza: da giovane fece l'avvocato ed il diplomatico; più avanti negli anni divenne bibliotecario e consigliere di corte.

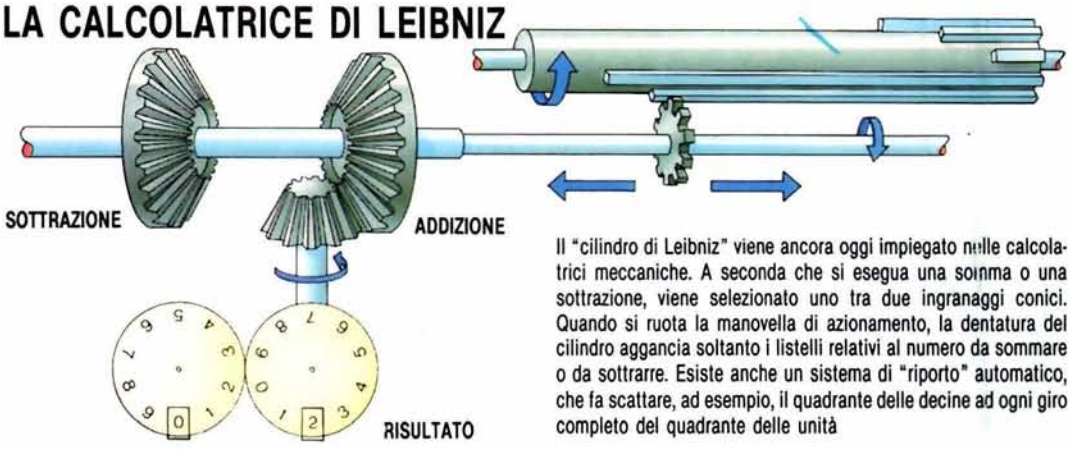
Uomo dagli interessi vastissimi, durante i suoi spostamenti in Europa, conobbe quasi tutti i grandi pensatori del suo tempo ed intrattenne una fittissima corrispondenza epistolare con un vasto numero di persone. Dette il suo primo notevole contributo alla filosofia nel 1672, formulando il "principio di ragion sufficiente", in base al quale esiste una ragione "sufficiente" per collegare ogni evento che si verifica insieme con altri.

Nel campo matematico, si dedicò al perfezionamento della "Pascalina", la calcolatrice inventata da Blaise Pascal (vedere a pag. 86), affinché con essa si potessero eseguire le moltiplicazioni e le divisioni e a tal fine inventò un particolare meccanismo, detto appunto il "cilindro di Leibniz" (vedere illustrazione), che costituì un'enorme innovazione per i tempi di allora. La matematica, fino ad allora talmente complessa, che veniva insegnata solamente negli istituti ad alto livello, divenne improvvisamente accessibile a tutti, mediante un semplice strumento meccanico. In seguito, Leibniz si dedicò allo studio della matematica basata sul sistema binario, anziché decimale.

Il suo scopo era quello di formulare un linguaggio matematico universale, capace di risolvere qualsiasi problema umano. Questo linguaggio avrebbe dovuto impiegare dei simboli astratti, rappresentanti gli "atomi" della conoscenza, e delle regole per elaborare detti simboli. Il suo tentativo fallì, ma ai primi del XX secolo Bertrand Russell riprese in parte i concetti di Leibniz nel tentativo di spiegare la matematica secondo un linguaggio logico.

In questi ultimi anni, il pensiero di Leibniz è stato "rispolverato" dagli scienziati impegnati nella creazione della quinta generazione di computer, ossia gli elaboratori che dovrebbero essere in grado di risolvere qualsiasi problema umano.

LA CALCOLATRICE DI LEIBNIZ



KEVIN JONES

Il "cilindro di Leibniz" viene ancora oggi impiegato nelle calcolatrici meccaniche. A seconda che si esegua una somma o una sottrazione, viene selezionato uno tra due ingranaggi conici. Quando si ruota la manovella di azionamento, la dentatura del cilindro aggancia soltanto i listelli relativi al numero da sommare o da sottrarre. Esiste anche un sistema di "riporto" automatico, che fa scattare, ad esempio, il quadrante delle decine ad ogni giro completo del quadrante delle unità

Il Lisa e le "icone"

Un modo rivoluzionario di concepire il software applicativo

Il Lisa, l'ultimo prodotto della Apple Computer, è un elaboratore destinato essenzialmente ad uso commerciale ed ha un prezzo piuttosto elevato. Questo computer merita particolare attenzione, perché molte delle sue caratteristiche, decisamente innovative, saranno in futuro utilizzate anche negli home computer. La stessa Apple ha da poco realizzato una versione più economica del Lisa, il MacIntosh, e già diverse industrie concorrenti si accingono ad imitare il nuovo apparecchio.

Il Lisa non possiede un hardware particolarmente nuovo: è, piuttosto, il suo software che presenta radicali innovazioni. Il progetto dei microcomputer, infatti, è ormai talmente standardizzato che gli investimenti maggiori (sia di tempo che di denaro) vengono fatti per migliorare sempre più il software applicativo.

Prima di parlare del software, vediamo le caratteristiche principali del Lisa. Innanzitutto, questo computer possiede 1 megabyte di RAM come standard (una memoria mille volte superiore alla memoria standard di uno ZX81). Ovviamente, ciò costringe la CPU a dedicare una discreta quantità del proprio tempo alla gestione della memoria, elaborando i dati in essa presenti e mantenendo traccia di dove ogni cosa si trova. Il processore, un Motorola 68000, appartiene alla categoria delle CPU a 16 bit (capaci, cioè, di gestire 16 bit di dati alla volta, mentre la maggior parte delle attuali CPU ne gestiscono soltanto 8). Il 68000 è un processore molto veloce ed il suo linguaggio macchina dispone di istruzioni molto sofisticate. In quanto a memoria di massa, il Lisa prevede l'impiego di due unità a dischetti e di una a disco fisso. Quest'ultima si rende necessaria sia per la velocità di funzionamento, tipica delle unità a disco fisso, sia per disporre di sufficiente capacità (5 megabyte) per dati e programmi. Il Lisa, infatti, usa un gran numero di programmi, che devono essere frequentemente trasferiti tra la RAM ed il disco.

Un'altra caratteristica sorprendente dell'hardware del Lisa è lo schermo monocromatico incorporato, che ha una risoluzione di 720 × 364 pixel. Questo permette una varietà di differenti "corpi" dei caratteri di visualizzazione, oltre ad una grafica eccezionale, come vedremo poco oltre.

Collegandosi ad una stampante appropriata (come per esempio un'unità a matrice d'aghi ad alta velocità e di buona qualità), è possibile riprodurre su carta qualsiasi immagine visualizzata sullo schermo. Se la stampante non dovesse essere compatibile con l'alto livello di risoluzione dello schermo, il Lisa produrrà comunque un'immagine stampata della migliore qualità possibile.



IAN McKINNELL

Un computer amichevole

Il Lisa della Apple è stato progettato per essere usato da operatori commerciali privi di qualsiasi esperienza nel campo dei computer.

L'uso del mouse permette, rispetto ad altri sistemi, un uso assai ridotto della tastiera

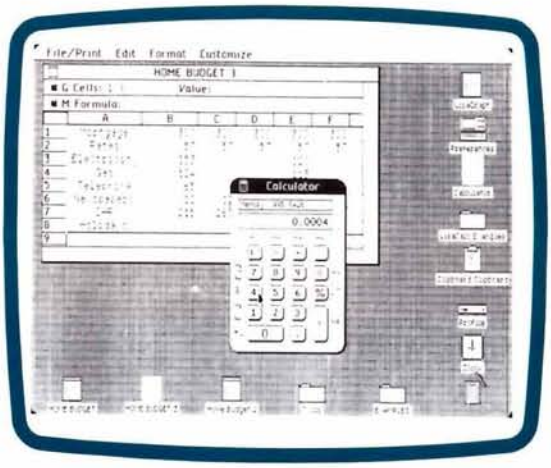
La tastiera del Lisa è contenuta in un involucro separato da quello del computer ed è ben concepita, nonostante ne sia previsto un uso molto sporadico: infatti, la principale unità per l'immissione dei dati è un "mouse" o "topo".

Se in altri computer il topo costituisce (assieme ai joystick e simili) un accessorio adoperato soprattutto per i giochi, nel Lisa permette di operare sullo schermo senza usare la tastiera. Il mouse è, in pratica, una piccola scatola, dalle dimensioni di un pacchetto di sigarette, che, fatta scorrere su di una superficie piana, trasmette direzione e lunghezza dello spostamento al computer, provocando così lo spostamento del cursore sullo schermo. In questo modo è possibile "puntare" sul video il dato o il comando desiderato e quindi, premendo il tasto SELECT (posto sull'involucro del mouse), si può selezionare quel particolare dato o comando. La tastiera viene quindi utilizzata solo per l'inserimento di nuovi dati in forma di testo o di numeri.

Quando il Lisa viene acceso, sullo schermo viene riprodotta la superficie di una scrivania, sulla quale sono disposti diversi oggetti: ogni oggetto rappresenta, sinteticamente, ma

Il simbolo giusto

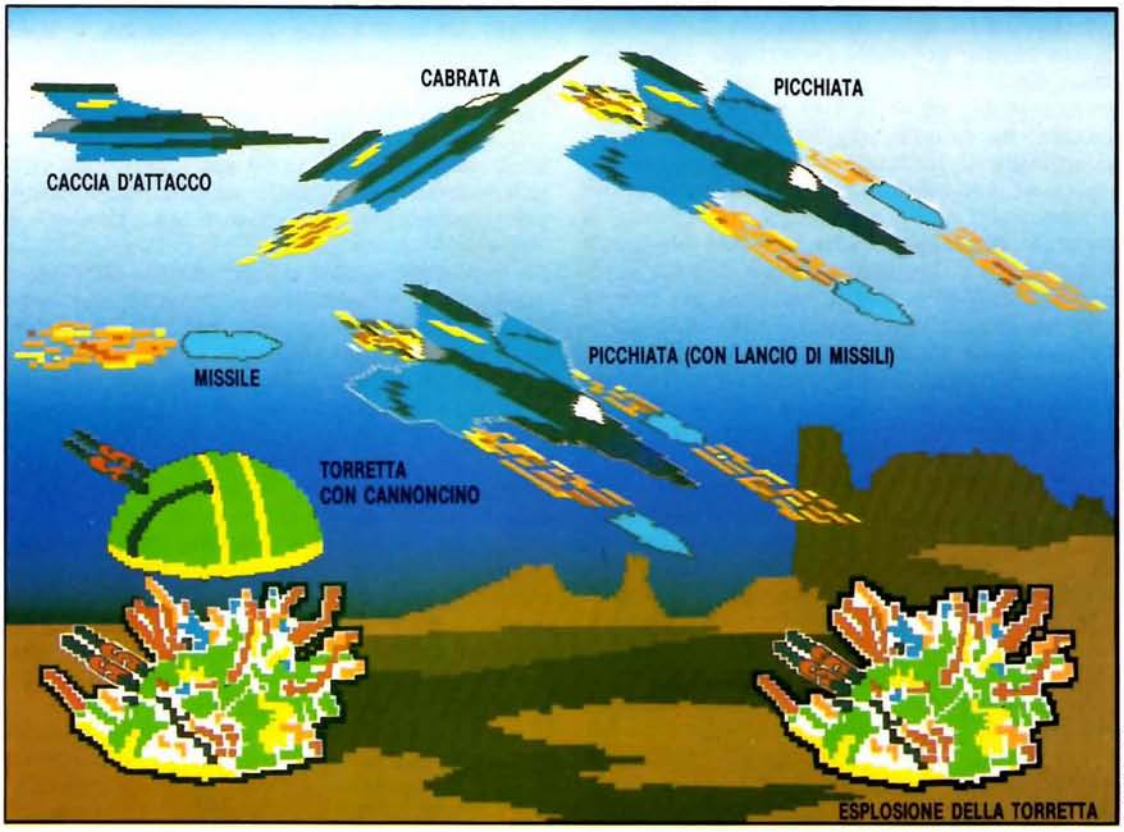
Ogni funzione, sul Lisa, è rappresentata sullo schermo da un simbolo chiamato icona. Per attivare una funzione, basta muovere il mouse per spostare il cursore sull'icona voluta e quindi premere il tasto SELECT del mouse. Questa semplice operazione avvia l'esecuzione della particolare applicazione, che viene visualizzata in dettaglio sullo schermo



efficacemente, una particolare funzione che possiamo selezionare impiegando il topo.
Questa facilità di selezione è una delle principali ragioni per cui i principianti trovano più facile lavorare con il Lisa che non con hardware e software convenzionali. Gli oggetti disposti sulla scrivania sono stati chiamati, dalla Apple, icone e ciascuna icona è accompagnata da un "sottotitolo", che ne chiarisce il significato.
Prendiamo, ad esempio, l'orologio. Se, usando il mouse, spostiamo il cursore sul piccolo orologio e premiamo il tasto SELECT, sullo schermo apparirà un'immagine ingrandita dell'orologio e la data. Ciò consente di immettere o modificare sia l'ora che la data del giorno in cui stiamo lavorando. Una volta terminate queste operazioni, possiamo "riporre" l'orologio nella sua posizione e dimensioni originali. Similmente, selezionando l'icona della calcolatrice, la sua immagine occuperà gran parte dello schermo, permettendo di eseguire semplici operazioni aritmetiche, sempre mediante l'impiego del topo.
Volendo, si può modificare la disposizione degli oggetti sulla scrivania e per far ciò è sufficiente manovrare opportunamente il topo. Una delle funzioni più divertenti, che bene evidenzia quanto il Lisa sia modellato sul nostro abituale modo di lavorare, è simboleggiato dall'icona del cestino dei rifiuti. Se non si ha più bisogno di una parte del lavoro svolto, la si può semplicemente "cestinare", usando il mouse. È comunque possibile riesaminare il contenuto del cestino dei rifiuti e recuperarlo (a meno che il Lisa non sia stato spento nel frattempo!), e questa procedura garantisce un certo grado di sicurezza, eliminando il rischio di distruggere accidentalmente i risultati di una elaborazione.
La maggior parte del lavoro viene compiuto, sul Lisa, utilizzando uno dei sei programmi applicativi di cui è dotato: il LisaWrite è un "word processor"; il LisaCalc è una lavagna elettronica; il LisaGraph è un sistema per la gestione di grafici; il LisaList è un sistema che permette la gestione di un "database" (archivio organizzato); il LisaProject è un sistema di sussidio alla preparazione di piani di lavoro; il LisaDraw, infine, è un sofisticato programma per la creazione di ogni genere di immagini grafiche. Le icone relative a queste applicazioni sono semplicemente dei blocchi di carta. Per effettuare un calcolo sulla lavagna elettronica, ad esempio, basta spostare il cursore sul blocco di carta a fogli quadrettati chiamato LisaCalc. Premendo il tasto SELECT, un foglio si staccherà dal blocco

La "programmazione orientata"

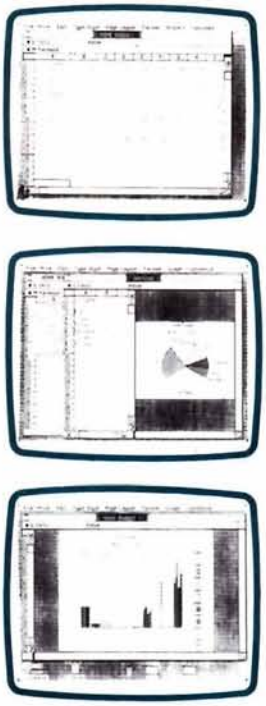
La creazione di un gioco per sala giochi del tipo "Defender", usando un tipo di programmazione convenzionale, richiede la progettazione di un certo numero di tracciati video e, quindi, la stesura di un programma che controlli, passo per passo, l'andamento del gioco. Con la "programmazione orientata" (attualmente difficile da implementare sugli home computer per la mancanza di linguaggi adeguati), ci si concentra su di ogni singolo elemento del gioco. Partendo dal "caccia d'attacco", la prima definizione potrebbe stabilire che il suo movimento sullo schermo sarà sempre da destra a sinistra, che con uno spostamento in avanti o indietro della "cloche" il caccia si muoverà in picchiata o in cabrata ed, infine, che la pressione del pulsante di FUOCO provocherà il lancio di un missile. La seconda definizione, invece, potrebbe riguardare le caratteristiche del missile, indicandone la forma, la traiettoria e la sua scomparsa al contatto con l'obiettivo. La torretta del cannoncino statico è una semplice sagoma che si trasforma nell'immagine di un'esplosione al contatto del missile. Esprimete queste 3 definizioni con un linguaggio adeguato e mettetele assieme: il gioco è fatto!

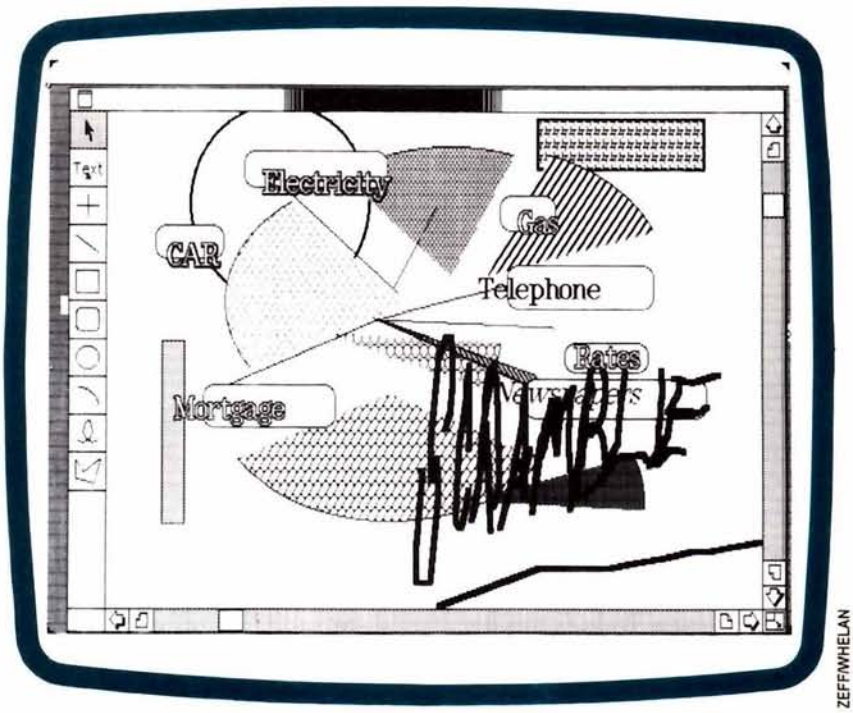


Trasferimenti successivi

Caratteristica distintiva del Lisa è la possibilità di trasferire informazioni da un'applicazione all'altra usando la funzione COPY. Questa permette di archiviare temporaneamente, nella "bacheca", dati o risultati di elaborazioni, che intendiamo, in seguito, utilizzare in un altro programma. Possiamo, per esempio, eseguire dei calcoli con la lavagna elettronica (LisaCalc) e trasferire i risultati ottenuti al programma LisaGraph: otterremo così automaticamente grafici "a torta" o "a barre". A questo punto, possiamo effettuare un nuovo trasferimento e, con il programma LisaDraw, aggiungere al grafico nuove etichette, frecce, diagrammi, ecc.

Il risultato finale può essere facilmente passato alla stampante, ottenendone una copia su carta





ZEFF/WHELAN

e si disporrà in un'altra parte della scrivania; ad esso si potrà attribuire una nuova etichetta, ad esempio "proiezioni di vendita".

In effetti l'utente può decidere di avere nello stesso tempo diverse applicazioni della lavagna elettronica sulla scrivania selezionando più volte la stessa icona. Su di un normale computer si dovrebbe procedere caricando il programma della lavagna elettronica e quindi specificando il file di dati su cui si desidera lavorare. Sul Lisa, al contrario, programma e dati non hanno più questa netta distinzione: questo è un magnifico esempio di "programmazione orientata", della quale abbiamo parlato presentando il *Flipper in scatola di montaggio* (vedere a pag. 241).

Un'altra importante caratteristica dell'"ambiente operativo" (così viene chiamato il sistema) del Lisa, è l'abilità di "aprire finestre" sul video. Quando si seleziona un'applicazione, sulla scrivania appare un ampio foglio di carta; le sue dimensioni possono essere definite usando nuovamente il mouse. Se sulla scrivania vi sono più applicazioni aperte nello stesso momento, i fogli si sovrapporranno l'un l'altro e quello su cui si sta lavorando si troverà sopra tutti gli altri. Può accadere che quest'ultimo, ad esempio il "word processor", venga usato su di un campo visivo più ampio di quello inizialmente previsto: in questo caso, la superficie del foglio funge da finestra sull'intero documento e può essere comodamente spostata per visualizzare sullo schermo ora una parte ora l'altra di esso. Il concetto di "finestre", sullo schermo del computer, è stato ampiamente spiegato quando abbiamo esaminato per la prima volta le lavagne elettroniche.

Inoltre, è anche possibile spostare informazioni e dati da un'applicazione all'altra, ricorrendo nuovamente alle icone: questo è un altro grande pregio del Lisa. Poniamo di voler analizzare i dati relativi alle vendite di un particolare mese. Dopo aver impostato e calcolato quanto ci occorre con il programma LisaCalc, possiamo adoperare la funzione COPY, selezionabile dal menu delle funzioni speciali (situato nella parte superiore dello schermo), e potremo temporaneamente trasferire i risultati ottenuti con la lavagna elettronica nell'icona che rappresenta una piccola bacheca. Quindi, dopo aver selezionato il LisaGraph, i risultati potranno essere estratti dalla "bacheca" ed inseriti nella sezione INPUT DATA della applicazione Lisagraph, semplicemente selezionando da menu l'opzione PASTE (letteralmente, INCOLLA). A questo punto, col LisaGraph possiamo produrre un chiaro grafico "a torta" o un istogramma, opportunamente colorati e corredati di titoli, sottotitoli e spiegazioni.

Usando nuovamente le funzioni COPY e PASTE si potranno trasferire queste immagini grafiche su un foglio del LisaDraw: ciò permetterà di arricchire il grafico "a torta" con frecce o diagrammi oppure di variare i caratteri tipografici delle etichette. Il risultato finale potrà essere stampato o riprodotto con un plotter, per esempio, su carta lucida per lavagne luminose, oppure usato in un processo di fotoincisione.

Come si è detto, sia i principi alla base della "programmazione orientata", sia il concetto di "ambiente operativo", di cui il Lisa si avvale in larga misura, troveranno presto concreta applicazione anche su computer più economici, via via che questi diverranno più sofisticati in termini di velocità del processore e di dimensioni della memoria RAM. Questo nuovo modo di concepire l'interazione tra l'uomo ed il computer, assieme alla possibilità di usare il normale linguaggio parlato, per dialogare con le macchine, apre nuovi orizzonti per la società umana.

Una finestra sul mondo

Il televisore di casa può diventare un terminale dal quale accedere alle banche dati

I sistemi "viewdata", come il servizio Videotel della Sip o il Prestel della British Telecom's, consentono di accedere ad un'ampia varietà di banche dati, sfruttando due apparecchi domestici comunissimi: un televisore (opportunamente modificato) ed un telefono. Un sistema viewdata può essere di tipo pubblico, destinato ed utile per tutti, come il Videotel; oppure può essere accessibile a tutti, ma di contenuto piuttosto specializzato (come il Fintel britannico, che è una banca dati di carattere finanziario costantemente aggiornata); oppure può essere privato.

I sistemi viewdata non devono tuttavia essere confusi con il sistema teletext, oggi disponibile sui canali televisivi, che impiega un'impaginazione ed una gamma di caratteri molto simili. I sistemi teletext, come ad esempio il Televideo della RAI, il Ceefax della BBC o l'Oracle della ITV, vengono diffusi mediante speciali segnali "sovrapposti" a quelli dei normali programmi televisivi: un apparecchio di decodifica li separa e ricostruisce una particolare pagina d'informazioni. Il testo e le informazioni grafiche inviate dal Televideo possono sostituire la normale immagine televisiva o essere sovrapposti ad essa così da apparire contemporaneamente (in Gran Bretagna, ad esempio, questo modo di funzionamento viene impiegato per la realizzazione di sottotitoli per le persone prive d'udito).

I sistemi viewdata, invece, usano il televisore come un monitor per visualizzare i dati ricevuti mediante la rete telefonica. In genere, gli utenti non possono modificare le banche dati, ma possono usare il loro telecomando per rispondere alle domande del sistema. Tutti i sistemi viewdata sono impostati sull'uso di menu: all'utente viene mostrato un elenco di opzioni, sempre più particolareggiato, sui vari argomenti delle "pagine". La selezione viene fatta digitando, sull'apposito telecomando, un numero. Sullo schermo appare quindi un "sottomenu" e questa procedura si ripete lungo tutta la gerarchia finché si giunge alla pagina di informazione richiesta.

Poiché ogni pagina è identificata da un particolare numero, è anche possibile selezionare immediatamente quella che interessa, saltando tutti i sottomenu.

Come esempio, seguiamo un utente immaginario, mentre prenota "via Prestel" le sue vacanze estive. Il primo messaggio che egli vede quando accende il sistema, lo invita a premere il simbolo "#" sul telecomando per visualizzare il menu principale. Da questo momento egli seguirà, di pagina in pagina, le indicazioni fornite dai sottomenu. La prima



per concessione della PRESTEL

"sosta" sarà fatta sulla pagina delle Informazioni Generali. La quarta voce di questa pagina ha per titolo: "Vacanze, Trasporti, Viaggi", per cui egli seleziona il numero 4. In risposta ottiene la possibilità di scegliere tra: Viaggi Ferroviari, Aerei, in Autobus, Altri Trasporti, Vacanze e Turismo, Informazioni Automobilistiche. Ci sono anche quattro altre opzioni, ciascuna delle quali porta ad un quadro informativo. Dopo aver battuto il tasto corrispondente a Vacanze e Turismo, l'utente del nostro esempio deve decidere se prenotare viaggio e sistemazione alberghiera separatamente o se scegliere una prenotazione "tutto compreso": la sua scelta lo porterà ad un'altra pagina di opzioni, ed il procedi-



per concessione della TELEVIDEO

Informazioni per tutti

Il servizio Prestel della British Telecom (che offre qualcosa come 250.000 pagine d'informazioni), è accessibile da qualsiasi punto del paese e, a determinate condizioni, anche da un telefono pubblico. Tutto ciò che è necessario per aver accesso alla banca dati è una linea telefonica e un ricevitore televisivo

Il servizio Televideo della RAI

Da non confondersi col Videotel (vedere pagina a fianco), che utilizza la linea telefonica, il servizio d'informazione Televideo invia dati via etere, come accade con le normali immagini televisive. Gli abbonati al servizio possono consultare, nella comodità del proprio salotto, le "pagine" d'informazioni (classificate per argomenti quali: viabilità, orari ferroviari, ecc.) semplicemente selezionandone il codice su di un apposito telecomando. Un particolare apparecchio, da collegare al televisore, provvede a decodificare le informazioni ed a ricostruire le "pagine" sullo schermo

mento continua. Alternativamente, per la selezione, si può consultare una guida (specie di piccolo elenco suddiviso per argomenti), mediante la quale l'utente può accedere direttamente alla pagina che lo interessa.
Abbiamo scelto l'esempio di un viaggio, perché questo tipo di richiesta è, nell'esperienza britannica (ormai pluriennale), la più usata dell'intero sistema.
Ideato nel 1971, nei laboratori di ricerca della British Telecom, il primo sistema viewdata è stato attivato nel 1976 per un esperimento in laboratorio e quindi reso accessibile al pubblico come Prestel alla fine del 1979. La British Telecom aveva inizialmente chiamato il suo sistema Viewdata ma fu costretta a cambiarne il nome in Prestel, in quanto l'Ufficio Brevetti considerò Viewdata un termine troppo generico. Comunque, il sistema britannico, fin dalla sua nascita, è stato riconosciuto come uno standard anche a livello internazionale.
L'accesso ai sistemi viewdata avviene impiegando la rete telefonica ed all'utente viene addebitato, sulla bolletta, il costo della chiamata telefonica (in base alla durata) oltre a quello dell'abbonamento. C'è poi la possibilità di una terza spesa: quella relativa alla particolare pagina richiesta, ma questa è a discrezione del fornitore dell'informazione. Per ridurre le spese telefoniche degli abbonati, la compagnia telefonica britannica ha installato un certo numero di "concentratori locali" (ossia linee telefoniche dedicate al sistema Prestel), che permettono all'utente di effettuare chiamate Prestel interurbane, ma al costo di una chiamata urbana.
L'hardware necessario per usare i servizi viewdata si divide in tre tipi principali. Molto sofisticato (ma anche costoso) è lo speciale terminale, completo, per Viewdata: di esso si avvalgono la maggior parte degli utenti che svolgono attività commerciali. La seconda alternativa è quella di acquistare uno speciale decodificatore da applicare al televisore. Questi apparecchi hanno diverse varianti: si va dai semplici telecomandi numerici, con chiamata telefonica manuale, a quelli costituiti da vere e proprie tastiere (simili a quelle dei computer) con chiamata automatica. Comunque, il "cuore" di tutti questi strumenti è un microcomputer "dedicato", che decodifica i segnali in arrivo e in partenza.

IL MICRONET 800

Il servizio Micronet 800 permette agli abbonati di acquistare software direttamente dal sistema computerizzato centrale del Prestel. Per fruire di questo servizio, è necessario ricevere i segnali viewdata con un microcomputer (non basta un televisore adattato) e quindi archiviare il programma su cassetta o su disco. Dopodiché esso potrà essere successivamente caricato e fatto girare come qualsiasi altro programma normale. Il Micronet offre circa un centinaio di programmi gratis e molti altri sono acquistabili a prezzi decisamente concorrenziali. Il sistema Micronet si può installare sulla maggior parte dei normali microcomputer, incluso il Micro della BBC, alcuni dei modelli Commodore, lo Spectrum e l'Apple II. Gli abbonati Micronet hanno anche accesso al resto delle banche dati del Prestel



per concessione della BRITISH TELECOM



per concessione della VIDEOTEL

Il servizio Videotel della SIP

Quella che vediamo qui accanto è una delle tante "pagine" Videotel consultabili comodamente da casa propria. Questo servizio, di recentissima diffusione, sfrutta tre comunissimi elettrodomestici: un telefono, un televisore (opportunamente adattato) ed un telecomando. Componendo un codice sulla tastiera del telecomando, sullo schermo dell'apparecchio televisivo possiamo far apparire, una alla volta, intere "pagine" d'informazioni, costantemente aggiornate, provenienti dal vastissimo archivio del Centro Videotel

La terza alternativa, molto diffusa tra gli utenti di home computer, è quella di acquistare il software viewdata per microcomputer. Quest'ultima forma è divenuta particolarmente interessante da quando è stato introdotto, in Gran Bretagna, un servizio conosciuto come Micronet 800, che offre agli abbonati la possibilità di scambiare programmi per computer direttamente sulla linea telefonica.
Il Prestel offre anche un altro interessante servizio, permettendo agli abbonati di scambiarsi messaggi mediante un sistema noto col nome di Mailbox (letteralmente, "cassetta delle lettere"). L'abbonato viene automaticamente avvisato della presenza di un messaggio quando accende il suo terminale Prestel, o, se il terminale è già in uso, quando termina una chiamata. Egli può controllare in qualsiasi momento, anche durante una chiamata, se ci sono messaggi in arrivo. Inoltre non è necessario, come si potrebbe pensare, possedere una tastiera alfanumerica per poter inviare messaggi. Ci sono infatti vari tipi di messaggi standard che possono essere compilati usando solo caratteri numerici.
Nel 1983 c'erano circa 35.000 abbonati al Prestel in Gran Bretagna, con più di 250.000 pagine d'informazioni disponibili e un numero imprecisato di compagnie e organizzazioni che usavano le informazioni viewdata per le loro necessità interne. Gli equivalenti sistemi italiani (il Videotel ed il Televideo) saranno presto in grado di offrire ai propri utenti servizi molto simili.

I giochi... postali

Per comunicare con altri utenti di computer si può usare anche il servizio postale

Moltissimi acquirenti di home computer asseriscono di voler imparare a programmare, ma non c'è dubbio che, oggi, i giochi siano le applicazioni più in voga per queste macchine. Niente di male in questo, perché, come abbiamo spesso ripetuto, i giochi per computer possono essere allo stesso tempo educativi e divertenti. Perché allora accontentarsi solo dei giochi tipo *Space Invaders*? Il computer ha permesso lo sviluppo di nuovi modelli concettuali di gioco come gli *Adventure* (vedere a pag. 161) e le simulazioni (vedere a pag. 81), per non parlare, poi, delle versioni computerizzate di popolari giochi da tavolino.

Esiste tuttavia una famiglia di giochi per computer di cui non abbiamo ancora parlato e che è ancora sconosciuta al grande pubblico. In netto contrasto con i giochi per "sale giochi", che richiedono tempi di gioco rapidi e prontezza di riflessi, questi giochi si sviluppano a velocità ridotta e una mossa può prendere anche diverse settimane! La maggior parte dei giochi per computer sono concepiti per un solo giocatore: questi, invece, possono coinvolgere decine di giocatori contemporaneamente.

Stiamo parlando dei giochi postali: i partecipanti a tali giochi possono abitare in luoghi diversi o, in caso di gare internazionali, in parti diverse del mondo. Ogni giocatore comunica le proprie mosse inviando lettere a intervalli di tempo prestabiliti. Le mosse vengono spedite ad un coordinatore del gioco, che le inserisce in un microcomputer (la maggior parte di questi giochi non obbliga infatti al possesso di un home computer). Il coordinatore, a sua volta, spedisce a ciascun giocatore uno stampato (prodotto dal computer), che lo informa della sua posizione e di altre importanti notizie, come la posizione degli altri giocatori con cui è entrato in contatto.

Questi giochi durano abitualmente molti mesi, se non all'infinito. Tuttavia, i giocatori possono iniziare o abbandonare il gioco in qualsiasi momento. Di solito, si corrisponde una tassa di iscrizione, che dà diritto a ricevere il regolamento del gioco ed il materiale iniziale occorrente ed una tassa di gioco (che può aggirarsi sulle 2000 lire) per ogni mossa. Una mossa alla settimana è considerato un ritmo di gioco "veloce", in quanto nelle gare internazionali possono passare anche sei settimane tra due turni.

Un gioco postale può coinvolgere molte decine di giocatori, ma se il numero di giocatori sale ancora, il coordinatore dà semplicemente il via ad un secondo gioco parallelo al primo usando sul suo computer gli stessi programmi, ma differenti dischi dati. Questi giochi esistevano molto prima della comparsa dei computer: l'esempio più classico è il gioco degli scacchi. Un altro è il *Diplomacy*: in esso i giocatori rappresentano 7 Nazioni Europee e mirano a conquistare l'intero continente stringendo e rompendo alleanze con ciascuno degli altri.

L'introduzione del computer per l'esecuzione di tutte le incombenze amministrative e di calcolo ha ampliato le possibilità di questi giochi rendendoli più sofisticati ed ingegnosi. Alcuni giochi, oggi, si svolgono in immense galassie, nelle quali i giocatori pilotano le loro flotte spaziali, altri, invece, hanno per scenario mitiche terre e imperi in guerra.

La singolarità di questi giochi è che i giocatori interagiscono tra loro e non sono solo degli "esploratori", come nei giochi di tipo *Adventure*. Una riprova dell'ottimo livello di questi giochi è che ad essi partecipano persone di ogni età, cosa che assai di rado succede per gli altri tipi di giochi per computer.

Guerre galattiche

Il "Signore delle stelle" è stato il primo gioco postale controllato da un computer in Gran Bretagna. Esso viene coordinato dal signor Mike Singleton usando un Pet Commodore 3032, un'unità a disco fisso da 7,5 Megabyte e una stampante a matrice d'aghi "Integrex colour". L'obiettivo di ciascun giocatore è quello di trovare la Stella del Trono e di divenirne imperatore. Dopo ogni mossa ognuno riceve una mappa delle zone intorno alle sue armate e una lista di quanti controllano le stelle vicine. La grande capacità di memoria dell'hard disk si è resa necessaria dalla quantità di programmi occorrenti al gioco e dalla massa di informazioni da immagazzinare provenienti da più di 700 giocatori. In Gran Bretagna esiste anche una pubblicazione periodica sull'argomento, la "Flagship"

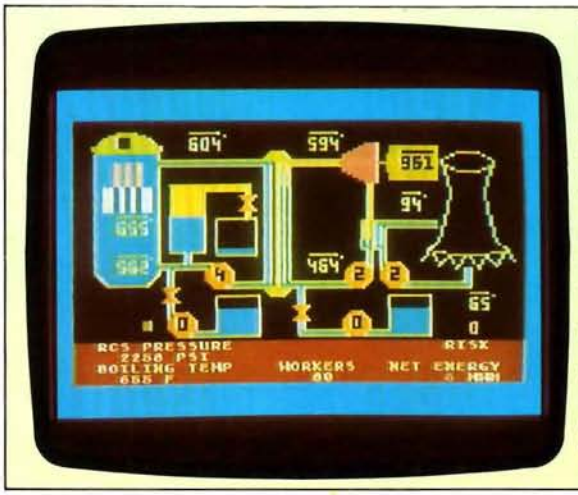


per concessione della STARLORD

Il comportamento dei modelli

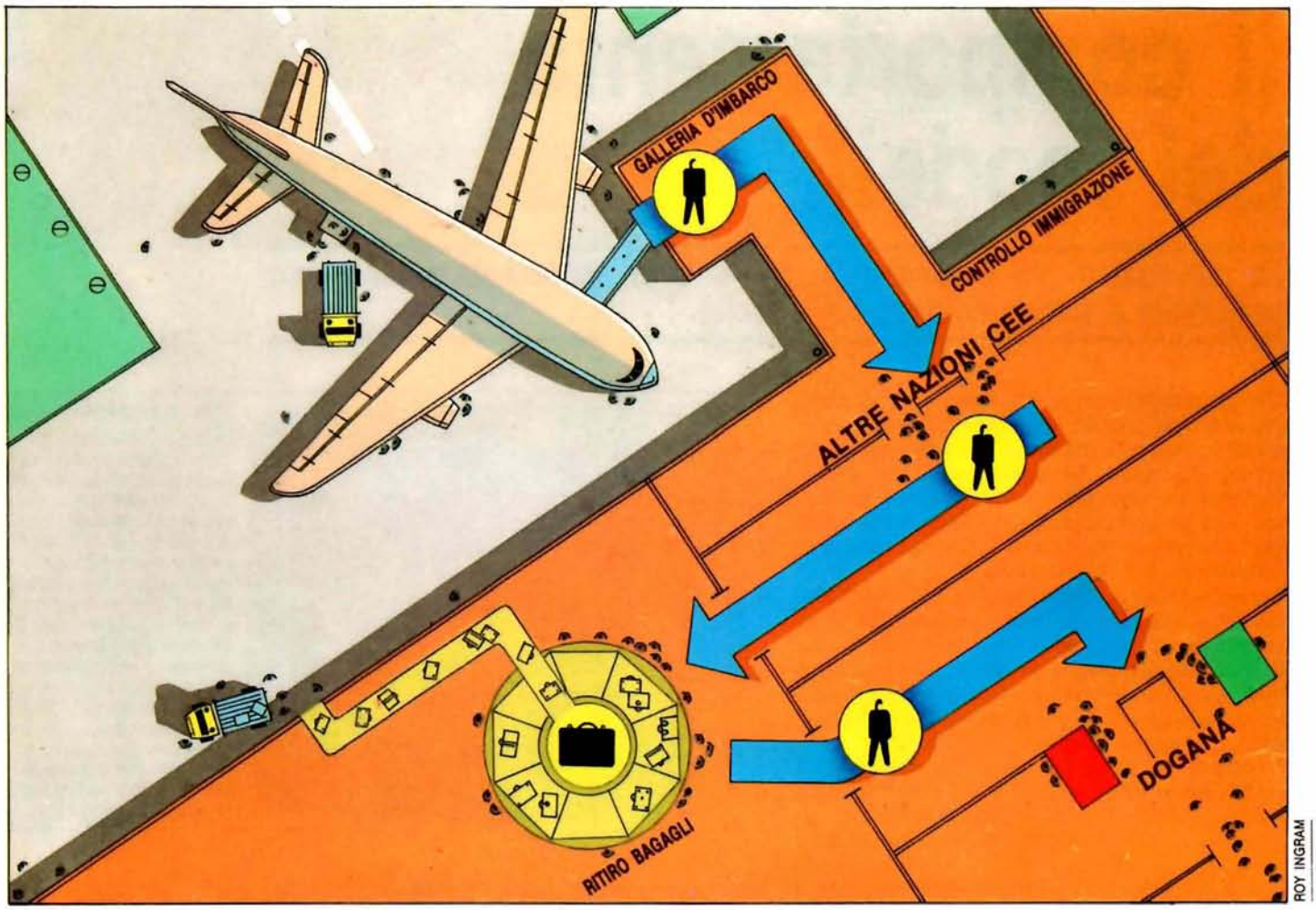
Le simulazioni su computer permettono di sperimentare situazioni pericolose o troppo costose nella realtà

Un'interessantissima applicazione dei computer è quella delle "simulazioni", grazie alle quali è possibile simulare su computer particolari fenomeni da analizzare. Il modello fornisce un'immagine semplificata del problema, pur conservandone le caratteristiche significative ed eliminando i dettagli che non influiscono sul risultato finale.
Consideriamo ad esempio due bicchieri: uno contiene vino rosso, l'altro vino bianco. Versiamo un cucchiaio di vino rosso in quello bianco, mescoliamo a fondo e riportiamo un cucchiaio della miscela ottenuta nel bicchiere di vino rosso. In quale dei due bicchieri si avrà il maggior grado di impurità?
Le strade per risolvere questo problema sono molte, ma la più semplice è di ricorrere ad un modello. Ad esempio, è possibile creare un modello in cui il volume di vino in ciascun bicchiere sia esattamente uguale a quello di un cucchiaio. In questo caso, entrambi i bicchieri alla fine conterranno lo stesso grado di impurità (cioè un'uguale miscela di vino bianco e rosso). L'ampliamento del modello a maggiori quantità di vino dimostrerà che la stessa cosa è vera anche negli altri casi.
I modelli si possono suddividere in tre categorie principali. La prima comprende i modelli illustrati (per esempio una foto o una mappa), che evidenziano le disposizioni e i rapporti spaziali fra gli elementi presenti in quell'immagine. Vi sono poi modelli in cui componenti interagiscono tra loro in modo molto simile agli elementi reali del problema che essi rappresentano, per esempio problemi risolti mediante computer analogici (vedere pag. 238). La terza famiglia è costituita da modelli, nei quali vengono impiegati simboli astratti e relazioni matematiche per descrivere una data situazione. È quest'ultimo tipo di modelli che i computer digitali utilizzano nelle simulazioni.
La decisione di risolvere un problema, mediante l'impiego di modelli computerizzati, dipende da più circostanze. La prima si verifica quando una sperimentazione reale sarebbe troppo pericolosa: per esempio, la determinazione di quale sia il livello di sicurezza della radioattività in un territorio vicino ad un reattore nucleare. La seconda situazione, di cui è ottimo esempio un modello dell'economia nazionale, si verifica quando sembra impossibile trovare una pura soluzione matematica a un gruppo di equazioni che determinano il problema. È allora meglio comporre le equazioni in forma di modello sul computer ed osservare gli effetti prodotti su di esse da processi ed eventi diversi.
Il terzo caso si ha quando il problema da analizzare implica costi talmente alti, che ogni modifica deve essere effettuata allo stadio di modello, prima cioè di prendere una decisione sulla versione finale. Nella progettazione di un nuovo aeroporto per Londra, ad esempio, è stato fatto un vasto lavoro di simulazione per problemi di carattere ingegneristico e progettuale.
L'ultima situazione è quella che implica un modello totalmente teorico, per il quale è impossibile realizzare esperimenti di carattere fisico. Gli astrofisici, ad esempio, indagano sui processi di formazione delle stelle e usano modelli per confrontare una teoria cosmologica (come quella del "Big Bang") con altre.
In ogni simulazione il primo lavoro consiste nella definizione del modello, che viene concepito analizzando il problema ed individuando gli elementi più importanti, tra loro correlati.
Sistemi e modelli si dividono in due famiglie: sistemi "deterministici" (o "chiusi") e "stocastici" (o "aperti"). Quando un nucleo familiare fa il bilancio delle proprie spese, il denaro può essere ripartito in molti modi, ma alla fine i conti devono tornare. Ciò rende questo un sistema deterministico. Ma se la famiglia, nello spendere il denaro, basasse ogni decisione sul lancio di una monetina piuttosto



Tutto sotto controllo
Le simulazioni vengono applicate con enorme successo nell'insegnamento: i partecipanti a corsi formativi professionali sono spesso addestrati su modelli di sistemi reali. Un ottimo esempio è la simulazione di una centrale nucleare, disponibile sull'ATARI: in essa l'utente deve controllare i diversi sistemi di raffreddamento per prevenire il surriscaldamento del reattore. La documentazione spiega ampiamente le diverse funzioni di controllo all'interno di una centrale di questo tipo ed è presente anche una funzione per la gestione automatica del modello

IAN McKINNELL SOFTWARE per concessione della PILOT SOFTWARE



Problemi grandi quanto ... un Jumbo!

Per il progetto del Terminale 4 dell'aeroporto di Heathrow (Londra) è stata impiegata la simulazione mediante computer. Il programma inizialmente doveva simulare il piano giornaliero degli aerei in arrivo con un numero variabile di passeggeri e bagagli, ma in seguito è servito per simulare anche tutte le operazioni di sbarco dei passeggeri. Grazie all'accurata simulazione è stato possibile valutare anche le dimensioni ottimali del progetto per far fronte al volume di traffico previsto

che sulla quantità di denaro disponibile, allora il sistema diverrebbe stocastico.

Con le simulazioni teoriche non è tuttavia possibile essere assolutamente certi che il modello scelto sia quello giusto. Gli uomini pensavano che la terra fosse al centro dell'universo, finché Copernico fornì un modello matematico molto più semplice, col sole al centro. Oggi gli astronomi, osservando le lontane galassie, hanno scoperto che esse si allontanano da noi a velocità crescente: ciò può far nuovamente pensare che la Terra si trovi al centro dell'universo. E l'errore si può dimostrare ricorrendo ad un diverso modello: se, infatti, spruzziamo delle gocce d'inchiostro (che rappresentano le stelle) su un palloncino di gomma e quindi lo gonfiamo, ogni "stella" si allontanerà da tutte le altre, ma tuttavia, nessuna si troverà al centro del palloncino.

L'uso di modelli offre comunque molti vantaggi: ci aiutano a formulare teorie più precise, velocizzano le analisi, permettono modifiche affidabili e, cosa assai importante, sono meno costosi delle situazioni reali, alle quali si riferiscono. La BL System (una consociata della British Leyland) commercializza un sistema di simulazione multi-uso originariamente sviluppato per la progettazione di linee di produzione e di magazzini automatizzati. Il sistema si chiama "See Why" (Scopri perché) e, al posto di tradizionali elenchi statistici, fornisce i risultati sotto forma di grafici molto accurati. Il sistema ha ottenuto un notevole successo commerciale ed è stato utilizzato dalla British Airport Authority per progettare un modello del quarto terminale dell'aeroporto di Heathrow.

Un problema tipico, da trattarsi con un sistema di simulazione, è quello della "formazione di code". In un aeroporto, ad esempio, se cambia improvvisamente il vento e solo una delle piste di atterraggio è utilizzabile, gli aerei in attesa di atterrare si dovranno mettere in coda. Poiché ognuno di essi ha una limitata riserva di carburante e per ogni atterraggio è necessario un determinato spazio di tempo, la sequenza di atterraggio varia di volta in volta e le regole di formazione della fila saranno programmate all'interno del sistema. In queste simulazioni vengono usati generatori di numeri casuali (vedere a pag. 209) per creare situazioni impreviste come, appunto, l'arrivo casuale di un aeroplano.

La simulazione è dunque un'importante area di applicazione per i computer digitali ed ha anche favorito lo sviluppo di nuovi linguaggi di programmazione creati espressamente per i piani di simulazione (ad esempio, il GASP, lo SIMSCRIPS ed il GPSS). Via via che il mondo diventa più complesso, l'uso di modelli per simulare problemi diverrà sempre più diffuso.

La generazione di caratteri

Spieghiamo come creare e conservare nuovi caratteri al computer

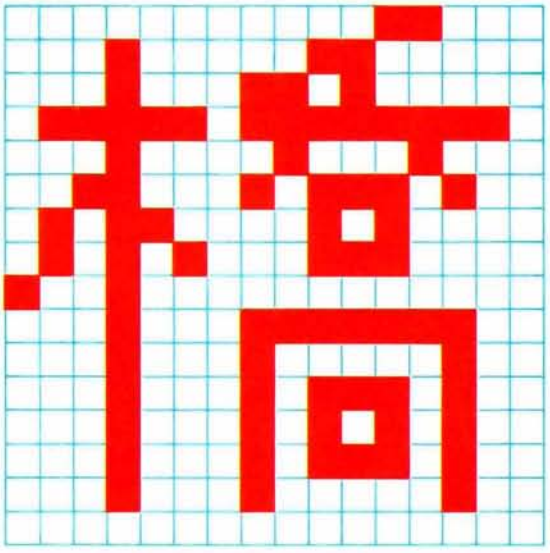
Quando viene visualizzata un'informazione, i codici relativi a ciascun carattere vengono depositati in un'area riservata di memoria, chiamata "RAM video". Se, per esempio, la lettera A viene visualizzata nell'angolo in alto a sinistra del video, il primo byte della RAM video conterrà il valore 65 (codice ASCII per A). Se una C viene stampata sotto la A, ed il computer ha un video di 40 colonne, allora il valore 67 si troverà nella quarantunesima locazione del video RAM, e così via. Come fa il computer a trasformare il valore 65 nello schema di punti che compongono il carattere A sullo schermo? La risposta risiede in un dispositivo, hardware o software, chiamato "generatore di caratteri". Un generatore di caratteri è semplicemente un insieme di modelli, archiviati nella memoria come combinazioni di bit. Negli home computer, il generatore di caratteri si trova nella ROM così da permettere, all'accensione dell'apparecchio, un'immediata visualizzazione dei caratteri. In questo caso, non è raro trovare in commercio ROM contenenti set di caratteri alfabetici stranieri o serie di caratteri speciali, ad esempio per ingegneri o matematici. Comunque, nella maggior parte dei computer, è possibile trasferire le definizioni dei caratteri nella RAM, permettendo al programmatore di progettare caratteri e simboli propri.

Tutti i caratteri sono costruiti su di una matrice di punti, che su molti computer ha una dimensione di 8 × 8 punti, sebbene una matrice di maggiori dimensioni permetta la creazione di un maggior numero di caratteri e una migliore definizione. La forma dei caratteri viene definita, sulla carta, inserendo dei quadratini in una griglia di dimensioni appropriate; i caratteri vengono poi codificati con un 1 (per ogni quadratino "pieno") e con uno 0 per ogni quadratino della griglia lasciato vuoto, fino ad un totale di 64 bit, ovvero 8 byte per ogni nuovo carattere definito. Il primo byte, nel generatore di caratteri, codifica la disposizione dei bit che compongono la riga più alta del primo carattere della serie. Se il computer può visualizzare solo caratteri ASCII con codici compresi tra 0 e 127, allora il generatore di caratteri impiegherà 128 × 8 byte (cioè 1 Kbyte di memoria).

La difficoltà per il computer consiste nel fatto che, quando la scansione del video genera sullo schermo la prima riga in alto, deve essere visualizzata la linea di punti più alta che è parte del carattere posizionato in alto a sinistra sul video, seguita dalla linea di punti più alta facente parte del carattere successivo, e così via attraverso lo schermo. Quando la scansione del video inizia il suo secondo passaggio, deve trovare e visualizzare la seconda linea di ogni carattere esattamente sotto la prima.

I circuiti di controllo del video eseguono questa operazione mediante due contatori indipendenti. Un contatore mantiene traccia di quale locazione della memoria video la scansione stia "spazzando", l'altro conta le linee della scansione del video partendo da 0 per la prima linea fino ad arrivare a 7 per l'ottava, ripartendo poi da 0 per la nona e così via. Quindi il computer legge dalla memoria video il codice ASCII (o quello impiegato), lo moltiplica per 8 e somma il risultato al valore corrente del contatore di linee. Questa operazione fornisce al computer l'indirizzo della particolare sequenza di 8 bit, nel generatore di caratteri, che corrisponde alla giusta linea del carattere che il computer sta "spazzando".

Prendiamo ad esempio la generazione del carattere A (codice ASCII = 65). Possiamo calcolare che la prima linea del carattere sarà immagazzinata nel byte n. 520 (65 × 8 + 0), la seconda linea nel byte n. 521 (65 × 8 + 1), la terza nel byte n. 522 (65 × 8 + 2) e così via. Ciò che il circuito di controllo del video deve fare adesso è convertire queste serie di 8 bit in una sequenza di tensioni che faranno illuminare o meno i vari punti sullo schermo.





Ideogrammi al computer

I caratteri giapponesi possono essere molto complessi ed una normale matrice 8 x 8 non riesce a rappresentarli in modo sufficientemente dettagliato, tale da renderli leggibili. L'ideogramma "ponte" qui mostrato, è appena leggibile su di una matrice 16 x 16. Una maggior chiarezza si ottiene usando una matrice 24 x 24

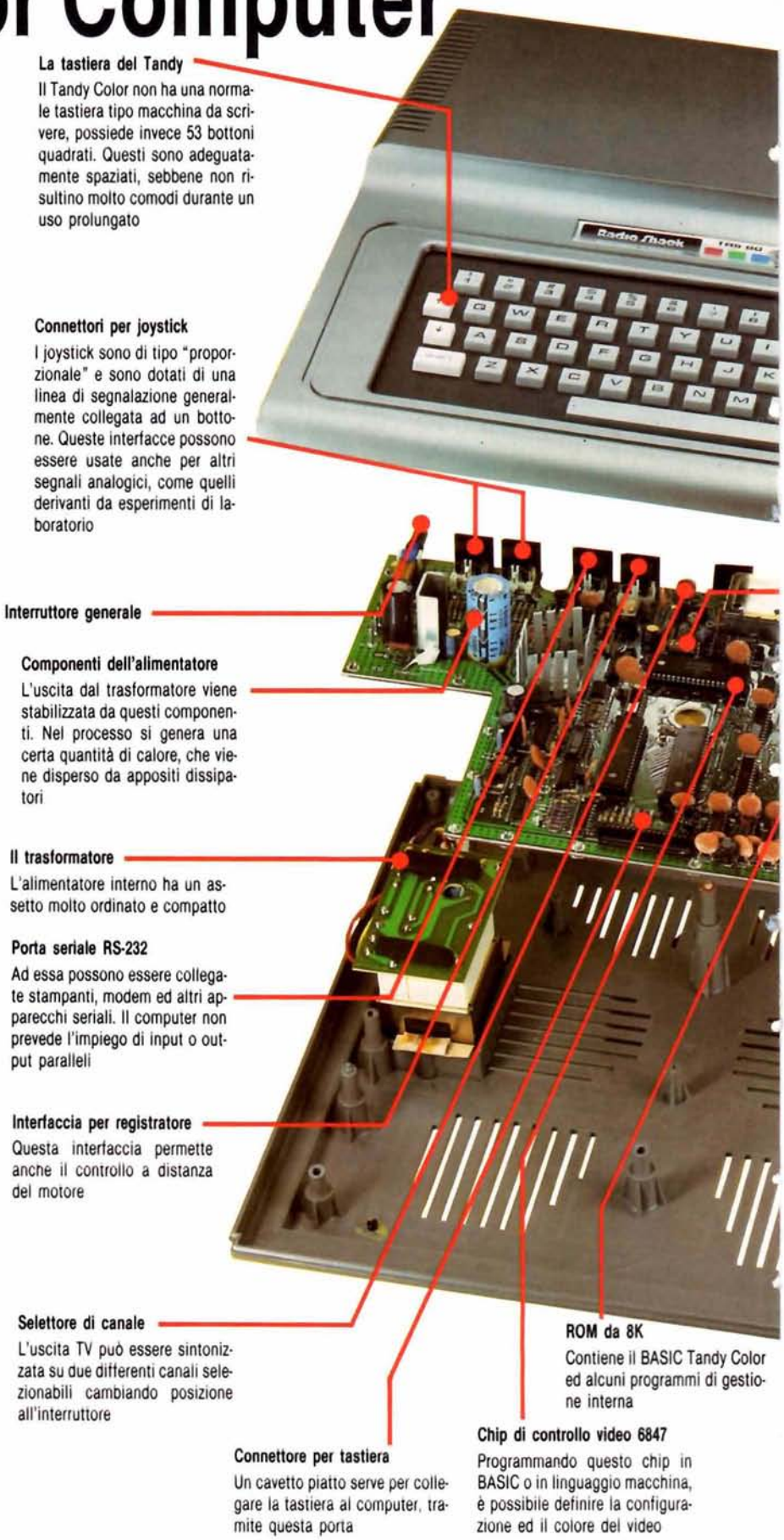
Il Tandy Color Computer

Anatomia di un microcomputer a diffusione europea: tecnologie a confronto

Per quanto molto diverso nell'aspetto, il computer Tandy Color presenta notevoli somiglianze nel modo di operare con il Dragon 32 (vedere a pag. 130). In effetti essi sono così compatibili che molti programmi girano tanto sull'uno che sull'altro apparecchio.
Vi sono anche altre somiglianze: hanno lo stesso tipo di porta cartucce e la stessa CPU, una 6809E. È questa un'unità di elaborazione molto potente, usata in pochi altri home computer. Ma per certi aspetti questa CPU è scarsamente compatibile con le altre componenti della macchina. Normalmente la 6809 viene utilizzata in macchine progettate per uso professionale e, in un Tandy Color, che funziona con un clock a soli 895 KHz, non viene sfruttata al suo massimo.
La macchina è corredata da una propria versione di BASIC: il Tandy Color BASIC. Nonostante la bassa velocità di temporizzazione, questo BASIC è abbastanza veloce e possiede parecchi comandi speciali, che facilitano l'uso dell'hardware. Una delle caratteristiche di questo computer è il chip di controllo video 6847, un circuito programmabile che, solitamente, produce un formato video di 16 linee di 32 caratteri. Normalmente, i caratteri sullo schermo vengono visualizzati



La tavoletta grafica del Tandy
La tavoletta grafica GT-116 può essere usata sia come digitalizzatore per trasferire disegni e immagini nella memoria del computer, sia come tavola di selezione per un menu visualizzato sullo schermo. La tavoletta utilizza fogli di normale formato A4



La tastiera del Tandy
Il Tandy Color non ha una normale tastiera tipo macchina da scrivere, possiede invece 53 bottoni quadrati. Questi sono adeguatamente spaziati, sebbene non risultino molto comodi durante un uso prolungato

Connettori per joystick
I joystick sono di tipo "proporzionale" e sono dotati di una linea di segnalazione generalmente collegata ad un bottone. Queste interfacce possono essere usate anche per altri segnali analogici, come quelli derivanti da esperimenti di laboratorio

Interruttore generale

Componenti dell'alimentatore
L'uscita dal trasformatore viene stabilizzata da questi componenti. Nel processo si genera una certa quantità di calore, che viene disperso da appositi dissipatori

Il trasformatore
L'alimentatore interno ha un assetto molto ordinato e compatto

Porta seriale RS-232
Ad essa possono essere collegate stampanti, modem ed altri apparecchi seriali. Il computer non prevede l'impiego di input o output paralleli

Interfaccia per registratore
Questa interfaccia permette anche il controllo a distanza del motore

Selettore di canale
L'uscita TV può essere sintonizzata su due differenti canali selezionabili cambiando posizione all'interruttore

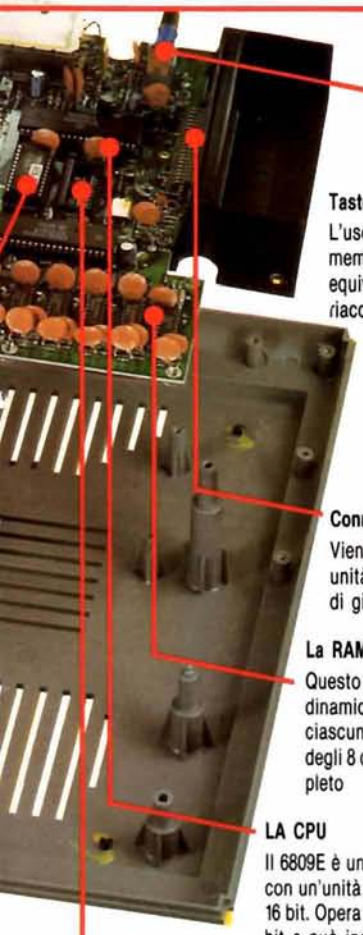
ROM da 8K
Contiene il BASIC Tandy Color ed alcuni programmi di gestione interna

Chip di controllo video 6847
Programmando questo chip in BASIC o in linguaggio macchina, è possibile definire la configurazione ed il colore del video

Connettore per tastiera
Un cavetto piatto serve per collegare la tastiera al computer, tramite questa porta



Oscillatore per video
La frequenza generata da questo cristallo determina la dimensione e il numero di caratteri visualizzabili sullo schermo. Esso serve, inoltre, anche all'effettiva produzione delle immagini



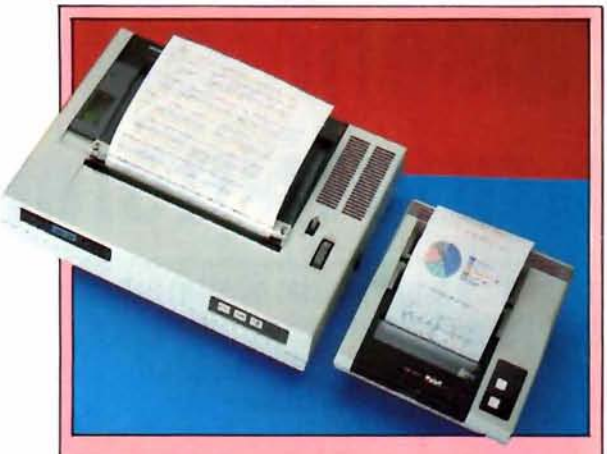
Tasto di RESET
L'uso di questo tasto azzera la memoria della macchina ed equivale a spegnere e quindi riaccendere l'apparecchio

Connettore per cartucce
Viene usato per collegare sia le unità a dischetti che le cartucce di giochi e programmi

La RAM
Questo computer ha 16K di RAM dinamica distribuiti su 8 chip, ciascuno dei quali contiene 1 bit degli 8 che formano un byte completo

LA CPU
Il 6809E è un chip d'avanguardia con un'unità aritmetica interna a 16 bit. Opera con un bus dati di 8 bit e può indirizzare 64K di memoria

Zoccolo per ROM aggiuntiva
Può essere usato per espandere le prestazioni o per estensioni del BASIC



Le stampanti Tandy
Il Tandy Color è corredato da una serie di stampanti commercializzate dalla stessa Tandy. La CGP-220, ad esempio, usa la tecnica a getto d'inchiostro per stampare in 7 colori ed è molto silenziosa. Di prezzo assai più basso è la CGP-115, che utilizza 4 penne a sfera miniaturizzate per produrre stampati e tracciare grafici a colori

in nero su sfondo verde, ma si possono selezionare 9 diversi colori ed il formato video può essere modificato.
Il computer Tandy Color impiega lo stesso tipo di connettore per cartucce ROM del Dragon 32, ed esiste una vasta gamma di software disponibile su questo genere di supporto: c'è la consueta varietà di giochi (dama, scacchi, carte, "caccia" al petrolio, flipper, ecc.). Il software disponibile comprende inoltre cartucce per musica, aritmetica e dattilografia. C'è anche una discreta gamma di programmi per applicazioni finanziarie e per l'elaborazione di testi, come pure alcuni utili sussidi per lo sviluppo di programmi in BASIC o in linguaggio macchina. Infine, è presente un programma diagnostico su ROM, progettato per verificare il corretto funzionamento del computer.
Un'interfaccia seriale RS232, posta sul retro della macchina, consente l'uso di varie periferiche, oltre ovviamente a stampanti seriali. Questa porta può anche essere usata per comunicare con altri computer tramite un modem (vedere a pag. 216). Con un simile apparecchio ed una cartuccia videotex è possibile accedere a varie banche dati (ad esempio quella del servizio Prestel).
Sono presenti anche 2 connettori per joystick, ciascuno con 2 assi di movimento e un pulsante di fuoco. Quest'ultimo sembra causare un piccolo inconveniente col Tandy Color BASIC: quando si preme il bottone, viene generata una serie di caratteri sullo schermo (inconveniente fastidioso, ma non grave).
I joystick sono migliori, poiché rispondono con continuità ai comandi. I circuiti di controllo potrebbero essere usati per interpretare altri segnali analogici di ingresso, come quelli prodotti da un esperimento di fisica.
L'espansione della macchina è ottenibile aggiungendo apposite cartucce. Si possono collegare fino a 4 unità a dischetti da 5 e 1/4 pollici, il che permette una memoria di massa con capacità totale di 626 Kbyte.

TANDY COLOR COMPUTER

DIMENSIONI
mm. 369 x 344 x 94

PESO
—

CPU
6809E

CLOCK
0,895 MHz

MEMORIA
4 Kbyte di RAM, 8 Kbyte di ROM (espandibile a 32 Kbyte)

VIDEO
16 linee di 32 caratteri, 9 colori disponibili (selezionabili indipendentemente per caratteri e sfondo). 127 caratteri predefiniti e 127 definibili dall'utente

INTERFACCE
Seriale RS232, per registratore, per TV (due canali), per due joystick proporzionali

LINGUAGGI FORNITI
BASIC

ALTRI LINGUAGGI DISPONIBILI
Linguaggio macchina 6809 con Assembler

ACCESSORI COMPRESI
Cavo di connessione per TV. Manuale d'installazione e manuale del BASIC

TASTIERA
53 tasti, con spaziatura tipo macchina da scrivere

DOCUMENTAZIONE
Il manuale operativo è piuttosto ricco d'informazioni ed è perfettamente adeguato ai principianti

L'inserimento di nuovi nominativi

Come la ricerca binaria ci può aiutare ad aggiungere nominativi ad una lista

Un file è costituito da più record, suddivisi a loro volta in più campi. Per accedere ad un particolare campo di un record si usa un "puntatore" o "indice". Vediamo adesso alcune delle tecniche impiegate per la ricerca di dati in una lista.
Creare dei record per la nostra "rubrica computerizzata" non è difficile. Supponiamo di usare una matrice alfanumerica per ogni campo del record: NOMIN$(), INDIR$(), LOCAL$(), e TEL$(). Spiegheremo più avanti perché convenga adoperare una matrice alfanumerica, anziché numerica, per il telefono. Nell'elenco di opzioni previste per la nostra rubrica, la sesta era quella relativa all'immissione di nuovi nominativi: selezionandola, l'esecuzione passerebbe alla parte di programma presentata come esercizio da svolgere.
Supponiamo che un certo numero di nominativi sia già stato inserito nella rubrica e che non ci si ricordi quanti. Poiché è molto importante che l'immissione di un nuovo nominativo non vada a sovrapporsi ad uno già esistente, una delle funzioni del programma deve essere quella di cercare, nelle matrici, il primo posto libero.
Questo tipo di ricerca non è difficile: le stringhe, in quasi tutte le versioni del BASIC, possono essere confrontate tra loro come accade con i valori numerici (scrivere, ad esempio, IF A$ = "CARLO" THEN... è altrettanto valido quanto IF A = 61 THEN...). Se un qualsiasi elemento delle nostre matrici alfanumeriche è occupato, esso conterrà almeno un carattere alfanumerico: pertanto, per cercare il primo elemento "vuoto", basta scorrere attraverso una matrice, finché non si trova un elemento che non contiene alcun carattere.
Noi prevediamo di impiegare quattro diverse matrici (nominativo, indirizzo, località e numero telefonico), ma non è affatto necessario effettuare la ricerca quattro volte: infatti, se il 15mo elemento della matrice relativa ai nominativi non contiene alcun valore, possiamo ragionevolmente supporre che anche i corrispondenti elementi di INDIR$(), LOCAL$() e TEL$() siano vuoti.
Se la variabile POSIZIONE viene usata per contenere il numero del primo elemento libero, un piccolo programma per individuarne il valore può essere, semplicemente:

PROCEDURA (trova 1° elemento libero)

```
INIZIO
  CICLO
    RIPETI FINCHÉ non si trova un elemento vuoto
    LEGGI Matrice (POSIZIONE)
    POSIZIONE = POSIZIONE + 1
    SE Matrice(POSIZIONE) = " "
      ALLORA annota il valore di POSIZIONE
      ALTRIMENTI passa oltre
    FINE SE
  FINE CICLO
FINE
```

Tradotto in BASIC:

```
1000 FOR L = 0 TO 1 STEP 0
1010 LET POSIZIONE = POSIZIONE + 1
1020 IF NOMIN$(POSIZIONE) = " " THEN LET L = X
1030 NEXT L
1040 REM resto del programma
```

Si noti che il valore X, nella linea 1020, viene usato per terminare il ciclo FOR-NEXT e che esso varia da computer a computer (vedere "Basic & Basic"). Inoltre, è molto importante notare che questo è un frammento di programma, dove si dà per scontato che, altrove nel programma principale, la matrice NOMIN$() sia stata già opportunamente DIMensionata e che la variabile POSIZIONE sia già stata inizializzata.
Sebbene abbiamo impiegato precedentemente la tecnica FOR X = 0 TO 1 STEP 0 per simulare una struttura REPEAT...UNTIL (ripeti...finché), è bene esaminarne il funzionamento in dettaglio. Finché il confronto contenuto nella linea 1020 fallisce, il valore di L (contatore del ciclo FOR-NEXT) rimane 0, mentre il valore di POSIZIONE viene incrementato di 1 ad ogni passaggio. Quando, però, la condizione alla linea 1020 è "vera" (ossia l'elemento di NOMIN$() è vuoto), ad L viene assegnato il valore di X, ed il ciclo termina alla linea 1030. Il valore contenuto nella variabile POSIZIONE, a questo punto, costi-

tuirà un "puntatore" al primo elemento "vuoto" di NOMIN$().
Nel nostro programma, il valore di POSIZIONE andrà determinato fin dall'inizio, ossia ogni volta che si usa il programma della rubrica. Inoltre, esso andrà aggiornato più volte durante l'uso del programma. POSIZIONE rappresenta, quindi, una variabile "globale", il cui valore dovrà essere stabilito dalla subroutine di "inizializzazione".
Possiamo stabilire il valore di POSIZIONE una volta per tutte, all'inizio del programma, oppure possiamo usare un "flag" (ossia un interruttore "logico"), che indichi al programma se e quando è necessario reinizializzare la variabile POSIZIONE.
Per il momento, adotteremo la procedura più semplice, cioè la prima.
Adesso riassumiamo, in maggiore dettaglio, la strategia generale di tutto il programma, supponendo che ogni particolare funzione venga svolta da una specifica subroutine (identificata dal nome racchiuso tra asterischi):

1. Trovare un record (dal nominativo) * TROVREC *
2. Trovare un nominativo (usando solo parte di esso) * TROVNOM *
3. Trovare un record (usando la località) * TROVCIT *
4. Trovare un record (usando le iniziali) * TROVINI *
5. Visualizzare l'elenco dei nominativi * LISTREC *
6. Aggiungere un record * AGGREC *
7. Modificare un record * MODREC *
8. Cancellare un record * CANREC *
9. Uscire dal programma (con aggiornamento) * USCPROG *

Grosso modo, sappiamo già quali saranno gli input e gli output del programma, possiamo quindi dedicarci alla struttura del programma principale. Tutto lo sviluppo dei particolari può avvenire seguendo la tecnica "top-down" precedentemente descritta. Sappiamo, tra l'altro, che ci occorre una subroutine di inizializzazione, che il programma si avvarrà di un menu di opzioni, tra le quali ne sceglieremo, a seconda del caso, almeno una.
Il corpo del programma principale può dunque avere la forma:

PROGRAMMA PRINCIPALE

```
INIZIO
  INIZIALIZZAZIONE (procedura)
  PRESENTAZIONE (procedura)
  SELEZIONE (procedura)
  ESECUZIONE (procedura)
FINE
```

In BASIC, possiamo scrivere (riservandoci in seguito di sostituire dei numeri di linea ai nomi delle subroutine):

```
10 REM RUBRICA COMPUTERIZZATA
20 GOSUB * INIZIALIZZAZIONE *
30 GOSUB * PRESENTAZIONE *
40 GOSUB * SELEZIONE *
50 GOSUB * ESECUZIONE *
60 END
```

La subroutine di PRESENTAZIONE, potrebbe prevedere semplicemente la visualizzazione di un messaggio di "benvenuto", al quale far seguire il menu delle opzioni. Ad esempio:

```
"QUESTA È LA"
"RUBRICA COMPUTERIZZATA"
"SVILUPPATA NEL CORSO DI PROGRAMMAZIONE"
"DE 'IL MIO COMPUTER'"
(PREMERE IL TASTO SPAZIO PER CONTINUARE)
```

Appena viene premuto il tasto spazio, il programma esegue la subroutine SELEZIONE, visualizzando quanto segue:

```
"MENU DELLE OPZIONI"

1. TROVARE UN RECORD (usando il nome)
2. TROVARE UN NOMINATIVO (usando parte del nome)
3. TROVARE PIÙ RECORD (usando la località)
4. TROVARE PIÙ RECORD (usando le iniziali)
5. LISTA DI TUTTI I RECORD
6. AGGIUNGERE UN RECORD
7. MODIFICARE UN RECORD
8. CANCELLARE UN RECORD
9. USCIRE DA PROGRAMMA

"DIGITARE UN VALORE TRA 1 E 9"
"SEGUITO DA UN <RETURN>"
```

Il programma, a questo punto, proseguirà in base alla selezione operata. Lentamente, la struttura sta prendendo una forma più precisa: ogni subroutine (eccetto quella che serve per uscire dal programma), dovrà terminare in modo da ritornare alla subroutine di SELEZIONE. Molti dettagli, sulla struttura dei dati da elaborare, verranno esaminati in seguito, per adesso continuiamo nella nostra panoramica generale.
Supponiamo che un certo numero di nominativi sia già presente nella rubrica e che noi desideriamo visualizzare un record usando il nome (opzione 1). Prima di progettare questa parte del programma, dobbiamo considerare alcuni importanti problemi relativi alla procedura computerizzata di ricerca.

La ricerca

I testi d'informatica, in genere, trattano la ricerca ed il riordino dei dati come argomenti inseparabili tra loro. C'è una buona ragione per questo: molti dei problemi che si incontrano sono, in effetti, comuni alle due procedure. Noi abbiamo già accennato al riordino di dati (vedere a pag. 134, quando si trattò di riordinare dei nomi in ordine alfabetico).

Se adoperassimo, come rubrica "manuale", un semplice quaderno senza indice alfabetico ed i nominativi (assieme alle altre informazioni) venissero annotati senza un preciso ordine, avremmo una struttura del tipo "a pila", ossia, i dati avrebbero un ordine che dipende esclusivamente da quello usato nell'annotarli via via.
Per trovare un particolare nominativo, dovremmo iniziare dal primo e scorrerli tutti, fino a quello cercato. Lo stesso può accadere con un archivio computerizzato.
Addirittura, talvolta i criteri di accesso sono talmente imprevedibili, che una simile struttura "disorganizzata" vale quanto una qualsiasi altra.
Ovviamente, il compito della ricerca risulta enormemente facilitato (sia in sistemi manuali che in quelli computerizzati), se viene adottata una qualche struttura organizzata dei dati, per esempio quella alfabetica (come negli elenchi telefonici).
Nella nostra rubrica computerizzata, i dati sono organizzati "a pila": un record è composto dall'elemento X della matrice dei nomi, dall'elemento X della matrice degli indirizzi e così via; il record successivo è composto dall'elemento X+1 della matrice dei nomi, e via dicendo.
Per trovare un nominativo in particolare (ad esempio: GIGI BASSI), dobbiamo partire dal primo elemento della matrice dei nomi e vedere se corrisponde a GIGI BASSI.
Se non corrisponde, si passa al secondo elemento, si ripete il confronto e si continua fino a che non si trova il record cercato (oppure scopriamo che GIGI BASSI non esiste nella rubrica!).
Facciamo un semplice esempio. In una classe di dieci alunni, ognuno di essi ha ottenuto un voto in geografia:

5,8,4,7,7,10,6,7,5,4

I voti sono riportati secondo l'ordine usato nel registro di classe, e non in ordine crescente di voto.
Per trovare chi abbia preso 10 in geografia, dovremo iniziare dal primo alunno ed operare una ricerca "sequenziale".
Una ricerca impostata su questo criterio, specialmente se la quantità di dati che compongono l'archivio è molto grande, può essere lenta a tal punto da rendere inutile avere un computer: tanto vale cercare il nome a mano!
Se, prima della ricerca, riordinassimo i valori, per esempio, in ordine crescente:

4,4,5,5,6,7,7,7,8,10

la procedura può essere semplificata seguendo un nuovo criterio: dividiamo questa serie di valori in due parti e cerchiamo il valore desiderato in quello che più probabilmente lo contiene. L'algoritmo, a parole, può essere così riassunto:

Trovare la matrice che contiene i voti

Stabilire quale voto desideriamo trovare

Trovare la lunghezza della matrice

Trovare il valore della posizione centrale della matrice

Eseguire un ciclo di ricerca fino a trovare il voto

  Se il valore alla posizione centrale è uguale a quello cercato la ricerca è conclusa

  Altrimenti, si guarda se il valore cercato è maggiore o minore di quello "mediano"

  Se è maggiore, si cerca la posizione centrale della metà superiore

  Se, invece, è minore, si cerca la posizione centrale della metà inferiore

  (Si ripetono queste operazioni fino a trovare l'elemento cercato)

Tradotto in pseudolinguaggio di programmazione:

```
INIZIO
  Trova la matrice dei voti
  INPUT del VOTO da cercare
  CICLO finché il voto non è localizzato
    SE VOTO = (posizione centrale)
    ALLORA annota posizione
    ALTRIMENTI
      SE VOTO > (posizione centrale)
      ALLORA trova posizione centrale superiore
      ALTRIMENTI trova posizione centrale inferiore
      FINE SE
    FINE SE
    SE VOTO è trovato
      ALLORA visualizza posizione
      ALTRIMENTI visualizza "NON TROVATO"
    FINE SE
FINE
```

Seguendo il percorso logico di questa procedura, si può notare che il VOTO cercato (se esiste) viene immancabilmente trovato. Questo metodo di ricerca, basato su successive partizioni dei dati, si chiama "ricerca binaria".
Vediamo come trasporre in BASIC la procedura che abbiamo espresso in pseudo linguaggio. Il programma crea una matrice di valori, chiede il valore da cercare ed esegue la ricerca.
Se la ricerca ha successo, viene visualizzata la posizione dell'elemento nella matrice.

```
10 REM RICERCA DI UN NUMERO IN UNA MATRICE
20 DIM VOTI(20)
30 FOR Z = 1 TO 20
40 READ VOTI(Z)
50 NEXT Z
60 DATA 3,3,4,4,4,5,5,5,5,6,6,6,6,8,8,8,8,8,9,10
```

```
70 LET L = 20
80 LET FONDO = 1
90 LET CIMA = L
100 INPUT "QUALE VOTO DEVO CERCARE";N
110 FOR Z = 0 TO 1 STEP 0
120 LET L = CIMA — FONDO
130 LET META = FONDO + INT(L/2)
140 IF N = VOTI(META) THEN LET Z = X
150 IF N > VOTI(META) THEN LET FONDO = META
160 IF N < VOTI(META) THEN LET CIMA = META
170 NEXT Z
180 PRINT "IL VOTO CERCATO È NELL'ELEMENTO N.";META
190 END
```

Si noti che il valore di X deve essere assegnato altrove nel programma e dipende dal particolare computer (vedere "Basic & Basic").
Se i dati sui quali operiamo la ricerca sono distribuiti omogeneamente, come accade più o meno in un elenco telefonico, una ricerca binaria è un metodo piuttosto efficiente e molto rapido per ricercare un particolare dato.
Comunque, questo sistema non è né l'unico né il migliore: esistono degli algoritmi che permettono di individuare un dato con un numero molto inferiore di "iterazioni" (ripetizioni della procedura di ricerca). Uno di questi è il metodo basato sull'"hashing" (ossia sull'indirizzamento casuale dei dati): il programma tenta di "indovinare" la posizione di un particolare dato, raffinando, ad ogni tentativo, la "mira". Per quanto riguarda la nostra rubrica, però, è sufficiente una ricerca binaria.

Esercizi

Se facciamo girare il programma appena presentato, si potranno notare alcuni inconvenienti. In primo luogo, se chiediamo di trovare un valore che non esiste (7 ad esempio), il programma non si ferma più e non viene visualizzato alcun messaggio d'errore. Inoltre, se si chiede il valore 9, che esiste nella matrice, il programma non riesce a localizzarlo. Infine, il programma si aspetta che ogni valore contenuto nella matrice sia diverso da tutti gli altri, mentre in effetti vi sono molte ripetizioni. Di conseguenza, non visualizza tutte le posizioni nelle quali appare il valore da noi cercato (né ci avverte che ne esiste più d'uno).
Per esercizio, quindi:

1. Trovare perché il programma non riesce a localizzare il valore 9
2. Si modifichi una linea di programma, affinché l'errore non si verifichi
3. Trovare la ragione del mancato funzionamento nel caso di numeri richiesti ma non esistenti nella matrice e si prospetti un possibile rimedio.

Basic & Basic

SPECTRUM

Per lo Spectrum, il primo frammento di programma è il seguente:

```
100 DIM F$(6,5)
110 LET POSIZIONE = 0
120 LET F$(1) = "MARIA"
130 LET F$(2) = "MARTA"
140 LET F$(4) = "GABRI"
1000 FOR L = 0 TO 1 STEP 0
1010 LET POSIZIONE = POSIZIONE + 1
1020 IF F$(POSIZIONE) = " " THEN LET L = 2
1030 NEXT L
1040 PRINT "POSIZIONE DEL 1MO ELEMENTO VUOTO";POSIZIONE
1050 STOP
```

Si noti che le linee da 100 a 140 trasformano il frammento (linee 1000-1030) in un programma dimostrativo eseguibile. I valori contenuti in queste linee d'"inizializzazione" si possono variare, per meglio verificare il funzionamento della parte restante.
Il secondo programma, per lo Spectrum, risulta:

```
10 REM RICERCA DI UN NUMERO IN UNA MATRICE
20 DIM V(20)
30 FOR Z = 1 TO 20
40 READ V(Z)
50 NEXT Z
60 DATA 3,3,4,4,4,5,5,5,5,6,6,6,6,8,8,8,8,8,9,10
70 LET L = 20
80 LET FONDO = 1
90 LET CIMA = L
100 INPUT "QUALE VOTO DEVO CERCARE";N
110 FOR Z = 0 TO 1 STEP 0
120 LET L = CIMA — FONDO
130 LET META = FONDO + INT(L/2)
140 IF N = V(META) THEN LET Z = 2
150 IF N > V(META) THEN LET FONDO = META
160 IF N < V(META) THEN LET CIMA = META
170 NEXT Z
180 PRINT "IL VOTO CERCATO È NELL'ELEMENTO N.";META
190 END
```

VARIABILI

Per le variazioni da apportare, vedere "Basic & Basic" a pag. 257

STEP 0

Ecco i valori da sostituire alla X nei programmi riportati nel testo:

Apparecchio	Valore di X
Oric-1	1
Dragon 32	1
Lynx	2
Micro BBC	2
Commodore 64 e Vic-20	1

LYNX

I due programmi riportati nel testo (tenuto conto di quanto detto sopra circa le VARIABILI e lo STEP 0) funzionano sul Micro BBC, sul Dragon 32, sull'Oric-1, sul Commodore 64 e sul Vic-20. Le linee iniziali usate per lo Spectrum (vedi sopra) possono essere impiegate come traccia anche per gli altri computer. Per il Lynx sostituire la linea 100 con:

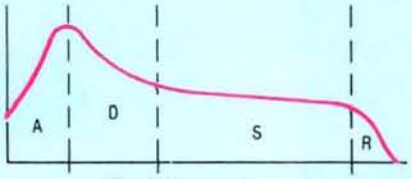
```
100 DIM F$(4) (6)
```

Consigli sul suono

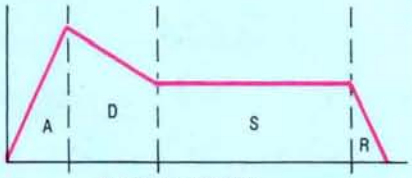
Ancora qualche principio di musica elettronica

I generatori d'inviluppo

L'inviluppo di un suono è la configurazione delle sue variazioni di volume sonoro dall'istante in cui l'emissione inizia a quello in cui cessa. Inviluppi simili a quelli di una nota di pianoforte e di altri strumenti sono mostrati nel diagramma. Gli inviluppi vengono comunemente suddivisi in 4 parti chiamate di solito: Attacco, Decadimento, Sustain (Prolungamento), Rilasciamento (ADSR). Nel caso del pianoforte, il volume, dopo che il tasto è stato battuto, sale rapidamente al suo livello più alto (attacco), quindi si abbassa lentamente (decadimento) ad un volume quasi costante mentre il tasto è tenuto abbassato (prolungamento) ed infine cade rapidamente a zero (rilasciamento), quando il tasto viene lasciato andare. Si noti che il "prolungamento" è un livello di volume, mentre gli altri tre sono intervalli di tempo. Uno strumento che permette di controllare tutti e quattro gli aspetti di un inviluppo è chiamato "generatore ADSR".

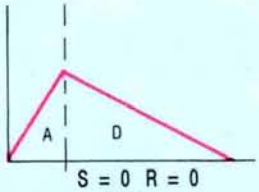


Pianoforte naturale

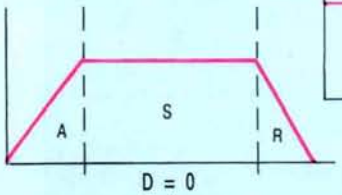


Pianoforte artificiale

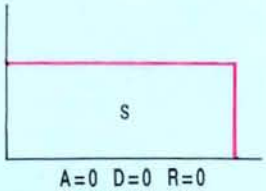
Molti computer possiedono soltanto un generatore di "beep", cioè un cicalino, che produce un suono di volume costante e di durata definita. In questo caso gli elementi A, D ed R sono uguali a 0.
L'Oric-1 è dotato di un qualche controllo di inviluppo ma, da questo punto di vista, gli home computer più versatili sono il BBC Micro e il Commodore 64, entrambi in possesso di generatori ADSR capaci di imitare gli inviluppi di molti comuni strumenti o di crearne dei nuovi "artificiali".



S = 0 R = 0
Xilofono



D = 0
Violino



A=0 D=0 R=0
Organo elettronico

... e sulla grafica

Animazioni semplici con le istruzioni POKE e PRINT

Ci siamo già interessati dei principi fondamentali della grafica dei computer: esamineremo ora come sia possibile realizzare alcune semplici animazioni.
Il programmatore, in BASIC, può avvalersi di due metodi principali: il comando POKE ed il comando PRINT con le sue funzioni associate. Il comando POKE deposita un numero nella locazione di memoria specificata. In tutti i computer c'è uno speciale gruppo di locazioni nella memoria, ciascuna in diretto rapporto con la specifica posizione di un carattere sullo schermo. Per un video standard (25 linee × 40 colonne), viene utilizzata una porzione di memoria di 1000 locazioni. Ogni locazione è contrassegnata da un numero che corrisponde ad un particolare carattere nel set di caratteri. Questo numero può essere il codice standard ASCII per quel carattere (vedere a pag. 214) o un codice stabilito dalla casa produttrice della macchina.
Sul Commodore 64, per esempio, ci sono pochissimi comandi BASIC a disposizione per la grafica e viene usato il comando POKE per creare immagini sullo schermo. Gli indirizzi delle locazioni che contengono i codici dei caratteri, vanno da 1024 a 2023 e le locazioni che contengono le informazioni del colore hanno indirizzi da 55296 a 56295. Perciò il carattere A possiede codice di video 1 e il colore nero è rappresentato dal codice di colore 0. Pertanto, i comandi necessari per posizionare la lettera A in nero nell'angolo in alto a sinistra dello schermo saranno:

```
10 POKE 1024,1
20 POKE 55296,0
30 END
```

Mediante semplici modifiche sarà possibile visualizzare una riga di A in nero sulla prima linea di video:

```
10 FOR X = 0 TO 39
20 POKE 1024 + X,1
30 POKE 55296 + X,0
40 NEXT X
50 END
```

Le A sono prodotte dal ciclo FOR-NEXT, che, ad ogni passaggio, incrementa di 1 gli indirizzi di locazione del carattere e del colore. Se aggiungiamo un comando per cancellare la A già presente ogni volta che ne compare una nuova sullo schermo, allora sembrerà che la A corra attraverso il video; questa è una forma primitiva di animazione!

Sul Commodore il codice per il carattere "spazio" è 32. Ciò che il programma deve fare è dunque collocare nella locazione appropriata (cioè nella locazione di memoria immediatamente precedente a quella occupata dalla nuova A) il carattere spazio. Al programma precedente aggiungiamo questa linea:

```
35 POKE 1024 + X,32
```

alternativamente:

```
15 POKE 1024 + (X—1),32
```

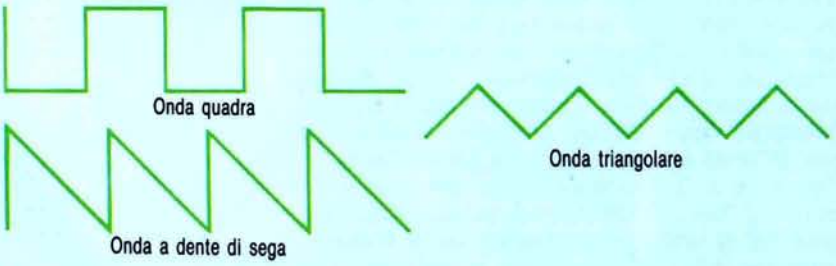
Ottenere immagini grafiche con questo sistema è piuttosto laborioso: probabilmente è più efficace per creare immagini statiche di sfondo, utilizzando prima i comandi READ e DATA per inserire i codici dei caratteri e quindi il comando POKE per collocarli nelle corrette locazioni.
Molti computer possiedono particolari istruzioni BASIC per la grafica: grazie ad esse l'utente può tracciare punti sullo schermo e poi congiungerli con linee rette, disegnare quadrati, archi e cerchi. È inoltre possibile colorare l'interno delle figure disegnate.
Per cancellare un'immagine dallo schermo, viene usato un procedimento molto semplice: si ridisegna la figura con lo stesso colore dello sfondo. Quindi, alla base di una semplice animazione grafica ci sono tre processi: il rapido disegno di un'immagine, la sua cancellazione e la sua nuova comparsa, sullo schermo, in una diversa posizione. Il realismo dell'azione dipende molto dalla velocità con cui questa sequenza può essere sviluppata. Gli "sprite" sono molto più efficienti perché non richiedono cancellazione quando vengono spostati in una nuova posizione: questo fatto accresce molto la velocità a cui sembrano muoversi. Inoltre la possibilità di impiegare gli sprite significa che oggi è finalmente possibile scrivere programmi di giochi tipo sala giochi in linguaggio BASIC, mentre prima era indispensabile il linguaggio macchina.
I principi dell'animazione possono essere usati insieme a semplici programmi BASIC. Molti home computer hanno comandi che permettono l'uso della PRINT ad una specifica posizione dello schermo, come ad esempio PRINT AT sullo Spectrum o PRINT@ sul Dragon.

Formazione di una stella
Questo è un piccolo programma per il Dragon che usa l'istruzione PRINT. In esso, la variabile X è la locazione dello schermo dove comparirà la stella. Notare che la linea 40 cancella la vecchia stella nel momento in cui la nuova compare sul video.

```
10 CLS: REM CLEAR SCREEN
20 FOR X = 160 TO 191
30 PRINT@X,"*";
40 PRINT@X—1," ";
50 NEXT X
60 END
```

Le forme d'onda

Una forma d'onda è, appunto, la "forma" ripetitiva del segnale emessa da un oscillatore (vedere a pag. 247): essa determina il carattere di un particolare suono. Due strumenti diversi, che suonino note dello stesso tono, non emettono lo stesso suono e ciò accade, in parte, perché le forme d'onda prodotte non sono uguali. Le più comuni forme d'onda sono l'onda quadra, quella triangolare e quella "a dente di sega"), come mostrato nella figura.



La maggior parte degli home computer produce una sola forma d'onda, solitamente quella quadra, molto ricca di armoniche e ciò spiega perché il suono generato è stridente. Il Commodore 64 è, attualmente, il più interessante computer musicale, soprattutto perché è possibile selezionare a piacimento una delle tre forme d'onda base su ciascuno dei tre oscillatori. Le forme d'onda possono essere modificate con filtri, che alterano il tono in modo molto simile ai controlli bassi/alti degli apparecchi ad alta fedeltà. Ciò permette di simulare suoni naturali con maggior fedeltà e di produrre nuovi suoni più stimolanti.

Il rumore

Il rumore è un tipo di suono complesso prodotto da vibrazioni casuali. Immaginiamo qualche suono comune come la pioggia, il vento e il tuono. Questi rumori non sono simili tra loro in quanto prodotti da una combinazione di rumore puro (e quindi casuale) con qualche tono dominante. La maggior parte dei computer con capacità di produrre rumore permette quindi di modulare il rumore stesso o di mixarlo con note pure. La gamma di effetti possibili varia da "vento fischiante" a violente esplosioni.

L'output

L'output è inviato, normalmente, all'altoparlante del televisore. Alcuni computer, tuttavia, possono emettere i suoni solo attraverso un piccolo altoparlante interno. In questo caso è impossibile ottenere una buona qualità di riproduzione senza modificare l'hardware del computer o acquistare un modulo esterno. Alcuni computer, però, possiedono un'uscita collegabile direttamente all'impianto ad alta fedeltà: ciò ripagherà piacevolmente l'impegno profuso nella produzione di forme d'onda complesse.

I display a cristalli liquidi

Un sistema alternativo, ed economico, per visualizzare dati

I display a cristalli liquidi (Liquid Crystal Display, o, più brevemente, LCD) sono stati impiegati, dal 1973, nelle calcolatrici tascabili. Più tardi sono stati usati nella produzione di orologi digitali ed hanno oggi un vasto impiego anche nell'industria dei microcomputer. Vengono usati nei "portatili", quali l'HX-20 della Epson (vedere a pag. 169), nel computer portatile Tandy TRS80 Modello 100 ed anche nel PC5000 ad alta capacità della Sharp.

Per capire il funzionamento dei display a cristalli liquidi, dobbiamo prima ricordare uno dei principi fondamentali della materia: essa varia il suo stato fisico a seconda della temperatura e della pressione cui è sottoposta. Da solida (o cristallina), può divenire gassosa, passando attraverso lo stato liquido. Solo allo stato solido si trova un qualche allineamento molecolare di un elemento. Fanno eccezione pochissime sostanze che mantengono, in parte, questo allineamento regolare anche allo stato liquido. Queste sostanze, la cui composizione chimica è un segreto industriale strettamente protetto, sono note col nome di cristalli liquidi.

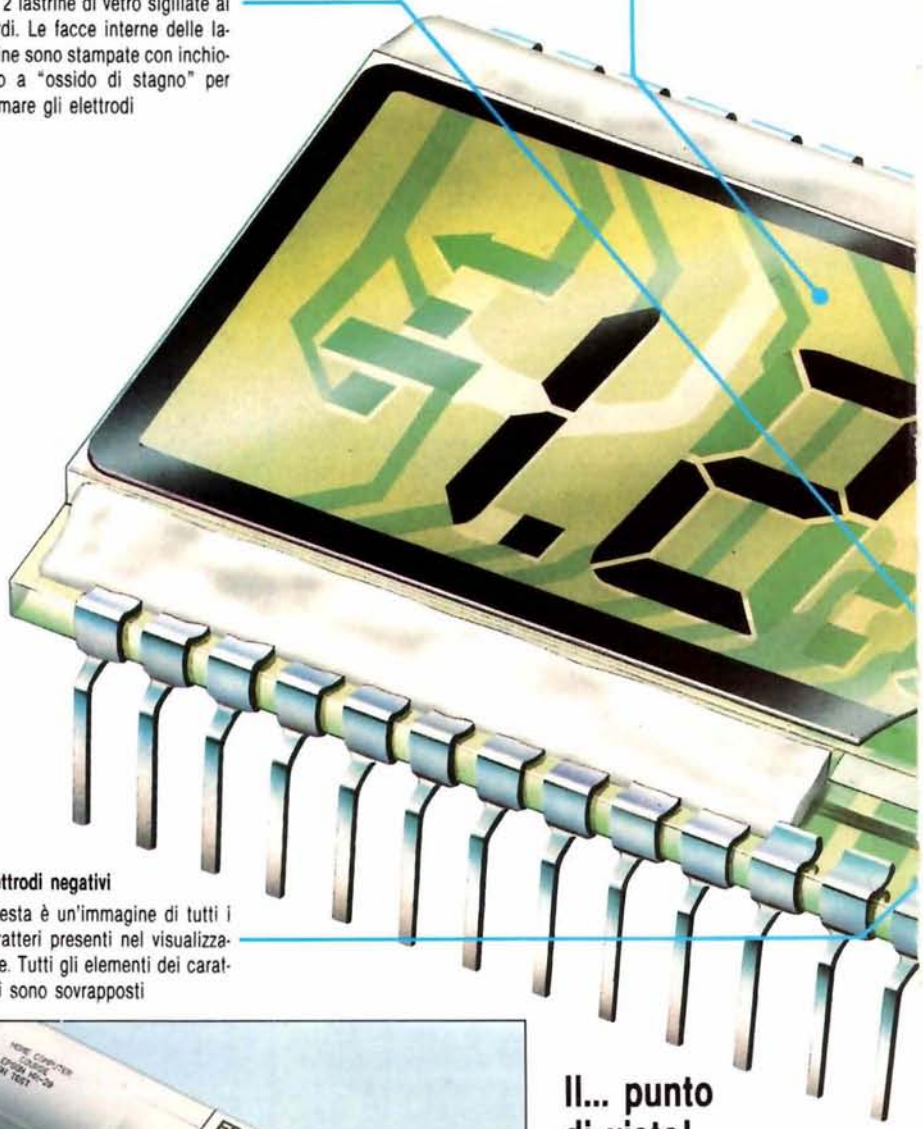
Sino alla metà degli anni Settanta, i visualizzatori sui calcolatori e sugli orologi erano composti da Diodi Fotoemittenti (Light Emitting Diode, da cui la sigla LED), disposti in modo da formare un'immagine piuttosto geometrica di lettere o numeri. I LED presentano però diversi inconvenienti: necessitano di una notevole quantità di corrente ed hanno dimensioni piuttosto ingombranti.

Nella ricerca di metodi alternativi per la visualizzazione di informazioni, si scoprì che l'allineamento molecolare nei cristalli liquidi poteva essere modificato da una corrente elettrica. Questa alterazione, inoltre, era di tipo puramente locale. Stabilito questo principio, fu possibile costruire uno strumento capace di visualizzare informazioni. Il primo passo fu quello di costruire degli elettrodi, a forma di caratteri, interposti tra due piccole lastre di vetro: tra di esse fu inserito un sottilissimo strato di cristalli liquidi e venne quindi applicata una tensione. Esposti alla normale luce ambientale sembrava che nulla accadesse, ma quando vi si aggiunsero dei filtri polarizzati (vedere disegno) ed uno sfondo riflettente, allora si ottenne l'effetto desiderato, ossia la comparsa di un carattere molto ben definito su uno sfondo neutro.

Il procedimento col quale viene definito un carattere richiede che la luce attraversi il primo filtro e che venga quindi polarizzata verticalmente. Essa viene pertanto deflessa di 90 gradi ed, infine, bloccata dal filtro

Filtro polarizzante verticale
Posto nella parte anteriore del display, "taglia" tutte le onde luminose eccetto quelle che oscillano sul piano verticale. Normalmente, filtra anche i raggi ultravioletti, per prolungare la vita dei cristalli liquidi

Cristalli liquidi
I cristalli liquidi vengono inseriti tra 2 lastrine di vetro sigillate ai bordi. Le facce interne delle lastrine sono stampate con inchiostro a "ossido di stagno" per formare gli elettrodi



Elettrodi negativi
Questa è un'immagine di tutti i caratteri presenti nel visualizzatore. Tutti gli elementi dei caratteri sono sovrapposti

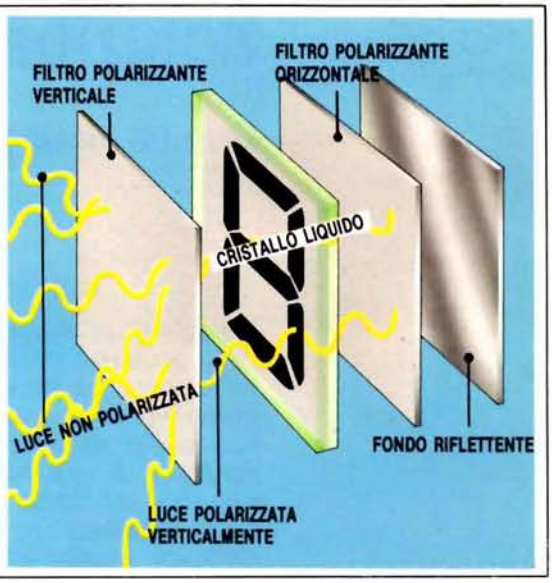


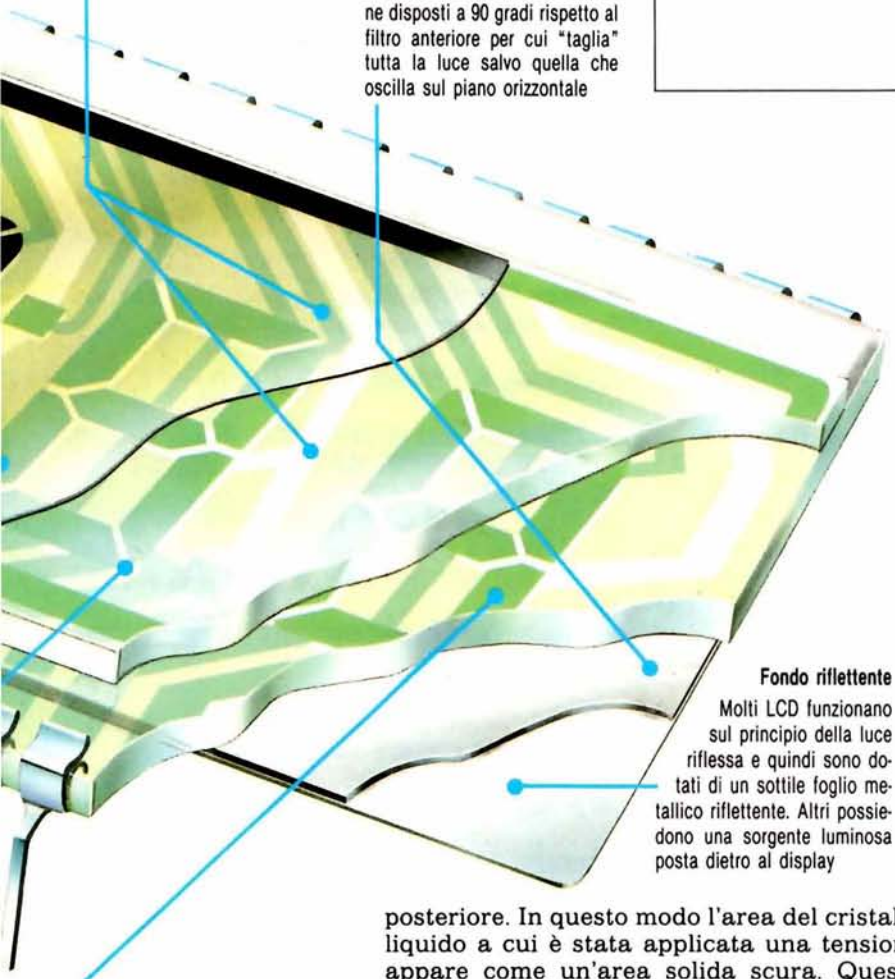
Il... punto di vista!

Una delle raffinatezze del compatto e sofisticato HX-20, il computer portatile della Epson, è il regolatore dell'angolo di visuale. I cristalli liquidi sono costituiti da lunghe e sottili molecole che presentano, a metà dei loro assi maggiori, dei poli magnetici. L'applicazione di una tensione elettrica produce una loro completa rotazione, mentre la loro naturale stabilità tende a mantenerli in posizione fissa. Maggiore è la quantità di corrente immessa, maggiore sarà la rotazione che essi subiranno

La polarizzazione

Il filtro anteriore di un LCD permette il passaggio di onde luminose con oscillazione elettromagnetica orientata verticalmente. Il cristallo liquido, quindi, ruota la polarizzazione di 90 gradi permettendo alla luce di passare attraverso il filtro posteriore (orizzontale) e, perciò, di essere riflessa. Quando una "barretta" di LCD viene attraversata da una corrente, essa non devia la luce polarizzata che l'attraversa, dando come risultato un'immagine nera





Piste di collegamento
Ciascun elettrodo è collegato elettricamente al relativo circuito da uno strato quasi invisibile di inchiostro conduttore, posto sulla faccia interna della lastrina del vetrino anteriore o posteriore

Filtro polarizzante orizzontale
Il filtro sul retro del visualizzatore possiede assi di polarizzazione disposti a 90 gradi rispetto al filtro anteriore per cui "taglia" tutta la luce salvo quella che oscilla sul piano orizzontale

Fondo riflettente
Molti LCD funzionano sul principio della luce riflessa e quindi sono dotati di un sottile foglio metallico riflettente. Altri possiedono una sorgente luminosa posta dietro al display

Elettrodi positivi
In questa "rappresentazione" dei caratteri, ciascun componente è autonomo ed individualmente indirizzabile mediante i circuiti di conduzione

posteriore. In questo modo l'area del cristallo liquido a cui è stata applicata una tensione appare come un'area solida scura. Questo stesso metodo è adottato nella fabbricazione degli LCD. Gli elettrodi vengono letteralmente stampati usando un inchiostro chiaro, quasi incolore, sulla superficie interna del vetro ed essiccati fino a divenire quasi invisibili.

Poiché è necessario visualizzare una certa gamma di caratteri in una stessa posizione, viene spesso impiegato il vecchio sistema "a barrette", ma si vanno sempre più diffondendo gli LCD a matrice di punti.

Ciascun punto della matrice è indirizzabile individualmente (un poco come accade nei televisori), quindi le possibilità grafiche sono molto progredite ed è possibile, con questo sistema, produrre sia forme grafiche continue che caratteri alfanumerici.

È teoricamente possibile utilizzare matrici a punti LCD indirizzate come schermi per ricezione televisiva o come monitor per i computer. Il principale inconveniente è l'impossibilità di variare il contrasto. Questo è stato uno dei problemi più grossi nello sviluppo dei televisori a schermo piatto, oggi in commercio. Per ottenere una capacità di risoluzione accettabile, infatti, occorre che le dimensioni di ciascun elemento della matrice LCD siano ridotte a qualcosa di molto simile a un pixel su un tubo a raggi catodici. Un'altra possibile soluzione richiederebbe l'uso di più LCD sovrapposti ed operanti simultaneamente. Questo metodo è attualmente usato per visualizzare immagini complesse in spazi limitati.

Il tempo di risposta di un LCD di alta qualità, che operi a temperatura normale (20 gradi centigradi), è di circa 70 millisecondi per il passaggio dal neutro al nero e di altri 80 millisecondi per il ritorno dal nero al neutro. In totale, 150 millisecondi: un tempo decisamente troppo lungo se confrontato con quello di un tubo a raggi catodici, che si aggira su 0,00025 millisecondi. Nonostante ciò, gli LCD presentano numerosi vantaggi. Si è già parlato della loro forma compatta, ma il vantaggio più rilevante è certamente il loro basso assorbimento: tipicamente, un LCD consuma circa 10 microwatt per ogni cm^2 di schermo.

Un altro vantaggio si manifesta in condizioni di forte illuminazione solare. Il contrasto di uno schermo a raggi catodici, in simili circostanze, risulta drasticamente diminuito, ma, poiché i cristalli liquidi sono visibili proprio perché non riflettono la luce incidente, essi appaiono ancor più contrastati e quindi più leggibili con l'aumentare della luce.

Nonostante abbia solo 10 anni, la tecnologia degli LCD ha già fatto significativi passi avanti nei mercati tradizionalmente riservati ai tubi a raggi catodici. Con l'aumentare delle esigenze della miniaturizzazione, c'è da aspettarsi che questa tecnologia si svilupperà ancor più in un futuro molto prossimo.

Soluzioni degli esercizi di riepilogo

Ecco un modo per risolvere gli esercizi di pagina 235, è il migliore?

A pagina 235 abbiamo presentato una serie di problemi, per esercitarsi sull'impiego delle istruzioni imparate. In questa pagina proponiamo le relative soluzioni: poiché non esiste un solo modo per stilare un programma, queste non sono le uniche soluzioni (né, forse, le migliori!).
Se le soluzioni qui presentate dovessero risultare enigmatiche quanto i problemi proposti, consigliamo di rileggere e studiare dalla pagina 149 in poi.

```
100 REM ESERCIZIO DI RIEPILOGO N.1
200 INPUT "PRIMO NUMERO";A
300 INPUT "SECONDO NUMERO";B
400 LET C = A + B
500 PRINT "A LORO SOMMA È ";C
```

```
100 REM ESERCIZIO DI RIEPILOGO N.2
200 LET A$ = "PRIMAPAROLA"
300 LET B$ = "SECONDAPAROLA"
400 LET C$ = A$ + B$
500 PRINT C$
```

```
100 REM ESERCIZIO DI RIEPILOGO N.3
200 INPUT "IMMETTERE UNA PAROLA";W$
300 LET L = LEN(W$)
400 PRINT "LA PAROLA È COMPOSTA DA ";L;" CA-
    RATTERI"
```

```
100 REM ESERCIZIO DI RIEPILOGO N.4
200 PRINT "DIGITARE UN CARATTERE QUALSIASI"
300 FOR C = 0 TO 1 STEP 0
400 LET A$ = INKEY$
500 IF A$ <> " " THEN LET C = 2
600 NEXT C
700 PRINT "IL CODICE ASCII DI ";A$;" È ";ASC(A$)
```

Vedere "Basic & Basic" alle pagg. 175 e 215. Per lo Spectrum sostituire la linea 700 con:

```
700 PRINT "IL CODICE ASCII DI ";A$;" È ";CODE(A$)
```

```
100 REM ESERCIZIO DI RIEPILOGO N.5
200 INPUT "IMMETTERE UNA PAROLA";W$
300 LET L$ = RIGHT$(W$,1)
400 PRINT "L'ULTIMO CARATTERE DELLA PAROLA È
    ";L$
```

Vedere "Basic & Basic" a pagina 149. Per lo Spectrum il programma diventa:

```
10 REM ESERCIZIO DI RIEPILOGO N.5
200 INPUT "IMMETTERE UNA PAROLA";W$
250 LET N = LEN(W$)
300 LET L$ = W$(N)
400 PRINT "L'ULTIMO CARATTERE DELLA PAROLA È
    ";L$
```

```
100 REM ESERCIZIO DI RIEPILOGO N.6
200 PRINT "IMMETTERE UN NOMINATIVO NELLA FOR-
    MA"
300 PRINT "GIOVANNI MARCHI"
400 INPUT "NOMINATIVO";N$
500 LET S = 0 : LET L = LEN(N$)
600 FOR P = 1 TO L
700 IF MID$(N$,P,1) = " " THEN LET S = P
800 NEXT P
900 PRINT "LO SPAZIO SEPARATORE È ALLA ";S;"
    POSIZIONE"
```

Vedere "Basic & Basic" a pag. 149. Per lo Spectrum sostituire la linea 700 con:

```
700 IF N$(P) = " " THEN LET S = P
```

```
100 REM ESERCIZIO DI RIEPILOGO N.7
200 INPUT "IMMETTERE UNA BREVE FRASE";N$
300 LET C = 1 : LET L = LEN(N$)
400 IF L = 0 THEN GOTO 200
500 FOR P = 1 TO L
600 IF MID$(N$,P,1) = " " THEN LET C = C + 1
700 NEXT P
800 PRINT "LA FRASE È COMPOSTA DA ";C;" PARO-
    LE"
```

Vedere "Basic & Basic" a pag. 149. Per lo Spectrum sostituire la linea 600 con:

```
600 IF N$(P) = " " THEN LET C = C + 1
```

```
100 ESERCIZIO DI RIEPILOGO N.8
200 FOR C = 128 TO 255
300 PRINT "IL CARATTERE CON CODICE ";C;" È
    ";CHR$(C)
400 REM CICLO PER RALLENTARE LA VISUALIZZAZIO-
    NE
500 FOR D = 1 TO 500
600 NEXT D
700 REM FINE DEL CICLO DI RITARDO
800 NEXT C
```

```
100 ESERCIZIO DI RIEPILOGO N.9
200 INPUT "DIGITA UN NUMERO";N
300 IF N < 32 OR N > 127 THEN GOTO 600
400 PRINT "AL CODICE ";N;" CORRISPONDE IL CA-
    RATTERE";CHR$(N)
500 GOTO 200
600 PRINT "CARATTERE NON VISUALIZZABILE"
700 GOTO 200
```

Un servo obbediente

I robot: il passo dalla fantascienza alla realtà non è stato lungo...

Il termine "robot" deriva dalla parola cecoslovacca "robota", che significa "lavoratore". Fu coniato dal commediografo Karel Čapek nel suo lavoro R.U.R. (Rossum's Universal Robots) del 1920 e fu quindi entusiasticamente adottato dagli scrittori di fantascienza. Molti sono i racconti fantastici sui poteri dei robot, ma essi non sono altro che estensioni elettromeccaniche dei computer, con tutti i loro limiti e difetti.

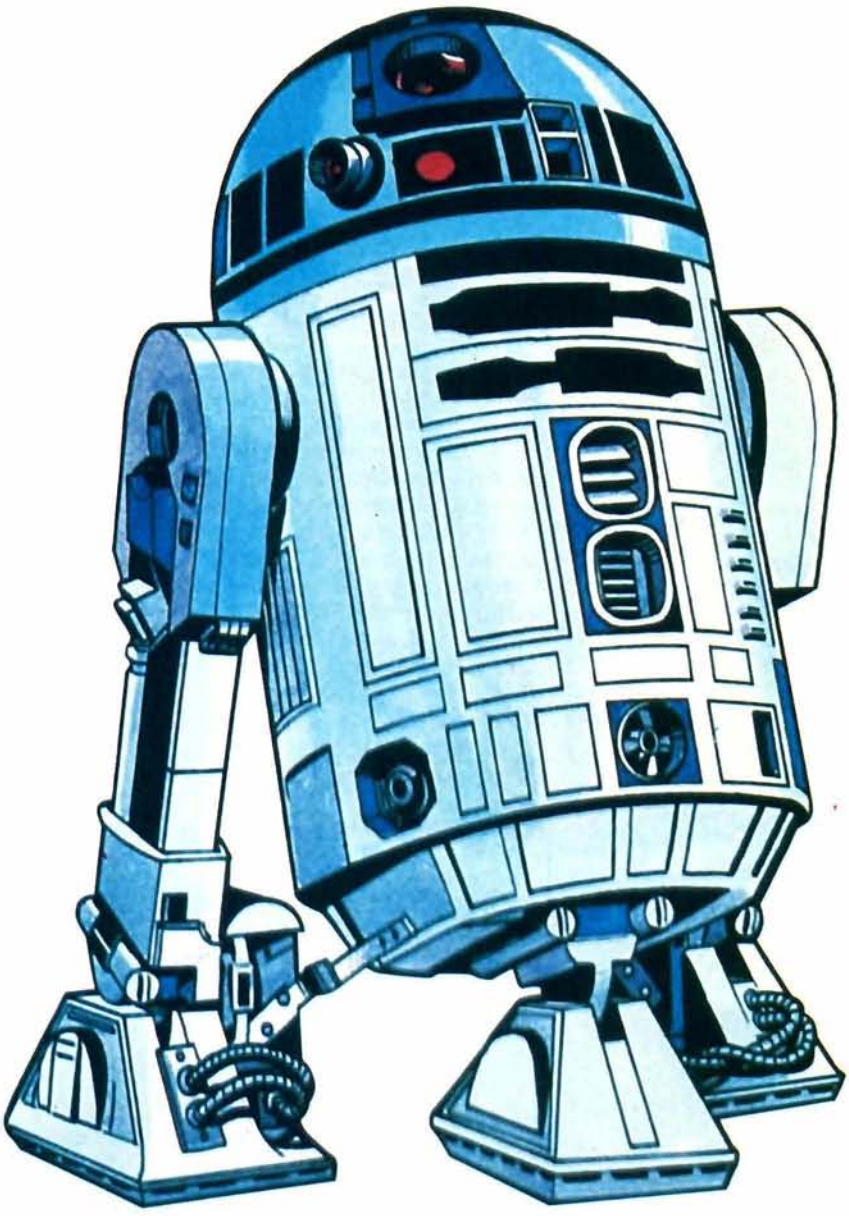
I primi robot sono comparsi nelle officine meccaniche degli anni '50, dove per la prima volta fu applicata la teoria del controllo numerico delle macchine utensili. Gli apparecchi sperimentali di allora non erano molto sofisticati: il funzionamento era controllato mediante un nastro di carta a 5 fori (simile a quello dei telex), e, tutt'al più, essi erano in grado di spostare un utensile fisso da un punto all'altro del pezzo su cui stavano lavorando.

Successivamente, fu trovato il modo per cui i robot potessero cambiare utensile durante il lavoro e ciò si ottenne usando una "giostra" o porta utensili ruotante. Su di essa venivano fissati, automaticamente, i vari utensili scelti sotto il controllo di un programma.

Nonostante queste migliorie, una macchina poteva svolgere un solo tipo di lavoro: un tornio era sempre un tornio, anche se poteva eseguire in successione tutte le operazioni necessarie per uno specifico procedimento.

Circa nello stesso periodo vennero impiegate le prime braccia e mani meccaniche controllate a distanza, per lavori in ambienti pericolosi: ad esempio, sott'acqua o in laboratori dove venivano trattati materiali radioattivi. Questi strumenti di manipolazione erano soltanto estensioni meccaniche delle mani stesse dell'operatore, ma ben presto furono usati dei computer per controllarli direttamente.

Per comprendere come vengano programmati i robot, occorre innanzi tutto esaminare l'ambiente nel quale essi operano. Molti robot industriali sono fissi, per cui il loro raggio d'azione è limitato ad una sfera appiattita alla base: possiamo quindi affrontare il problema del controllo del robot come un semplice esercizio di geometria tridimensionale. Al centro dello sferoide si troverà la "spalla" del robot ed il raggio sarà uguale alla lunghezza del braccio esteso, misurato dalla spalla fino alla punta delle "dita", cioè al dispositivo di presa o porta utensili. Ogni punto all'interno di questo spazio viene individuato da 3 coordinate, rispetto ad un punto di riferimento, detto anche "origine".



L'eroe di celluloide
R2D2, il simpatico robot del film "Guerre Stellari", era in realtà guidato da un uomo. Il suo aspetto, tuttavia, corrisponde all'idea che molte persone hanno dei robot

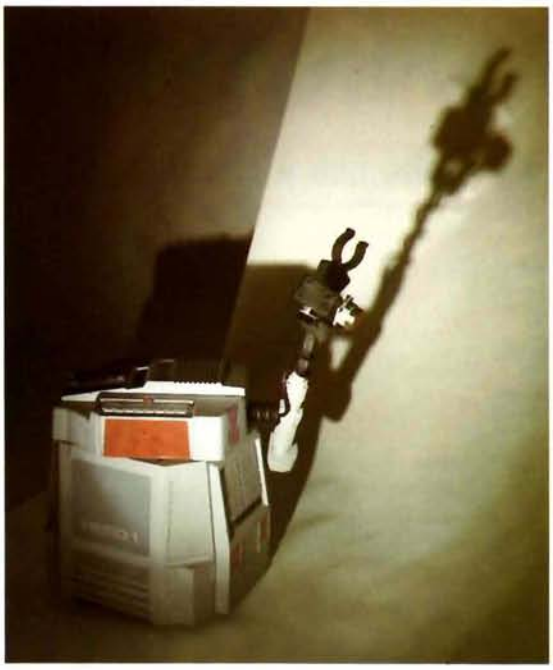
In altre parole, la posizione di un dato punto può essere così definita: "ad una distanza di 2 m. in direzione nord/est e 30 gradi sopra l'orizzonte". In questo caso, le misure s'intendono riferite alla spalla del robot, che viene considerata l'origine del sistema.

Tuttavia, la programmazione di un robot implica anche l'impiego di una serie d'istruzioni relative al movimento da un punto a un altro: esiste dunque un terzo metodo per posizionare il porta utensili. Noto come posizionamento "da punto a punto", esso richiede

Robot autosufficienti

L'Hero-1 è un robot, azionato da batterie interne, che somma alcune caratteristiche delle "tartarughe" all'abilità di manipolare oggetti servendosi di un braccio meccanico. Anche se a prima vista può sembrare un giocattolo, in realtà questo robot è basato su di un computer piuttosto sofisticato, in grado di controllare un sintetizzatore vocale, più sensori di luce, un sensore acustico e, poiché è mobile, anche un dispositivo di posizionamento ad ultrasuoni. Negli Stati Uniti è venduto anche in scatola di montaggio ed il prezzo si aggira sui cinque milioni di lire

IAN McKINNELL per concessione della COMPUTING PLUS



che l'origine (punto 0) si sposti assieme al porta utensili.

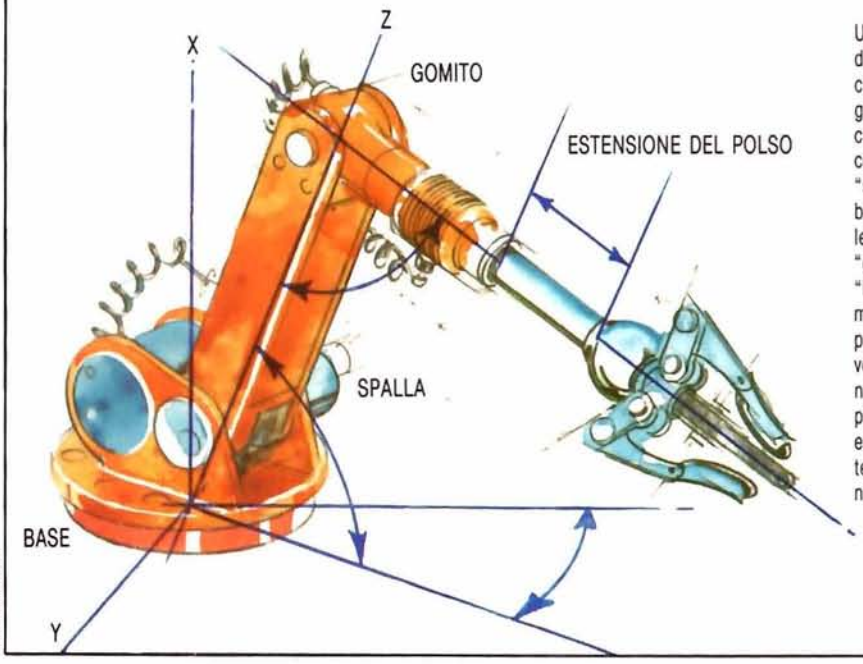
I robot industriali hanno, solitamente, un margine d'errore inferiore al millimetro, ma persino i modelli più semplici ed economici (utilizzabili con qualsiasi home computer che abbia un'uscita parallela ad 8 bit) presentano un margine d'errore inferiore ai 2 millimetri. I metodi impiegati per ottenere l'azionamento dei bracci di un robot sono, principalmente, due. Per quelli con una bassa capacità di carico utile sono sufficienti dei motori "passo passo" (motori elettrici che compiono rotazioni prefissate ad ogni impulso di corrente applicata). Ma quando i robot devono manovrare carichi molto pesanti, è più comune l'uso di pistoni idraulici per muovere le varie parti del braccio intorno al proprio fulcro. Poiché lo spostamento del pistone (e quindi del braccio) è in diretto rapporto con il volume del liquido impiegato in esso, è piuttosto facile ottenere grande precisione nei movimenti.

I robot industriali vengono invariabilmente collegati ad un microcomputer "dedicato" (o, in modelli più recenti, ad un microcomputer di grande capacità) che controlla il braccio ed usa un linguaggio di programmazione progettato specificatamente. Il microcomputer provvede ad indicare le coordinate di spostamento e ad inviare semplici comandi come, ad esempio, APRI LA MORSA o CHIUDI LA MORSA, ed il linguaggio di programmazione non possiede istruzioni per la trattazione di testi. Le istruzioni di programma vengono immesse usando una tastiera numerica, collegata al computer da un lungo "cordone ombelicale": l'operatore può così manovrare il braccio del robot, restando a distanza di sicurezza. Le versioni più moderne di queste tastiere sono corredate di joystick di precisione.

Un altro metodo di programmazione, conosciuto come "Follow Me" (Seguimi), è molto usato in operazioni (come la verniciatura a spruzzo) che non richiedono un posizionamento dell'utensile particolarmente accurato. In questo caso, il braccio del robot prevede la possibilità che l'operatore manovri manualmente il porta utensili intorno al pezzo da verniciare: i movimenti vengono automaticamente registrati nella memoria del computer ed, in seguito, il robot ripeterà da sé le stesse operazioni.

In tutti i metodi finora esaminati, la posizione da definire è quella del porta utensili stesso. L'operatore non deve preoccuparsi delle posi-

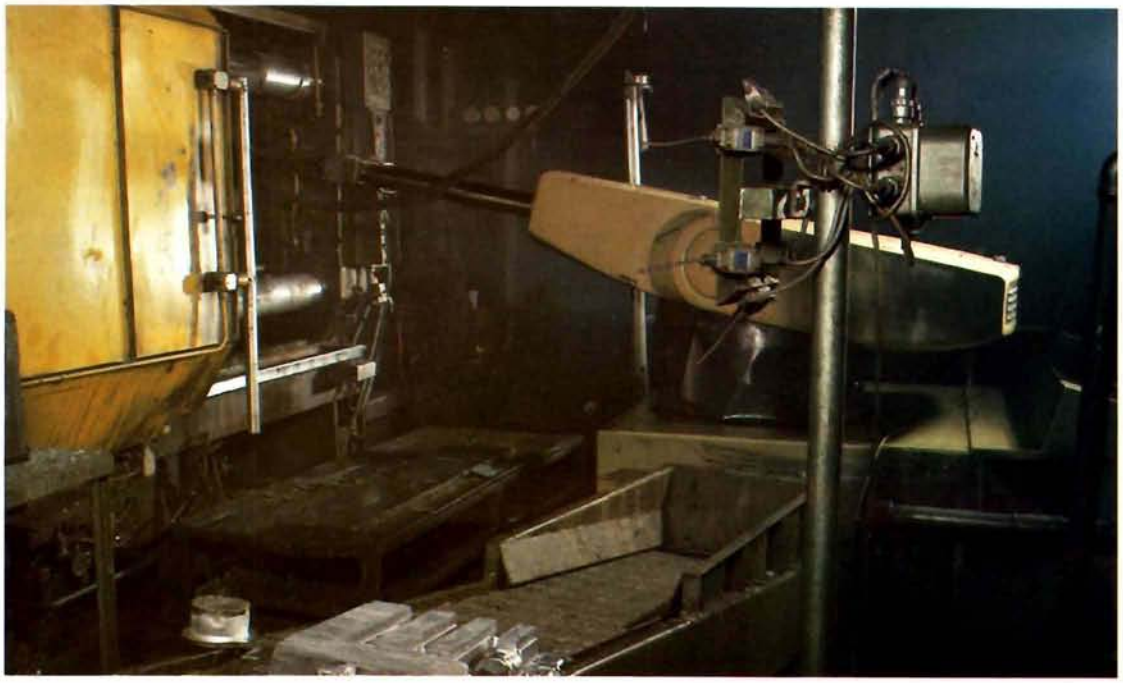
IL BRACCIO DEL ROBOT



BOB FREEMAN

Uno degli aspetti più complessi della programmazione di un braccio per robot è la conversione geometrica. Noi siamo soliti specificare una posizione usando le coordinate cartesiane x, y, z. Le "specifiche" di cui un robot ha bisogno sono: l'angolazione per le giunture della "spalla" e del "gomito", la rotazione della "base", la misura dell'allungamento del "polso". Nei sistemi più semplici i programmatori devono inserire i valori per ciascuna "specifica", mentre, i robot più sofisticati, sono capaci di eseguire tutte le conversioni partendo dalle coordinate cartesiane

zioni relative di ogni singola parte del robot: è il programma memorizzato nel computer di controllo del robot che provvede a ciò. Inoltre, è sempre il programma a calcolare quali siano i percorsi migliori, garantendo che l'utensile si muova da un punto all'altro seguendo la via più breve. L'orientazione del porta utensili è controllata automaticamente e conserva le coordinate orizzontali e verticali, salvo diversa istruzione. Anche la velocità di spostamento da punto a punto è automatica: in genere, è previsto che il porta utensili si allontani lentamente dal pezzo da lavorare, si sposti rapidamente in una nuova posizione di lavoro, e, quindi, rallenti di nuovo prima di iniziare la nuova fase di lavorazione.



I robot al lavoro

Sempre più spesso, i bracci robotizzati, come quello qui illustrato, vengono impiegati per i lavori più pericolosi, sporchi e ripetitivi presenti nell'industria. La pulitura dei pezzi fusi, che precede la loro lavorazione, ne è un buon esempio. Il pezzo fuso, "fresco" di stampo, è troppo caldo per le mani di un uomo e dovrebbe quindi essere messo da parte a raffreddarsi. Il robot, essendo assai meno sensibile al calore, può lavorarlo immediatamente ed inviarlo alla successiva fase di produzione

I robot di cui abbiamo sinora parlato sono capaci solo di "cieca obbedienza" e ripetono la stessa operazione in modo esatto e nello stesso punto, indipendentemente da interferenze esterne. Essi vengono impiegati, soprattutto, nell'industria automobilistica, dove, già da tempo, la produzione è organizzata in catene di montaggio, nelle quali la singola parte (o il veicolo parzialmente ultimato) vengono posizionati con precisione nello spazio e nel tempo. Ciò è di estrema importanza per un perfetto funzionamento del processo di produzione mediante robot: infatti, se il componente viene posizionato in modo errato, il robot non è capace di adattarsi alla nuova situazione. Nel tentativo di eliminare questo problema sono stati impiegati vari tipi di sensori, da semplici interruttori a scatto, a sofisticate telecamere capaci di riconoscere le varie parti da lavorare.

Un'alternativa al sensore di contatto può essere un sistema a fotocellula: una sorgente luminosa viene collocata di fronte ad un sensore di luce posto sul porta utensili. Se il pezzo su cui si deve operare non è posizionato in modo corretto, ciò impedirà alla luce di colpire il sensore, facendo entrare il robot in uno stato di "attesa" o di "allarme". Riportato il pezzo in posizione corretta, il lavoro potrà regolarmente riprendere. Naturalmente anche questa procedura non esclude possibilità d'errore. Per operazioni in cui è necessaria una totale affidabilità, è però possibile istallare un sistema a ricognizione visiva basato su telecamere a sorveglianza accoppiata (CCD). Queste telecamere focalizzano l'immagine direttamente su uno speciale microchip (un chip suddiviso in 100 o più sensori individuali) ciascuno capace di registrare non solo il bianco e il nero ma anche una gamma di toni intermedi. Ciascun sensore individuale necessita di un byte di memoria per definire il contrasto nella scala dei grigi. Inizialmente, ciascun componente della lavorazione viene "fotografato" più volte nella memoria del computer ed uno speciale programma elabora le informazioni ottenute. Durante la fase operativa, la telecamera CCD rileva un'immagine dell'oggetto, confrontandola con quella presente in memoria. Se le due immagini coincidono, la lavorazione può andare avanti. Con questo metodo è possibile verificare la presenza del pezzo giusto e la correttezza della sua posizione.

Un ulteriore uso di questo sistema d'analisi per immagini è la selezione di componenti da utilizzare in un particolare assemblaggio. Oltre che nelle catene di montaggio, i robot industriali vengono comunemente usati per la verifica dei prodotti finiti e per controlli di qualità.

Abbiamo iniziato col parlare dei robot nella fantascienza: essi sono un ottimo esempio di come la realtà tenda a superare la fantasia e non c'è motivo per cui i robot, in futuro, non debbano divenire quelle entità autonome ed automotivate, ipotizzate nei romanzi fantascientifici. Tuttavia ciò non avverrà finché l'Intelligenza Artificiale resterà solo un'ipotesi teorica.

I suoni del Vic

La generazione di effetti musicali sul Vic

Il Vic-20 è stato uno dei primi home computer ad apparire sul mercato. Di conseguenza, le sue capacità possono apparire in parte superate se confrontate con quelle di alcuni apparecchi più recenti. Inoltre, la creazione di effetti musicali non è particolarmente facile, né sul Vic-20, né sul Commodore, per la mancanza di specifiche istruzioni BASIC. Il controllo del suono viene infatti realizzato con una serie di POKE, per depositare opportuni valori nelle locazioni di memoria. Questo sistema vale per ambedue i computer della Commodore.
È possibile intervenire sul volume del suono prodotto (equivalente ad un inviluppo dove A=D=R=0), sulla frequenza di tre oscillatori e su un generatore di rumori. I suoni vengono riprodotti direttamente sull'altoparlante del televisore adoperato come monitor. Purtroppo, non è sempre possibile selezionare con precisione le frequenze (e quindi ottenere un timbro corretto per tutte le note della scala musicale), ma con un po' di riflessione, pazienza ed una relativa conoscenza della programmazione BASIC, queste limitate capacità possono essere sfruttate al meglio per creare "linee melodiche" con accordi di due o tre note.

Controllo del suono

Il Vic-20 è fornito di tre oscillatori ad onda quadra e di un generatore di rumori. Ciascun oscillatore copre approssimativamente tre ottave, sviluppate in frequenza nel modo seguente:

Osc. 1	Osc. 2	Osc. 3	Gamma di frequenze (Hz)	Ottava
•			65,41 - 123,47	1
•	•		130,81 - 246,94	2
•	•	•	261,63 - 493,88	3
	•	•	523,25 - 987,77	4
		•	1046,5 - 1975,53	5

Questa combinazione permette di coprire complessivamente cinque ottave con almeno un oscillatore disponibile in ciascuna ottava. L'ottava 3, che parte dal "do centrale" e contiene il "la" di riferimento standard a 440Hz, è ottenibile su tutti e tre gli oscillatori; il controllo di questi si effettua cambiando così il contenuto delle cinque locazioni di memoria:

Locazioni di memoria	Oscillatore
POKE 36874,X	1
POKE 36875,X	2
POKE 36876,X	3
POKE 36877,X	rumore

In ciascun caso X è un numero intero compreso tra 135 e 241 (0 spegne l'oscillatore) che si riferisce ad una tabella di valori, equivalenti alle note, contenuta nel manuale che correda ogni Vic-20. Prima che la frequenza selezionata possa essere udita, il livello del volume deve essere selezionato nel seguente modo:

```
POKE 36878,V
```

dove V può avere un valore compreso tra 0 (spento) e 15 (volume massimo) ed influenza tutti gli oscillatori ed il rumore. Per esempio:

```
POKE 36874,219:POKE 36875,19:POKE 36876,219:POKE 36878,7
```

producono il "la" di riferimento a 440Hz sull'oscillatore 1, un "la" di un'ottava più alto sull'oscillatore 2 e un "la" ancora più alto di un'ottava sull'oscillatore 3, tutti ad un volume medio di 7. Per "spegnere" i generatori, occorre azzerare ciascuna locazione con altrettante POKE.

Note e pause

Se non definiamo un tempo di durata per ciascuna nota e corrette pause tra note successive, esse si sovrappongono. Per facilitare l'introduzione di queste pause, è possibile utilizzare due distinti metodi che consentono al computer di "marcare il tempo" fra le istruzioni POKE. Il primo utilizza dei cicli FOR-NEXT:

```
10 POKE 36878,7
20 POKE 36876,203
30 FOR P = 1 TO 200
40 NEXT P
50 POKE 36878,0
60 POKE 36876,0
```

Questi comandi suonano la nota re# per la durata dei 200 "passaggi", stabiliti nel ciclo FOR-NEXT. Tuttavia, la precisione di questo metodo dipende da una definizione esterna molto accurata. Un metodo più semplice per determinare la lunghezza delle note si basa sull'impiego dell'orologio incorporato nel Vic-20 che "conta" in sessantesimi di secondo (chiamati "jiffy") ed al quale ci si può riferire usando la variabile TI. Questo procedimento è molto utile: è infatti possibile costruire un comando d'attesa per una pausa ben definita. Ad esempio, le istruzioni:

```
10 POKE 36878,7
20 POKE 36876,203:D=TI
30 IF TI-D<15 THEN 30
40 POKE 36878,0
50 POKE 36876,0
```

servono per suonare la stessa nota di prima, ma per una durata di 15 jiffy, ossia un quarto di secondo. Alla variabile D viene assegnato il valore di TI quando inizia il suono. La linea 30 conta 15 jiffy prima di cedere il comando alla linea 40. Le "linee melodiche" possono essere costruite usando lo stesso sistema, adottato per le pause. Più avanti, vedremo come costruire melodie col Vic-20.

... le luci del Dragon

... e le capacità grafiche del Dragon

Il computer Dragon 32 utilizza un particolare dialetto BASIC, noto come "Microsoft Extended Colour Basic".
Numerosi altri computer oggi sul mercato usano questa versione di BASIC: è facile da usare ed ha una buona serie di comandi per tracciare linee, cerchi ed altre figure geometriche, che possono essere facilmente colorate, utilizzando un semplice programma, per creare, sullo schermo, immagini più vive e stimolanti.
Il Dragon 32 possiede sette livelli di risoluzione, consentendo all'utente di lavorare su di uno schermo suddiviso, al livello più alto, in 49152 singoli punti. Sono disponibili otto colori, ma la scelta si riduce a quattro o a due, quando si lavora in alta risoluzione.

I livelli di risoluzione

Il normale schermo di 16 righe per 32 colonne di caratteri costituisce il livello di risoluzione più basso ed il comando PRINT@ permette di posizionare un carattere su una qualunque delle 512 locazioni del video. Oltre alla serie di caratteri normali, sono disponibili anche 16 caratteri grafici a bassa risoluzione, in otto colori.
Il livello superiore di risoluzione suddivide lo schermo in 32 righe e 64 colonne. La dimensione di ciascun quadratino, in questo caso, è un quarto di quella di un normale carattere. Punti di queste dimensioni possono essere tracciati sullo schermo con il comando SET e cancellati con il comando RESET.
Entrambi i livelli sopra descritti possono essere utilizzati contemporaneamente. Sono disponibili altri cinque livelli, ad alta risoluzione, ma questi non possono essere impiegati simultaneamente. I cinque modi di funzionamento ad alta risoluzione sono selezionabili mediante il comando PMODE:

PMODE	Risoluzione	Colori disponibili
0	128×96	2
1	128×96	4
2	128×192	2
3	128×192	4
4	256×192	2

Naturalmente, è necessario un compromesso tra grado di risoluzione, numero di colori e quantità di memoria; di ciò bisogna tener conto quando si scrivono lunghi programmi BASIC, che utilizzano anche visualizzazioni ad alta risoluzione nella quale è disponibile solo un numero limitato di colori, ma col Dragon 32 esiste la possibilità di scegliere tra due gamme di colori, usando il comando SCREEN. Per esempio, SCREEN 1,0 seleziona uno schermo ad alta risoluzione e la gamma di colori 0. SCREEN 1,1 uno schermo ad alta risoluzione che utilizza la gamma di colori 1.

PAINT
Questo comando è utilissimo, nella creazione d'immagini grafiche, per colorare, sullo schermo, aree delimitate. Si possono facilmente colorare cerchi, triangoli o qualsiasi altra figura a perimetro chiuso.

DRAW
Imita sullo schermo il movimento di una matita e permette di tracciare linee in una qualsiasi delle quattro direzioni. Il comando DRAW permette anche di far ruotare o d'ingrandire un'intera figura.

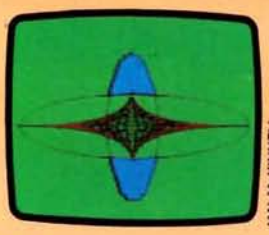
GET e PUT
Il comando GET ordina al computer di archiviare in memoria un'immagine presente sullo schermo, mentre PUT permette di richiamare sullo schermo un'immagine precedentemente archiviata.

PSET e PRESET
Questi comandi sono gli equivalenti, in alta risoluzione, dei comandi SET e RESET già trattati: essi accendono o spengono un particolare punto dello schermo. È anche possibile definire il colore del punto stesso.

LINE
Questo comando permette di congiungere due punti con una linea retta ad alta risoluzione.

CIRCLE
Permette di tracciare circonferenze ad alta risoluzione fornendone il centro ed il raggio. È anche possibile tracciare archi di cerchio o comprimere una forma circolare per ottenere un'ellisse.

Il Dragon 32 è un computer economico, ma con molti utili comandi per la grafica, anche se, in definitiva, è più adatto ad immagini statiche che non a quelle in rapido movimento.
I comandi disponibili, in alta risoluzione, fanno del Dragon 32 uno strumento ideale per i giovani che sono dotati di una fantasia prorompente.
Il maggior inconveniente è l'impossibilità di visualizzare contemporaneamente sullo schermo testi ed immagini grafiche ad alta risoluzione.



IAN McKINNELL

Colori a piacere!
Questa immagine è un tipico esempio di ciò che si può ottenere su un Dragon grazie ai comandi disponibili in alta risoluzione

Alta risoluzione
Questo è un breve programma per dimostrare alcune delle capacità grafiche del Dragon 32, in alta risoluzione. È stato impiegato il PMODE 3, che, pur non essendo il livello più alto di risoluzione, permette una discreta gamma di colori

```
10 PCLS:PMODE3,1
20 SCREEN1,0
30 COLOR 0, 1
40 FOR X = 0 TO 127 STEP 10
50 LINE(X,85)—(127,85—X/3),PSET
60 LINE(X,85)—(127,85 + X/3),PSET
70 LINE(255—X,85)—(127,85—X/3),
   PSET
80 LINE(255—X,85)—(127,85 + X/3),
   PSET
90 NEXT X
100 CIRCLE(127,85),128,4,0.3
110 CIRCLE(127,85),30,4,3
120 PAINT(130,30)3,4
130 PAINT(130,130),3,4
140 GOTO 140
150 END
```

I vari tipi di "sort"

Una delle operazioni più necessarie, in un sistema computerizzato, è il riordinamento dei dati

Il riordinamento di dati è una delle applicazioni più diffuse sui computer, ma è un lavoro per il quale essi sono, nel loro standard abituale, altamente inefficienti. Secondo alcune stime operative, il 30 o 40 per 100 di tutto il tempo di elaborazione, viene speso riordinando dati. Se si sommano le relative operazioni di fusione ("merge") e di ricerca, allora la percentuale supera probabilmente il 50 per 100.

I programmatori dedicano moltissimo del loro tempo ad inventare nuovi algoritmi di riordinamento, sempre più rapidi ed efficienti. È molto difficile analizzare i metodi più sofisticati di riordino, ma le procedure più semplici si possono illustrare impiegando un mazzo di carte da gioco.

Prendiamo 13 carte dello stesso seme, disponiamole sul tavolo, una accanto all'altra, ma in ordine sparso, evitando di collocare l'Asso ed il Due all'estremità destra della linea. Decidiamo di sistemare le carte in ordine discendente (Re, Donna, Jack,... Asso) a partire da sinistra. Questa è un'operazione che noi compiamo quasi istintivamente e ci impegna così poco, che è difficile descriverne esattamente l'esecuzione. Se, invece, fossimo costretti a muovere una sola carta alla volta, a non sovrapporre le carte tra loro ed a disporle in modo da occupare il minor spazio possibile, allora questa operazione diverrebbe molto meno semplice e sarebbe difficile stabilire un modo di procedere veramente efficiente. In questa analogia, le carte rappresentano parti di informazioni, la massima superficie coperta corrisponde alla quantità di memoria necessaria al computer per compiere l'operazione, il ragionamento che noi facciamo è il programma. Come risolvere dunque il problema?

1) Poniamo al di sotto della carta più a sinistra una moneta, usandola come marcatore di posizione che ci ricordi dove siamo arrivati nel riordinamento. Confrontiamo la carta "marcata" con quella alla sua destra: sono in ordine decrescente? Se non lo sono, scambiamo le loro posizioni lasciando la moneta dove si trova, seguendo la regola di muovere una sola carta alla volta e di non sovrapporre le carte tra loro.

2) Quando le due carte sono in ordine decrescente, si sposti la moneta a destra di una posizione e si ripeta la fase 1. Ci troviamo adesso in un ciclo, che terminerà soltanto quando sposteremo la moneta nell'ultima posizione a destra. Avremo fatto, a quel punto, un "passaggio" completo delle carte allineate sul tavolo.

3) Alla fine del primo passaggio, osserviamo le carte. L'Asso, che è la carta più bassa del seme, ha trovato la sua strada per arrivare all'estremità destra della sequenza, quindi è in posizione corretta. Se effettuiamo un secondo passaggio delle carte, come descritto nelle fasi 1 e 2, al termine il Due verrà a trovarsi in posizione corretta. Ripetiamo questa operazione, passaggio dopo passaggio, finché tutta la sequenza sarà ordinata in modo decrescente.

La procedura sopra descritta presenta diversi difetti. È noiosa, non è economica, dal momento che il semplice scambio di posizione tra due carte richiede tre diverse operazioni e, soprattutto, molti dei confronti tra le carte non sono necessari. Per esempio, dopo il primo passaggio troviamo l'Asso in posizione corretta, per cui non c'è motivo di spostare la moneta in posizione 13 (dove, comunque, non è possibile alcun confronto). Nel secondo passaggio, l'ultima carta a destra nella sequenza è già in posizione corretta e non c'è alcun bisogno di spostare la moneta in posizione 12. In generale, ogni passaggio termina una posizione prima rispetto al passaggio precedente.

Un altro problema è sapere quando fermarsi: un computer continuerà a confrontare carte indefinitamente, finché non gli si ordinerà di fermarsi. L'unica regola sicura consiste nel terminare il riordinamento dopo un passaggio nel quale non si sia verificato alcuno scambio di carte. In altre parole: se abbiamo esaminato la sequenza di carte senza alterarne l'ordine, ciò significa che esse si trovano nell'ordine voluto.

Questo metodo di riordinamento è chiamato "Bubble Sort" e presenta i seguenti vantaggi: è semplice convertirlo in un programma, impiega poca memoria ed ha una discreta efficienza con piccole quantità di dati parzialmente ordinati. Questi sono i criteri secondo i quali bisogna giudicare un algoritmo di riordinamento anche se, quando la quantità di dati da riordinare è molto grande, la velocità d'esecuzione può essere sacrificata a vantaggio dell'economia di memoria. Infatti la memoria di un computer può non essere in grado di contenere sia i dati "disordinati" che una loro copia riordinata. Per questa ragione, non prenderemo in considerazione gli algoritmi che utilizzano forme di duplicazione dei dati.

Un secondo semplice metodo di riordinamento, anch'esso rapportabile ad un metodo manuale, è il seguente:

1) Sistemiamo sul tavolo le carte dopo averle

Il "Bubble sort"

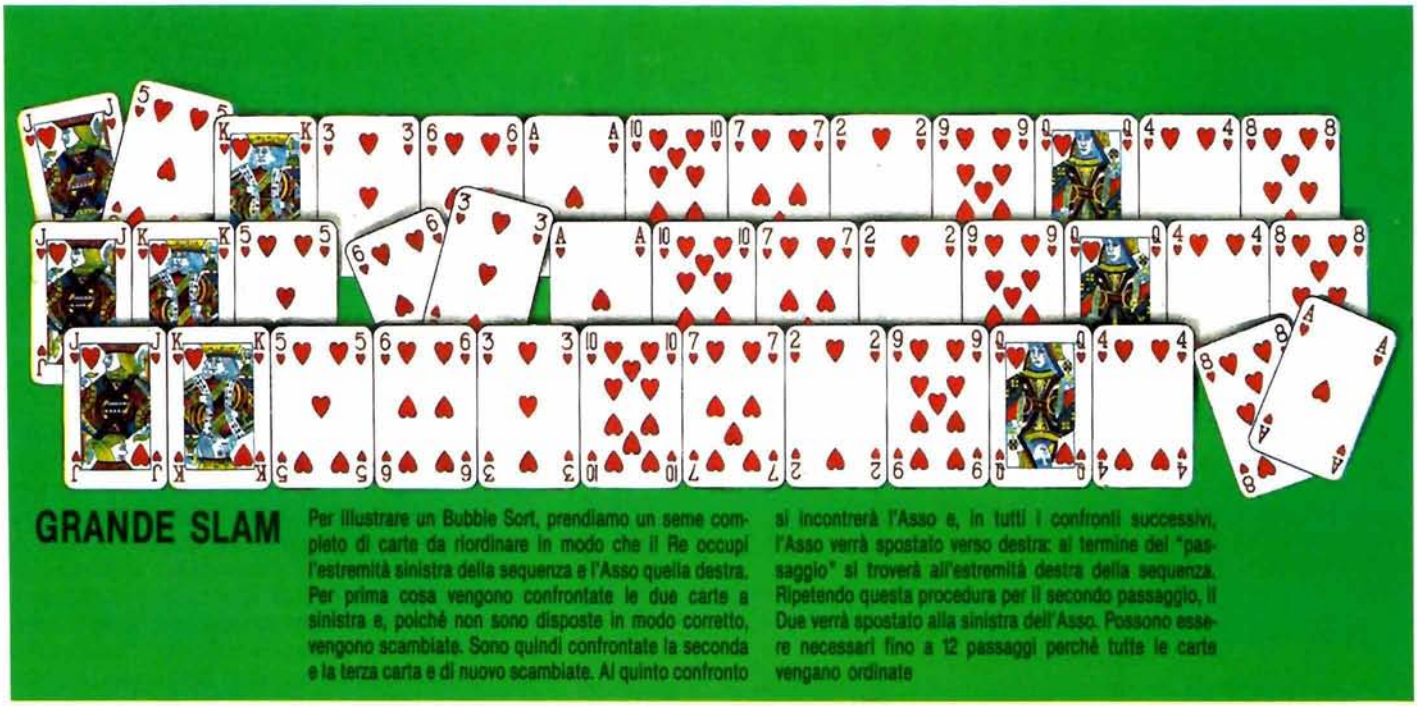
Questo schema illustra il metodo chiamato "Bubble Sort" (letteralmente: "riordino a bolle"), per una mano ridotta di 9 carte (D è la carta Dieci). La zona ordinata della mano cresce, a partire dall'estremità destra, dopo ogni "passaggio". I numeri 1 e 2 sotto la mano di carte indicano le due carte che vengono confrontate in quel momento.

```
Inizio Sort
2 8 9 3 D 5 K 6 7  Inizio passaggio 1
1 2
8 2 9 3 D 5 K 6 7
  1 2
8 9 2 3 D 5 K 6 7
    1 2
8 9 3 2 D 5 K 6 7
      1 2
8 9 3 D 2 5 K 6 7
        1 2
8 9 3 D 5 2 K 6 7
          1 2
8 9 3 D 5 K 2 6 7
            1 2
8 9 3 D 5 K 6 2 7
              1 2
8 9 3 D 5 K 6 7 2  Fine passaggio 1
9 8 D 5 K 6 7 3 2  Fine passaggio 2
9 D 8 K 6 7 5 3 2  Fine passaggio 3
D 9 K 8 7 6 5 3 2  Fine passaggio 4
D K 9 8 7 6 5 3 2  Fine passaggio 5
K D 9 8 7 6 5 3 2  Fine passaggio 6
Fine Sort
```

Riordinamento per inserzione

Con questo metodo, la zona ordinata della mano cresce a partire dall'estremità sinistra. Le carte vengono inserite direttamente nella loro posizione corretta via via che vengono esaminate.

```
Inizio Sort
2 8 9 3 D 5 K 6 7
2 1
8 2 9 3 D 5 K 6 7
2   1
9 8 2 3 D 5 K 6 7
    2 1
9 8 3 2 D 5 K 6 7
2     1
D 9 8 3 2 5 K 6 7
      2   1
D 9 8 5 3 2 K 6 7
2         1
K D 9 8 5 3 2 6 7
          2     1
K D 9 8 6 5 3 2 7
          2       1
K D 9 8 7 6 5 3 2
Fine Sort
```



GRANDE SLAM Per illustrare un Bubble Sort, prendiamo un seme completo di carte da riordinare in modo che il Re occupi l'estremità sinistra della sequenza e l'Asso quella destra. Per prima cosa vengono confrontate le due carte a sinistra e, poichè non sono disposte in modo corretto, vengono scambiate. Sono quindi confrontate la seconda e la terza carta e di nuovo scambiate. Al quinto confronto si incontrerà l'Asso e, in tutti i confronti successivi, l'Asso verrà spostato verso destra: al termine del "passaggio" si troverà all'estremità destra della sequenza. Ripetendo questa procedura per il secondo passaggio, il Due verrà spostato alla sinistra dell'Asso. Possono essere necessari fino a 12 passaggi perché tutte le carte vengano ordinate

TONY LODGE

di nuovo mescolate e poniamo una moneta da 50 lire sotto la seconda carta da sinistra. Qualunque sia la carta così "marcata" all'inizio di ciascun passaggio, noi la chiameremo carta "50".

2) Estraiamo dalla sequenza la carta 50 (lasciamo quindi uno spazio vuoto) e piazziamo una moneta da 100 lire sotto la carta a sinistra dello spazio vuoto. Chiameremo questa carta, carta "100".

3) Confrontiamo ora la carta 50 con quella 100. Se sono in ordine, rimettiamo la carta 50 al suo posto e passiamo la carta 100 nello spazio rimasto vuoto (creandone in questo modo un altro in posizione diversa) e spostiamo la moneta da 100 lire a sinistra di una posizione. Marcheremo così una nuova carta 100 (se però la precedente carta 100 si trovava all'estremità sinistra della sequenza, allora lo spostamento della moneta non verrà effettuato; inseriremo quindi la carta 50 nell'unico spazio vuoto presente e passeremo alla fase 4).

Confrontiamo ora la nuova carta 100 con la 50 (quella spostata) e ripetiamo la fase tre, fino a trovare la giusta posizione per la carta 50.

4) Spostiamo le 50 lire a destra di una posizione e ripetiamo le fasi 2 e 3. Quando non potremo più muovere le 50 lire verso destra, le carte saranno tutte ordinate in modo decrescente.

Questo procedimento è chiamato "Riordinamento per inserzione" ed è molto simile al modo usato da un giocatore per ordinare una "mano" di carte. Anche se risulta un po' più difficile da programmare rispetto ad un Bubble Sort, è certamente assai più efficiente. Vedremo in una fase successiva alcuni algoritmi di riordinamento più complessi.

Riordinamento ad alta velocità

Questo programma BASIC illustra il diverso grado d'efficienza tra un "Bubble Sort" ed un riordinamento "per inserzione". Il programma è stato scritto per la massima velocità di esecuzione, per cui non sono documentate le operazioni svolte dalle "routines" (la presenza di istruzioni REM e relative frasi a commento, infatti, rallenta le operazioni). Questo listato dovrebbe girare sulla maggior parte dei computer, ma sarà bene rivedere le istruzioni ON...GOSUB e RANDOMIZE, rispettivamente a pagina 215 e 175

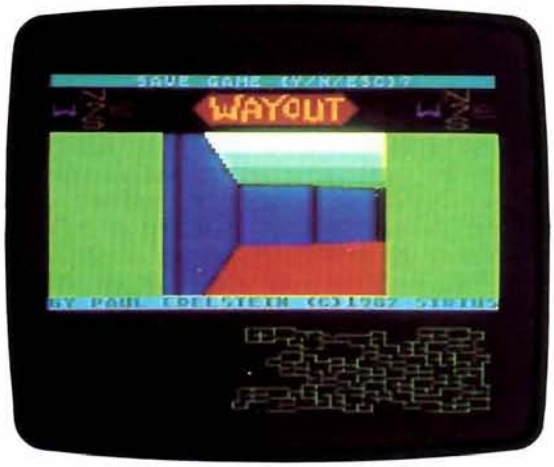
```
9   REM*************************************************
10  REM*      ALGORITMI PER RIORDINAMENTO DATI          *
11  REM*************************************************
100 INPUT "QUANTI ELEMENTI DA RIORDINARE";LT
150 IF LT < 3 THEN LET LT = 3
200 LET LT = INT(LT)
250 DIM R(LT), C(LT)
300 LET Z = 0 : LET Q = 0 : LET P = 0
350 LET I = 1 : LET O = 0 : LET II = 2 : LET TH = 2
400 INPUT "QUANTE PROVE";N
450 FOR CT = I TO N
500 GOSUB 4000
550 FOR SR = I TO TH
600 GOSUB 5000
650 PRINT:PRINT:PRINT:PRINT
700 PRINT "TEST N.";CT + SR/10
750 INPUT "BATTERE RETURN PER INIZIARE IL SORT";A$
800 PRINT "L'ELENCO NON ORDINATO E'"
850 GOSUB 3000
900 ON SR GOSUB 6000,7000
950 PRINT "L'ELENCO ORDINATO E'"
1000 GOSUB 3000
1050 NEXT SR
1100 NEXT CT
1150 END
2999 REM*********************************
3000 REM*   VISUALIZZA L'ELENCO           *
3001 REM*********************************
3100 FOR K = I TO LT
3200 PRINT R(K);
3300 NEXT K
3400 PRINT
3500 RETURN
3999 REM*********************************
4000 REM*      GENERATORE CASUALE         *
4001 REM*********************************
4100 REM INSERIRE QUI 'RANDOMIZE' SE NECESSARIA
4200 FOR K = I TO LT
4300 LET C(K) = INT(100*RND(1))
4400 NEXT K
4500 RETURN
4999 REM*********************************
5000 REM* RIGENERATORE DI VALORI CASUALI*
5001 REM*********************************
5100 FOR K = I TO LT
5200 LET R(K) = C(K)
5300 NEXT K
5400 PRINT : PRINT
5500 RETURN
5999 REM*********************************
6000 REM*      BUBBLE SORT              *
6001 REM*********************************
6050 PRINT "BUBBLE SORT - INIZIATO !"
6100 FOR P = LT - I TO I STEP -I
6150 LET F=-I
6200 FOR Q = I TO P
6250 LET Z = Q + I
6300 IF R(Q) < R(Z) THEN LET D = R(Q) :
     LET R(Q) = R(Z) : LET R(Z) = D : LET F = O
6350 NEXT Q
6400 IF F =-I THEN LET P=I
6450 NEXT P
6500 PRINT "BUBBLE SORT - TERMINATO !"
6550 RETURN
6999 REM*********************************
7000 REM*  SORT PER INSERIMENTO          *
7001 REM*********************************
7050 PRINT "SORT PER INSERIMENTO - INIZIATO !"
7100 FOR P = II TO LT
7200 LET D = R(P)
7300 FOR Q = P TO II STEP -I
7400 LET R(Q) = R(Q-I)
7500 IF D <= R(Q) THEN LET R(Q) = D: LET Q = II
7600 NEXT Q
7700 IF D > R(I) THEN LET R(I) = D
7800 NEXT P
7850 PRINT "SORT PER INSERIMENTO - TERMINATO !"
7900 RETURN
```

Quattro bit... per un labirinto!

Vediamo come si progettano i labirinti, elementi essenziali di tanti giochi per computer



"A sirene spiegate"
Questo gioco (per il Commodore 64) è uno sviluppo del tradizionale gioco di "veduta aerea". Una macchina della polizia pattuglia una città, completa di strade ed edifici



Il "Way Out"
È possibile ottenere sullo Spectrum una realistica immagine tridimensionale con il gioco "Way Out". Muovendo, anche di poco, il joystick si produrrà un cambiamento nell'angolazione della visuale



IAN McKINNELL

"Cerchio di tenebre"
Anche se questo gioco (per il Dragon) è, in realtà, del tipo "Adventure", contiene come elemento caratterizzante un labirinto tridimensionale. Burroni e scale permettono di muoversi verso l'alto ed il basso

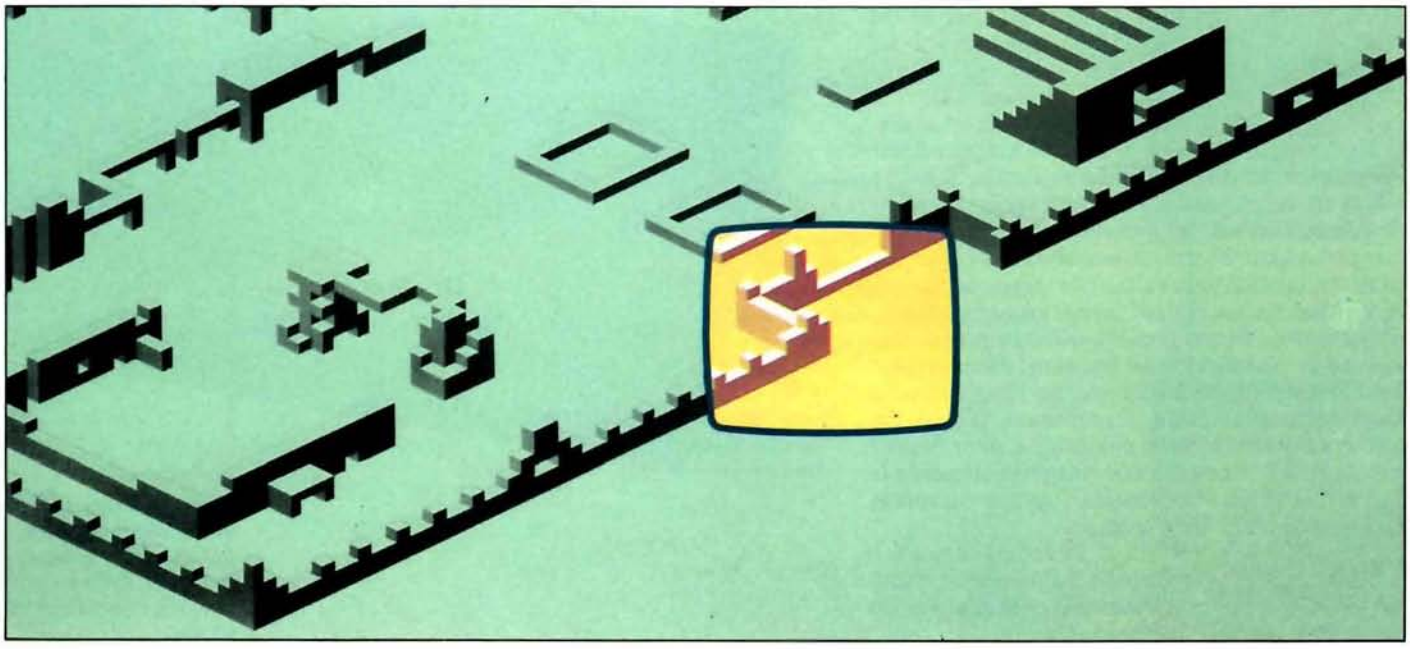
Per giovani e meno giovani, i labirinti sono sempre stati una fonte inesauribile di divertimento, siano essi di dimensioni tali da perdercisi dentro oppure così piccoli da poterli tenere nel palmo di una mano. È ovvio, quindi, che i labirinti siano stati impiegati in una grande varietà di giochi per computer: da una semplice vista aerea, bidimensionale, a labirinti tridimensionali estremamente complessi. I tipi più recenti simulano la veduta di un labirinto dall'interno per cui il giocatore ha proprio la sensazione di trovarsi in un labirinto reale. Per aiutarlo ad orizzontarsi (o forse per confonderlo ancora di più?), in questi labirinti tridimensionali, periodicamente viene mostrata una fugace immagine di una veduta aerea del labirinto stesso.
Via via che gli effetti sonori e grafici si sono fatti più sofisticati, l'immaginazione dei programmatori si è potuta sbizzarrire liberamente: mostri, pozzi senza fondo, trappole e trabocchetti, tutti ben realizzati graficamente, sono gli ingredienti più comuni. Un esempio tipico è il *3D Glooper* (disponibile per il Commodore 64), in cui il giocatore esplora il labirinto seguendo particolari mattonelle del pavimento e può essere attaccato in qualsiasi momento da mostri che gli si avventano contro dallo schermo. Tuttavia, l'improvvisa comparsa di queste creature viene annunciata dal rumore delle loro mascelle battenti!
Esaminiamo adesso alcune tecniche di programmazione di base utilizzate nella progettazione dei labirinti.

Come si crea un labirinto

Il modo più comune di immagazzinare informazioni riguardanti un labirinto è quello di utilizzare matrici a due dimensioni, come ad esempio: M$(RIGA, COLONNA). Ciascun elemento della matrice definisce le caratteristiche di una cella del labirinto. Potremmo usare una stringa di 4 caratteri per rappresentare sud, ovest, nord ed est. Il numero 0 potrebbe indicare l'assenza di una parete, il numero 1 la sua presenza. Quindi, se M$(5,6) contiene la stringa "1011", ciò potrebbe indicare che la cella situata all'incrocio della riga cinque con la colonna sei è circondata da pareti a sud, nord ed est.
Per risparmiare spazio in memoria, i dati potrebbero essere espressi usando numeri e

non stringhe. Inoltre, un numero di quattro cifre potrebbe essere considerato come un numero binario: nell'esempio precedente, pertanto, la cella dotata di pareti a sud, nord ed est, potrebbe contenere il numero 11 decimale (corrispondente al binario 1011).
Inizialmente, tutte le celle possiedono quattro pareti. Determinata casualmente la cella d'ingresso al labirinto, in qualsiasi punto del perimetro, la cella successiva verrà scelta a caso tra una delle tre confinanti. Effettuata questa scelta il procedimento continuerà, selezionando a caso una nuova cella tra una delle tre adiacenti e trascurando quella dalla quale si proviene. Ad ogni nuova scelta di direzione verrà eliminata la parete che divide la cella che stiamo per lasciare da quella in cui vogliamo entrare. Durante la progettazione è necessario verificare se non ci si è spostati al di fuori del perimetro del labirinto (a meno che la parete da eliminare di una particolare cella non debba rappresentare il punto d'uscita dal labirinto). Occorre anche prestare attenzione a non creare circuiti chiusi (tutti i punti del labirinto devono essere accessibili da qualsiasi altro punto).
Quando si incontra una cella che ha meno di quattro pareti (cioè, una cella già visitata), il programma deve scegliere tra una delle due celle adiacenti rimaste. Se tutte le celle confinanti sono già state visitate, il programma tornerà indietro all'ultima cella attraversata e prenderà una nuova via.
Un diverso metodo, per registrare le caratteristiche di una cella, sfrutta un più sofisticato uso della numerazione binaria ed è particolarmente utile nella visualizzazione di immagini tridimensionali.
Una cella può essere configurata in 16 modi diversi: nessuna parete, pareti su tutti i lati, una sola parete (quattro possibilità), pareti su lati opposti (due possibilità), due pareti adiacenti (quattro possibilità), tre pareti adiacenti (quattro possibilità).
Usando l'appropriato numero binario (0-15) per ciascuna di queste, è possibile far "ruotare" il numero verso sinistra o destra per ottenere la corretta prospettiva che il giocatore deve avere di fronte. Per esempio, una parete nord potrebbe essere rappresentata dal valore 2 (0010), una sud da 8 (1000), una est da 1 (0001) ed una ovest da 4 (0100).
Se il giocatore, inizialmente rivolto a nord in una stanza con una sola parete a ovest, volge lo sguardo verso ovest, deve trovarsi di fronte una parete. Quando il giocatore volge lo sguardo alla sua sinistra (verso ovest), la rotazione di una posizione verso destra del gruppo di bit, preso come esempio, ci fornirà la descrizione che noi vogliamo, cioè il numero binario 0100 (4 in decimale), che identifica la parete ovest, si trasformerà nel numero binario 0010 (2 in decimale) che corrisponde a una parete nord.
Quando il giocatore si volta a sinistra, i bit vengono ruotati nella direzione opposta e ciò avviene due volte consecutivamente se la rotazione del giocatore è di 180 gradi.
Ovviamente, è necessario includere un sistema per reinserire dal lato opposto i bit che vengono persi all'estremità destra o sinistra del semi-byte per effetto della rotazione, altrimenti le caratteristiche di una cella cambierebbero ogni volta che il giocatore presente al suo interno si gira verso una nuova direzione. Una cella originariamente definita come 0011 ad esempio (pareti a nord e ad est), deve diventare 0110 se il giocatore si gira verso destra e 1100 se il giocatore compie una nuova rotazione verso destra.
In codice macchina ci sono speciali istruzioni per far ruotare i numeri binari a sinistra e a destra. In BASIC, un numero binario a 4 bit, espresso come numero decimale compreso tra 0 e 15, può essere fatto ruotare a sinistra moltiplicandolo per 2 e quindi sottraendo 15, qualora il risultato sia superiore a 15. Per farlo ruotare a destra occorre dividerlo per 2 se è un numero pari oppure sommare 15 e poi dividere per 2 se è un numero dispari.

"Attacco di formiche"
Quando si usa questo gioco sullo Spectrum, lo schermo del computer si comporta come una finestra aperta su di un grande labirinto. Durante il gioco la finestra si sposta mostrando altre zone dello scenario di gioco



Il Memotech MTX 512

Anatomia di un microcomputer a diffusione europea: tecnologie a confronto

Le caratteristiche del Memotech MTX 512 si avvicinano moltissimo a quelle richieste dallo standard MTX (vedere a pag. 252), salvo il fatto che esso impiega, come generatore di effetti audio, l'integrato 76489 della Texas Instruments (lo standard MTX, infatti, prevede l'uso del AY-3-8910 della General Instruments). Gli altri componenti principali sono: una CPU del tipo Z80, un chip video TMS9918/9928 (Texas Instruments) ed un BASIC che si avvicina moltissimo alla versione Microsoft.
Il Memotech MTX 512 ha un aspetto molto elegante e solido. Sia il contenitore che la tastiera hanno una buona rifinitura. L'accesso all'interno è facilissimo: basta togliere due viti e si separano le due parti dell'involucro. La scheda dei componenti è molto compatta, e sono forse presenti più circuiti integrati di quelli che si trovano in altri apparecchi simili. Chi ha progettato questo computer ha evidentemente preferito impiegare più chip (facilmente rimpiazzabili in caso di guasto), che non sofisticate unità fatte "su misura".
Il manuale operativo non è di facilissima consultazione, essendo privo di un indice, ma gli argomenti sono trattati in maniera molto approfondita. Inoltre, la Memotech ha adottato una politica molto "aperta", nel senso che vengono fornite ampie spiegazioni sul funzionamento interno dell'apparecchio, sulla distribuzione della memoria, sui valori di particolari locazioni utili della memoria, ecc. Il linguaggio BASIC è illustrato ampiamente, ed una parte del manuale riguarda il NODDY (vedere pagina a fianco), l'assemblatore/disassemblatore, la gestione della grafica.
La presenza di un assemblatore/disassemblatore, piuttosto insolita per apparecchi della medesima categoria, assieme al pacchetto di software "Front Panel", offre la possibilità (anche se parzialmente limitata) di scrivere e verificare programmi, di media lunghezza, in linguaggio macchina. Il software "Front Panel" meriterebbe una più ampia descrizione nel manuale, dove invece vengono elencate le varie funzioni, ma senza un'approfondita spiegazione sul loro impiego.
Al Memotech MTX 512 si possono collegare espansioni di RAM fino ad un massimo di 512 Kbyte, una capacità fuori dal comune per un computer basato sul chip Z80.



La tastiera
È una delle migliori nella categoria degli home computer. Dotata di 79 tasti professionali, disposti in maniera convenzionale QWERTY, è molto comoda, oltre ad essere solida ed affidabile

Interfaccia per registratore a cassette

Connettori per joystick
A queste due prese possono essere collegati joystick di varie marche, purché funzionanti secondo lo standard Atari

Connettore per espansioni
Il Memotech MTX 512 è stato concepito per avere notevoli espansioni: una capace scheda di RAM aggiuntiva ed una doppia interfaccia seriale (RS232) sono quelle principali. Con esse è anche possibile collegare più apparecchi, per formare un network e questo aspetto è decisamente allettante per il mercato della didattica

Il clock
Un chip CTC Z80 fornisce tutte le funzioni di temporizzazione necessarie al funzionamento del computer

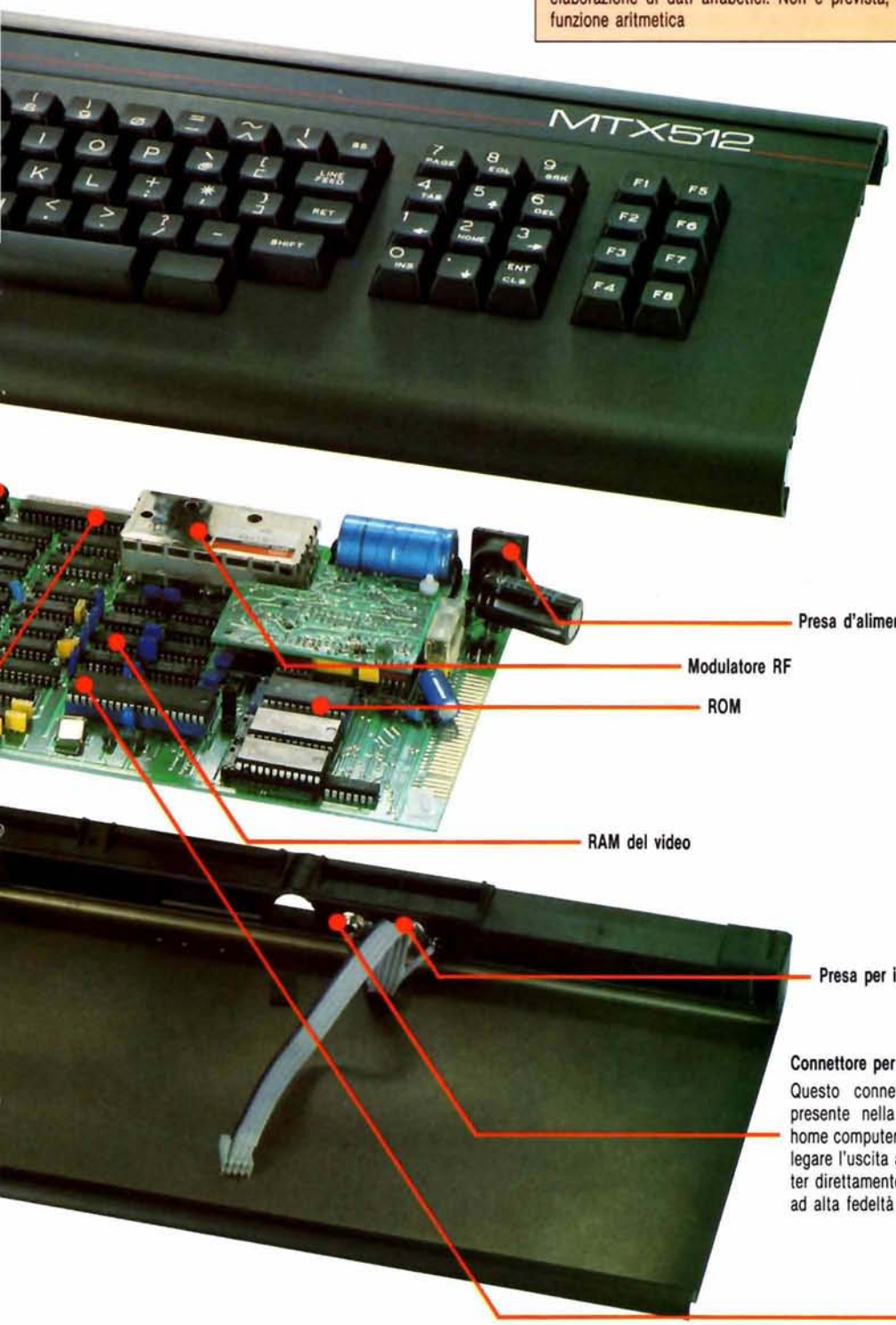
La RAM
L'MTX 512 ha 64 Kbyte di RAM come standard, mentre l'MTX 500 ha 32 Kbyte

La CPU
Il chip impiegato è il noto Z80, impiegato ad una frequenza di 4 MHz

Interfaccia parallela
Questa interfaccia corrisponde allo standard Centronics, per cui, disponendo anche di una porta seriale RS232, si può collegare l'MTX 512 a qualsiasi tipo di stampante

NODDY

Il NODDY è un particolare linguaggio di programmazione, ancora poco diffuso, soprattutto indirizzato ai non esperti. Il software fornito assieme al Memotech MTX 512 comprende una versione ridotta del NODDY. Sono disponibili soltanto 11 comandi (in compenso molto sofisticati), che permettono un discreto grado di elaborazione di dati alfabetici. Non è prevista, infatti, alcuna funzione aritmetica



Connettore per HI-FI

Questo connettore, raramente presente nella categoria degli home computer, permette di collegare l'uscita audio del computer direttamente ad un impianto ad alta fedeltà

Memotech MTX 512

DIMENSIONI

mm. 488 x 202 x 56

PESO

—

CPU

Z80A

CLOCK

4 MHz

MEMORIA

24 Kbyte di ROM. 64 Kbyte di RAM (espandibili a 512 Kbyte), 16 Kbyte di RAM video

VIDEO

24 linee di 40 caratteri, 16 colori disponibili. 127 caratteri predefiniti, 127 caratteri definibili dall'utente

INTERFACCE

Connettore per TV, per registratore a cassette, per video composito e per monitor

LINGUAGGI FORNITI

BASIC, NODDY ed Assembler

ALTRI LINGUAGGI DISPONIBILI

—

ACCESSORI COMPRESI

Cavo di connessione per TV, manuale d'installazione e manuale del BASIC

TASTIERA

79 tasti professionali con disposizione convenzionale

DOCUMENTAZIONE

Ragionevolmente completa. Contiene sufficienti informazioni sul funzionamento interno del computer e sulle varie sezioni della memoria, da permettere ai programmatori un poco più esperti di sfruttare appieno le doti dell'apparecchio

Chip della grafica

Il chip, un TMS 9918/9928, è lo stesso impiegato nei computer TI99/4A e Sord M5. Tuttavia, il sistema operativo dispone di speciali funzioni grafiche che migliorano la gestione grafica in generale (ad esempio, la possibilità di definire più "finestre" sullo schermo)

Scegliamo il menu adatto!

La nostra "rubrica computerizzata" offre molte funzioni: organizziamone la scelta

Nell'ultima lezione abbiamo esaminato alcuni problemi concernenti la ricerca di un particolare elemento in una lista (che per semplicità abbiamo supposto già ordinata). L'argomento verrà trattato a fondo quando si tratterà di scrivere le parti del nostro programma che serviranno per estrarre i nominativi. In questa lezione ci dedicheremo, invece, a sviluppare le altre due parti del programma.

PROGRAMMA PRINCIPALE

```
INIZIO
  INIZIALIZZAZIONE (procedura)
  PRESENTAZIONE (procedura)
  SELEZIONE (procedura)
  ESECUZIONE (procedura)
FINE
```

La prima procedura, INIZIALIZZAZIONE, dovrà svolgere numerosi e complessi compiti (dimensionare le matrici, leggere ed assegnare dei valori iniziali, eseguire una serie di controlli, ecc.): la svilupperemo in un secondo momento, quando avremo definito le varie funzioni in dettaglio.
Svilupperemo le due procedure successive, PRESENTAZIONE e SELEZIONE, seguendo il criterio della programmazione top-down, prestando particolare cura a non sovrapporre, né a duplicare, le varie parti del programma.
Il maggior inconveniente, nella programmazione "top-down", è che il numero di raffinamenti necessari per giungere all'effettiva codifica varia da caso a caso: per una procedura semplice sono, magari, sufficienti due o tre passi, ma, per procedure più complesse, questi possono divenire alquanto numerosi prima che si possa scrivere il "codice sorgente" (così si chiama il listato del programma, scritto in un linguaggio di programmazione). Le ramificazioni, che partono dalla prima descrizione globale e che conducono ai vari livelli di raffinamento, raramente possono essere tracciate su di un solo foglio di carta: è facile, a questo punto, creare confusione e perdere traccia del ragionamento logico.
Un semplice metodo, per tenere tutto quanto in ordine, consiste nel numerare sistematicamente tutti i vari livelli di sviluppo. Noi abbiamo impiegato i numeri romani per indicare i livelli di raffinamento e quelli arabi per le sottosezioni del programma. Ecco come abbiamo strutturato tutto l'insieme:

I. PROGRAMMA PRINCIPALE

```
INIZIO
  1. INIZIALIZZAZIONE
  2. PRESENTAZIONE
  3. SELEZIONE
  4. ESECUZIONE
FINE
```

Lasciando INIZIALIZZAZIONE da parte, per il momento, passiamo al livello successivo:

II. 2 (PRESENTAZIONE)

```
INIZIO
  1. Visualizza messaggio iniziale
  2. CICLO (finché non viene digitato uno spazio)
     FINE CICLO
  3. Chiama il sottoprogramma SELEZIONE
FINE
```

III. 2 (PRESENTAZIONE) 1 (Visualizza messaggio)

```
INIZIO
  1. Pulizia schermo
  2. Visualizza il messaggio iniziale
FINE
```

III. 2 (PRESENTAZIONE) 2 (CICLO di attesa)

```
INIZIO
  1. CICLO (finché non viene digitato uno spazio)
       SE viene digitato uno spazio
       ALLORA
     FINE CICLO
FINE
```

III. 2 (PRESENTAZIONE) 3 (chiama SELEZIONE)

```
INIZIO
  1. GOSUB "SELEZIONE"
FINE
```

A questo punto risulta evidente che, se le sezioni III-2-1 e III-2-3 sono pronte per la codifica in BASIC, la III-2-2 deve essere sviluppata ancora:

IV. (PRESENTAZIONE) 2 (CICLO)

```
INIZIO
  1. CICLO (finché non viene digitato uno spazio)
       SE INKEY$ non è uno spazio ALLORA continua ciclo
```

```
    FINE CICLO
FINE
```

La fase successiva è la codifica in BASIC:

IV. 2 (PRESENTAZIONE) 1 (visualizza messaggio) CODICE BASIC

```
REM SUBROUTINE "PRESENTAZIONE"
PRINT
PRINT
PRINT
PRINT
PRINT TAB(26);"* * RUBRICA COMPUTERIZZATA * *"
PRINT TAB(28);"PRESENTATA NEL CORSO DE"
PRINT TAB(29);"* * IL MIO COMPUTER * *"
PRINT
PRINT TAB(22);"(DIGITARE UNO SPAZIO PER CONTINUARE)"
```

V. 2 (PRESENTAZIONE) 2 (CICLO di attesa) CODICE BASIC

```
LET L = 0
FOR L = 1 TO 1
IF INKEY$ <> " " THEN LET L = 0
NEXT L
```

VI. 2 (PRESENTAZIONE) 3 (Chiama "SELEZIONE") CODICE BASIC

```
GOSUB "SELEZIONE"
RETURN
```

Si noti che, nella V-2-2, abbiamo inizializzato il valore della L (LET L = 0). È buona abitudine, infatti, assicurarsi che le variabili impiegate nelle varie parti del programma siano opportunamente inizializzate e ciò per tre motivi: primo, perché un elenco di istruzioni LET, all'inizio di una sezione di programma, serve per evidenziare quali variabili "locali" vengano impiegate; secondo, perché i valori di dette variabili potrebbero essere stati modificati in altre parti del programma; terzo, perché un'opportuna sequenza di LET può influire positivamente sulla velocità dell'esecuzione, come verrà spiegato più avanti nel corso.
Si noti, inoltre, il diverso metodo impiegato per simulare la struttura REPEAT-UNTIL (RIPETI-FINCHÉ): invece di usare FOR L = 0 TO 1 oppure FOR L = 0 TO 1 STEP 0, abbiamo adoperato la forma FOR L = 1 TO 1, che può essere impiegata su quasi tutte le versioni di BASIC senza apportarvi modifiche. Il ciclo FOR L = 1 TO 1 consiste in un solo passaggio, ma, se al suo interno, il valore di L viene azzerato, il ciclo continua indefinitamente. Per uscirne, possiamo adottare una delle seguenti soluzioni:

```
FOR L = 1 TO 1
IF INKEY$ <> " " THEN LET L = 0
NEXT L
```

oppure:

```
FOR L = 1 TO 1
LET I = 0
IF INKEY$ = " " THEN LET L = 1
NEXT I
```

La sezione di programma (PRESENTAZIONE), appena sviluppata in BASIC, è completa, eccetto per quanto riguarda i numeri di linea, che non si possono aggiungere finché tutti i blocchi del programma non siano pronti per la codifica. Se volessimo provare questo modulo, indipendentemente dagli altri, dovremmo creare dei valori fittizi per le variabili, oltre a stabilire un numero di linea temporaneo per la subroutine SELEZIONE. Si sarà notata la presenza di una nuova istruzione, la TAB, usata congiuntamente alla PRINT. La funzione TAB serve per ottenere che, sullo schermo, il cursore si posizioni ad una colonna prestabilita (il valore numerico racchiuso tra parentesi) prima di visualizzare l'argomento della PRINT. I valori da noi indicati per le TAB faranno apparire le scritte ben centrate su di uno schermo da 80 colonne, ma, impiegandone uno con un diverso numero di colonne (per esempio lo Spectrum ne ha solo 32), i valori andranno opportunamente modificati. La seconda nuova funzione è la "pulizia dello schermo". Questa si può ottenere, in molte versioni di BASIC, con l'istruzione CLS, ma con la particolare versione da noi impiegata (BASIC Microsoft) si impiega la linea di programma PRINT CHR$(12). Il codice 12 corrisponde generalmente al carattere (non visualizzabile) che cancella il contenuto dello schermo e posiziona il cursore nell'angolo in alto a sinistra. In alcuni apparecchi, invece del valore 12 si usa il 24.

```
10 REM PROGRAMMA PRINCIPALE (DI PROVA)
20 PRINT CHR$(12)
30 GOSUB 100
40 END
100 REM SUBROUTINE "PRESENTAZIONE"
110 PRINT
120 PRINT
130 PRINT
140 PRINT
150 PRINT TAB(26);"* * RUBRICA COMPUTERIZZATA * *
160 PRINT TAB(28);"PRESENTATA NEL CORSO DE"
170 PRINT TAB(29);"* * IL MIO COMPUTER * *"
180 PRINT
190 PRINT TAB(22);"(DIGITARE UNO SPAZIO PER CONTINUA-
    RE)";
195 LET L = 0
200 FOR L = 1 TO 1
210 IF INKEY$ <> " " THEN LET L = 0
220 NEXT L
230 PRINT CHR$(12)
240 GOSUB 1000
250 RETURN
1000 REM SUBROUTINE "FITTIZIA" (SELEZIONE)
1010 PRINT "SUBROUTINE FITTIZIA"
1020 RETURN
```

Lo stesso criterio viene adottato adesso per sviluppare la subroutine SELEZIONE.

II. 3 (SELEZIONE)

```
INIZIO
```

1. Visualizza il menu
2. Input della scelta
3. Chiama la subroutine SCELTA

FINE

III. 3 (SELEZIONE) 1 (Visualizza menu)

INIZIO
1. Pulizia schermo
2. Visualizza menu

FINE

III. 3 (SELEZIONE) 2 (Input della scelta)

INIZIO
1. INPUT scelta
2. Controllo di validità

FINE

III. 3 (SELEZIONE) 3 (chiama SCELTA)

INIZIO
1. Trattamento del CASO scelto
   FINE CASO

FINE

III-3-1 (Visualizza menu) può essere ora codificato in BASIC:

IV. (SELEZIONE) 1 (Visualizza menu) CODICE BASIC

```
REM PULIZIA DELLO SCHERMO
PRINT CHR$(12) : REM OPPURE LA FUNZIONE 'CLS'
PRINT
PRINT
PRINT
PRINT
PRINT "1. TROVARE UN RECORD (USANDO IL NOME)"
PRINT "2. TROVARE UN RECORD (USANDO PARTE DEL
      NOME)"
PRINT "3. TROVARE UN RECORD (USANDO LA LOCALITÀ)"
PRINT "4. TROVARE UN RECORD (USANDO LE INIZIALI)"
PRINT "5. ELENCO DI TUTTI I RECORD"
PRINT "6. IMMISSIONE DI UN RECORD"
PRINT "7. MODIFICA DI UN RECORD"
PRINT "8. ANNULLAMENTO DI UN RECORD"
PRINT "9. FINE LAVORO"
```

Le sezioni **III-3-2 (Input della scelta)** e **III-3-3 (chiama SCELTA)** devono essere sviluppate ulteriormente. Partiamo dalla prima di esse.
Assegnare un valore numerico alla variabile che rappresenta la SCELTA è molto semplice: basta un'istruzione del tipo INPUT SCELTA. Tuttavia, poiché le opzioni disponibili sono 9, che cosa accadrebbe se, per errore, digitassimo 0 oppure 99? Per essere certi che il valore della variabile sia corretto, possiamo introdurre un controllo di validità. Ecco una semplice subroutine che serve a questo scopo:

SUBROUTINE DI INPUT E CONTROLLO DI VALIDITÀ

```
1 REM INPUT E CONTROLLO DI VALIDITÀ
10 LET L = 0
20 FOR L = 1 TO 1
30 INPUT "QUALE (1-9)";SCELTA
40 IF SCELTA < 1 THEN LET L = 0
50 IF SCELTA > 9 THEN LET L = 0
60 NEXT L
70 PRINT "LA SCELTA ERA :";SCELTA
80 END
```

Molte versioni di BASIC permettono l'impiego di operatori logici, che semplificano la verifica:

```
10 LET L = 0
20 FOR L = 1 TO 1
30 INPUT "QUALE (1-9)";SCELTA
40 IF SCELTA < 1 OR SCELTA > 9 THEN LET L = 0
50 NEXT L
60 PRINT "LA SCELTA ERA :";SCELTA
70 END
```

In questi esempi, abbiamo usato l'istruzione INPUT per l'immissione della variabile SCELTA. Una particolarità di questa istruzione è che l'esecuzione si arresta in attesa che l'operatore immetta un dato (numerico nel nostro caso) e termina l'immissione con la pressione del tasto RETURN.
Molti programmatori preferiscono, quando si tratta di scegliere fra più opzioni di un menu, offrire una forma d'immissione più pratica ed "amichevole", che permette al programma di proseguire non appena è stata digitata una risposta valida (senza attendere il RETURN). Ciò si ottiene facilmente usando, al posto della INPUT, la funzione INKEY$. Siccome la INKEY$ controlla una sola volta se è stato digitato un carattere sulla tastiera e prosegue oltre, questa funzione è solitamente impiegata all'interno di un ciclo, che viene ripetuto indefinitamente finché il valore "letto" da INKEY$ è diverso da una stringa vuota:

```
LET L = 0
FOR L = 1 TO 1
LET A$ = INKEY$
IF A$ = " " THEN LET L = 0
NEXT L
```

La INKEY$, però (come indica il simbolo "$" nel suo nome), è una funzione che elabora stringhe alfanumeriche e non valori numerici, mentre a noi serve, invece, proprio un valore numerico per operare la successiva selezione. Fortunatamente, il BASIC dispone di una funzione che fa al caso nostro: la VAL (o NUM in certe versioni), che converte il carattere di una stringa in un valore numerico. L'impiego è molto semplice:

```
LET N = VAL(A$) oppure LET N = NUM(A$)
```

Se, per esempio, A$ contiene la stringa "5", alla variabile N viene assegnato il valore numerico 5. Abbiamo così eliminato la necessità di battere il tasto RETURN, ma è ancora consigliabile operare un controllo di validità sulla risposta. Ecco come potremmo riscrivere la parte di programma relativa all'immissione della scelta:

```
FOR L = 1 TO 1
   PRINT "QUALE (1-9) ?"
```

```
    FOR I = 1 TO 1
      LET A$ = INKEY$
      IF A$ = " " THEN LET I = 0
    NEXT I
  LET SCELTA = VAL(A$)
    IF SCELTA < 1 THEN LET L = 0
    IF SCELTA > 9 THEN LET L = 0
NEXT L
```

Infine, riportiamo il listato BASIC completo del modulo SELEZIONE, nel quale sono incluse, a scopo di verifica, subroutine e variabili fittizie, che verranno in seguito sostituite quando il programma sarà sviluppato per intero.
Ricordiamo che i numeri di linea sono provvisori, e potranno essere stabiliti soltanto nella fase finale dello sviluppo.

```
10 PRINT CHR$(12)
20 PRINT "MENU DELLE OPZIONI"
30 PRINT
40 PRINT
50 PRINT
60 PRINT "1. TROVARE UN RECORD (USANDO IL NOME)"
70 PRINT "2. TROVARE UN RECORD (USANDO PARTE DEL
   NOME)"
80 PRINT "3. TROVARE UN RECORD (USANDO LA LOCALITÀ)"
90 PRINT "4. TROVARE UN RECORD (USANDO LE INIZIALI)"
100 PRINT "5. ELENCO DI TUTTI I RECORD"
110 PRINT "6. IMMISSIONE DI UN RECORD"
120 PRINT "7. MODIFICA DI UN RECORD"
130 PRINT "8. ANNULLAMENTO DI UN RECORD"
140 PRINT "9. FINE LAVORO"
150 PRINT
160 PRINT
170 LET L = 0
180 LET I = 0
190 FOR L = 1 TO 1
200 PRINT "QUALE (1-9) ?"
210 FOR I = 1 TO 1
220 LET A$ = INKEY$
230 IF A$ = " " THEN LET I = 0
240 NEXT I
250 LET SCELTA = VAL(A$)
260 IF SCELTA < 1 THEN LET L = 0
270 IF SCELTA > 9 THEN LET L = 0
280 NEXT L
290 ON SCELTA GOSUB 310,330,350,370,390,410,430,450,470
300 END
310 PRINT "(SUBROUTINE 1)"
320 RETURN
330 PRINT "(SUBROUTINE 2)"
340 RETURN
350 PRINT "(SUBROUTINE 3)"
360 RETURN
370 PRINT "(SUBROUTINE 4)"
380 RETURN
390 PRINT "(SUBROUTINE 5)"
400 RETURN
410 PRINT "(SUBROUTINE 6)"
420 RETURN
430 PRINT "(SUBROUTINE 7)"
440 RETURN
450 PRINT "(SUBROUTINE 8)"
460 RETURN
470 PRINT "(SUBROUTINE 9)"
480 RETURN
```

Nella prossima lezione esamineremo la struttura degli archivi e svilupperemo la procedura INIZIALIZZAZIONE.

Basic & Basic

SPECTRUM

In tutti gli esempi riportati, sostituire PRINT CHR$(12) con CLS, ed END con STOP.
SUBROUTINE DI INPUT E CONTROLLO DI VALIDITÀ

```
1 REM INPUT E CONTROLLO DI VALIDITÀ
10 LET L = 0
20 FOR L = 1 TO 1
30 INPUT "QUALE (1-9)";SCELTA
40 IF SCELTA < 1 THEN LET L = 0
50 IF SCELTA > 9 THEN LET L = 0
60 NEXT L
70 PRINT "LA SCELTA ERA :";SCELTA
80 STOP
```

Nel listato finale, apportare le seguenti modifiche:

```
10 CLS
   :
   :
240 NEXT I
250 LET SCELTA = CODE A$ — 48
260 IF SCELTA < 1 THEN LET L = 0
270 IF SCELTA > 9 THEN LET L = 0
280 NEXT L
290 GOSUB (SCELTA * 20 + 290)
300 STOP
```

TAB

Nelle versioni di BASIC prive di questa funzione, è possibile simularla inserendo, all'inizio del programma, una linea

```
5 LET S$ = "  "
```

dove il numero di spazi racchiusi tra doppi apici è arbitrario. Poi, quando dovrebbe essere impiegata, ad esempio, una TAB(12), sostituirla con LEFT$(S$,12).

CHR$(12)

Sull'Oric-1, il Dragon 32, il Lynx ed il Micro BBC, sostituire PRINT CHR$(12) con CLS. Sul Commodore 64 ed il Vic-20, invece, sostituire con PRINT "(tasto shift) + (tasto CLR/HOME)": sullo schermo apparirà un simbolo a cuore. Consultare il manuale in caso di dubbio.

ON.. GOSUB

Non esiste sul Lynx, ma può essere impiegato il metodo illustrato per lo Spectrum (vedere sopra).

VARIABILI

Vedere "Basic & Basic" a pag. 257.

INKEY$

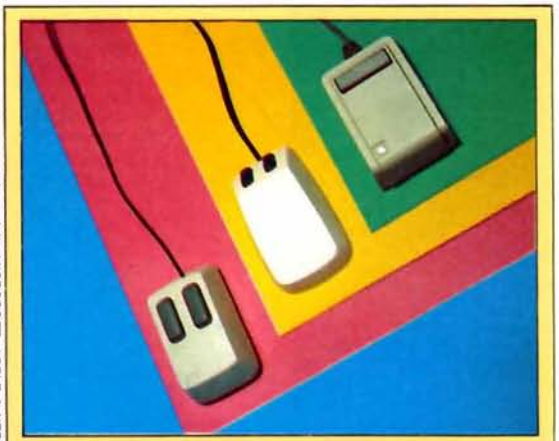
Vedere "Basic & Basic" a pag. 175. Sul Commodore 64 e Vic-20 sostituire LET A$ = INKEY$ con GET A$, ed IN INKEY$ = " " THEN con:

```
GET A$ : IF A$ = " " THEN
```

Uomini e topi

Molte delle funzioni di un programma si selezionano più facilmente con un "topo"

Sino a poco tempo fa, era possibile comunicare con i computer solo attraverso macchine da scrivere elettromeccaniche chiamate "telescriventi": apparecchi rumorosi, ingombranti e di scarso affidamento. In seguito, sono stati sostituiti da veloci e silenziose unità video (Visual Display Unit, o VDU), dotate di tastiera. Le VDU evitano molti inconvenienti tipici delle telescriventi, non ultimo l'enorme spreco di nastro perforato, usato durante l'inserimento d'informazioni. Comunque, tanto i terminali meccanici quanto le VDU con tastiera sono legati ad un formato sequenziale d'immissione: "carattere dopo carattere" e "linea dopo linea". L'utente non può "muoversi" rapidamente sul video, per selezionare una particolare procedura da un menu o per modificare dati sullo schermo. La tastiera non è indispensabile, per esempio, quando si usano terminali grafici oppure quando, nei giochi, si impiegano "track-ball" o joystick. È possibile utilizzare questi apparecchi anche per applicazioni "serie"?
Molti home computer permettono di spostare il cursore, nelle quattro direzioni, per apportare modifiche o correzioni ad un listato di programma o su un testo da elaborare. Ma, in genere, il cursore può essere mosso solo per intervalli corrispondenti alle dimensioni di un carattere o di una linea e l'operatore non può spostarlo direttamente da un punto all'altro dello schermo.
Una soluzione a questo problema fu per la prima volta tentata nel 1960 allo Stanford Research Institute (California). Il primo "topo" (così venne chiamato il nuovo strumento di controllo) fu brevettato nel 1970. Questo strano nome deriva dalla particolare forma: un topo è abbastanza piccolo da stare sul palmo di una mano, ha una "coda" (il cavo di collegamento) e i primi apparecchi di questo tipo avevano 2 "orecchie" (pulsanti di controllo). I comuni joystick, in genere, non garantiscono una sufficiente precisione nel posizionamento del cursore.
Il funzionamento del topo è molto semplice: esso viene fatto scivolare, su una superficie piana, in una qualsiasi direzione; i movimenti vengono direttamente convertiti in spostamenti del cursore sul video. Due sono i sistemi principali per produrre i necessari segnali elettrici del movimento del topo: in entrambi i casi la sua base è caratterizzata da una grande sfera che poggia sulla superficie su cui viene mosso il topo. Il moto di rotazione impresso alla sfera viene poi trasferito meccanicamente a due rulli cilindrici interni. Nel primo sistema le estremità di questi cilindri sono collegate a speciali ingranaggi, caratterizzati da piste alternate di materiale conduttore e non conduttore. Gli impulsi ricevuti vengono decodificati dal software operativo del topo, che traduce gli impulsi direttamente in coordinate per il posizionamento del cursore sullo schermo. Nel secondo caso, a ciascun rullo è collegato un disco munito di una serie di fessure. Una fonte di luce è costantemente diretta sul disco ed il raggio luminoso viene captato da una fotocellula posta dall'altro lato del disco. Gli impulsi luminosi che attraversano le fes-



I tipi ... di topi!
Alcuni dei più recenti microcomputer per applicazioni commerciali vengono venduti già dotati di un topo, per altri viene offerto come "optional". La maggior parte dei topi ha uno, due o tre pulsanti ed è dotata di una sfera ruotante nella parte inferiore

per concessione della MICROSOFT, APPLE & XEROX IAN McKINNELL



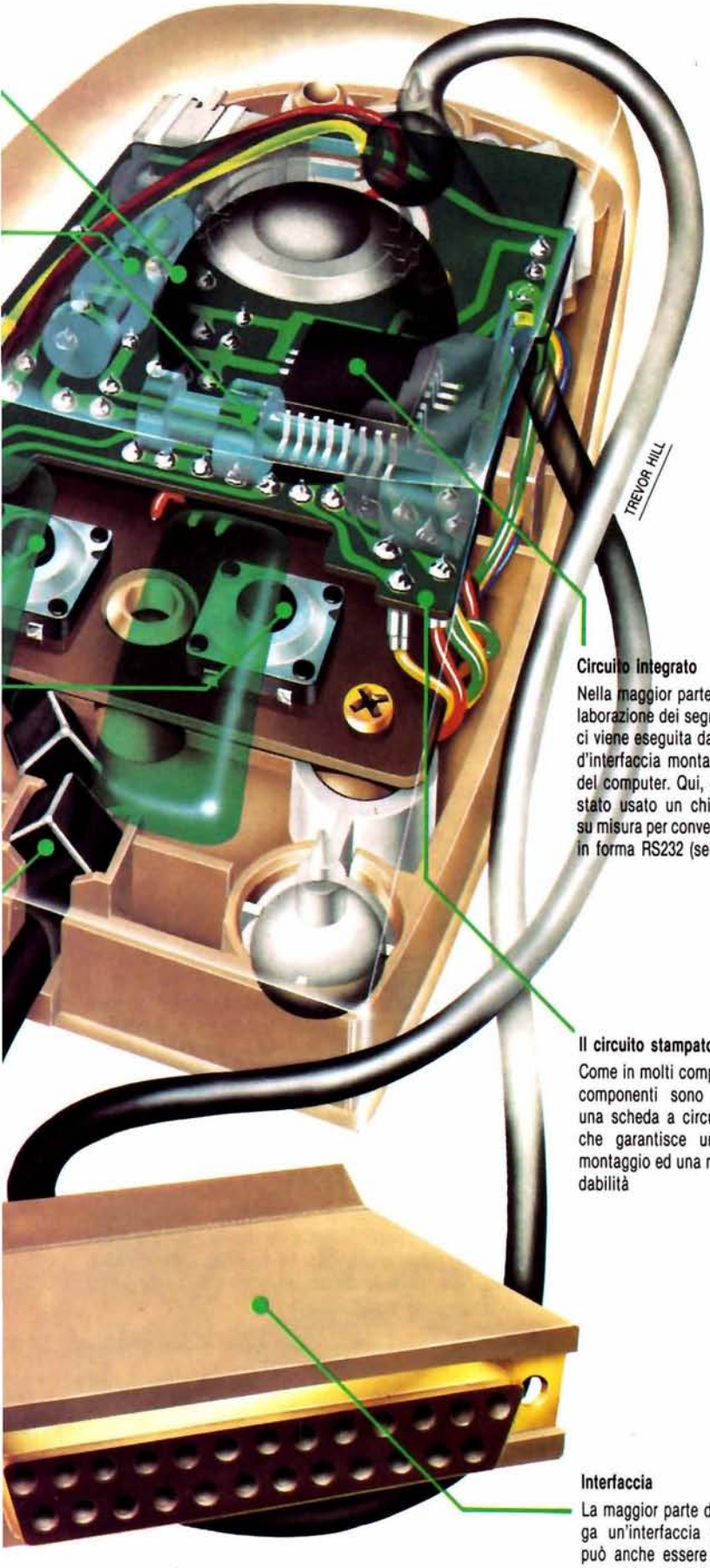
Sfera principale
Un grosso cuscinetto a sfera poggia sulla superficie sulla quale viene mosso il topo. In alcuni, la sfera è fatta di gomma rigida per evitare slittamenti

Cilindri di codifica
Questi due cilindri sono costantemente in contatto con la sfera e rilevano i suoi movimenti secondo i due assi X e Y. Ad una estremità di ciascun rullo c'è un apparecchio di codifica che, al suo ruotare, produce impulsi elettrici

Pulsanti
La funzione dei due pulsanti dipende dal software usato. Di norma, uno viene usato per selezionare opzioni da un menu, l'altro per spostare oggetti visualizzati sullo schermo

Microinterruttori
Sono montati sul circuito stampato, al di sotto dei pulsanti. È sufficiente una piccola pressione sul topo per aprire o chiudere il circuito

Passacavi in gomma
Il topo deve essere libero di muoversi sul tavolo di lavoro ed il passacavi è molto importante perché ammortizza le tensioni tra il cavo di collegamento ed il circuito stampato



Circuito integrato

Nella maggior parte dei topi, l'elaborazione dei segnali elettronici viene eseguita da una scheda d'interfaccia montata all'interno del computer. Qui, comunque, è stato usato un chip progettato su misura per convertire i segnali in forma RS232 (seriale)

Il circuito stampato

Come in molti computer, tutte le componenti sono montate su una scheda a circuiti stampati, che garantisce un più facile montaggio ed una maggiore affidabilità

Interfaccia

La maggior parte dei topi impiega un'interfaccia speciale, ma può anche essere collegata ad una porta seriale RS232 mediante il connettore standard a 25 canali

sure vengono convertiti in segnali elettrici, che subiscono lo stesso trattamento riservato a quelli prodotti dal sistema precedente.
Esistono anche altri sistemi. In un caso, ad esempio, il topo viene mosso su di una speciale piattaforma ricoperta da una particolare configurazione di punti. Una luce, posta all'interno del topo, illumina la superficie della piattaforma e permette all'apparecchio di ispezionare, mediante uno speciale chip di elaborazione ottica, la configurazione di punti che si trova sotto di esso. Ogni movimento del topo produce un cambiamento nelle informazioni lette dal chip, permettendo di calcolare immediatamente la distanza e la direzione dello spostamento. Questo sistema ha il vantaggio di non possedere parti mobili, ma è molto più costoso dei precedenti.
Dopo che il cursore è stato posizionato sul video nel punto voluto, la sua posizione può essere registrata nella memoria del computer premendo una delle orecchie (pulsanti) del topo. Il numero di pulsanti varia a seconda della casa produttrice: alcuni sistemi ne usano molti, altri soltanto due o tre. La Microsoft ha deciso di inserirne due, mentre il topo del Lisa (il computer della Apple) ne ha uno solo. Il pulsante può essere usato per selezionare una opzione da un menu (programmi come il Microsoft's MultiTool Word adottano questa procedura) e per altre funzioni particolari.
Questi apparecchi sono addirittura indispensabili, se usati con software molto sofisticato, come quello del Lisa della Apple. In esso il pulsante viene premuto una prima volta per selezionare un'"icona" (vedere a pag. 262) da un menu presente sul video ed una seconda volta per rendere operativa quella particolare applicazione.
Il vantaggio principale di tutti i topi (e del software progettato per completarli) è che possono essere usati da utenti non troppo esperti nell'uso della tastiera. Invece di scrivere il nome di un programma o di battere particolari lettere o numeri per selezionare una funzione, l'utente muove il topo in modo da puntare il cursore sull'applicazione o la procedura di cui ha bisogno e quindi preme il pulsante per renderla operativa.
Purtroppo, il topo non può sostituire del tutto la tastiera, poiché nuovi testi o numeri dovranno sempre essere inseriti nel computer mediante la tastiera, ma rende molto più semplice la manipolazione di questi dati. Alcune prove, eseguite dalla Apple durante lo sviluppo del Lisa, hanno dimostrato che un utente del tutto inesperto nell'uso del computer può imparare ad usare il software gestito dal topo del Lisa in poco più di un quarto d'ora. L'apprendimento, su sistemi convenzionali, delle stesse funzioni, richiede circa 20 ore sia perché si incontrano difficoltà ad imparare l'uso della tastiera e sia perché è necessario imparare a memoria comandi complicati e lunghi. I topi elettronici presto diventeranno un normale corredo degli home computer: sono efficaci, facili da usare e non intimoriscono i meno preparati come accade con le normali tastiere QWERTY.

Il codice Hamming

Un ottimo sistema per individuare e correggere gli errori durante il trasferimento dei dati

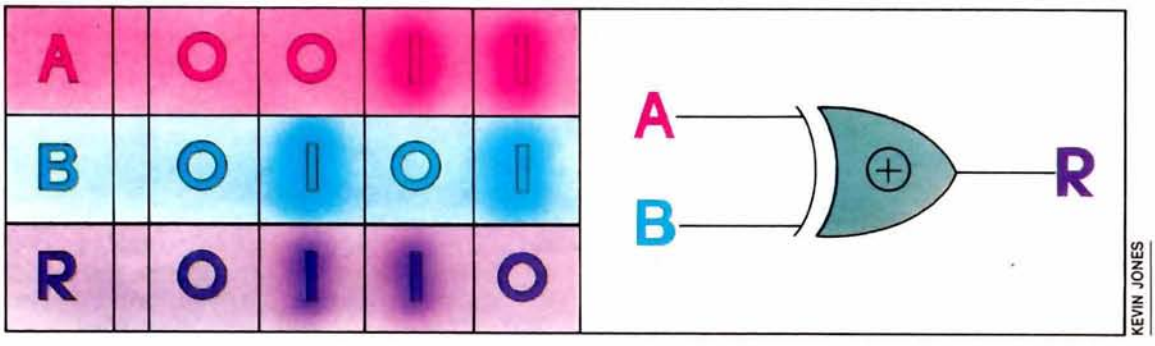
Tutti hanno qualche storiella da raccontare su presunti errori commessi dai computer. In realtà le macchine raramente sbagliano ed eventuali errori sono imputabili all'operatore del computer (a volte basta un banale errore di battitura). Tuttavia, anche se raramente, può accadere che un computer commetta un tipo d'errore non addebitabile all'intervento umano: si tratta dei cosiddetti "bit error", e si verificano quando un singolo bit di un gruppo di dati, cambia il suo valore da 0 ad 1 o viceversa. I bit error dipendono dal cattivo funzionamento di un componente dell'hardware, ad esempio di un chip della RAM. Ecco perché in molti home computer, all'accensione, parte automaticamente un programma "autodiagnostico" per rilevare eventuali errori di questo tipo.

Purtroppo, esistono dei bit error più subdoli, che, dipendendo da altri fattori esterni, si verificano anche se tutta la RAM è risultata perfettamente funzionante dopo il test diagnostico. Gli home computer sono progettati per ambienti domestici, ma durante un'ondata di caldo estiva è abbastanza probabile che la temperatura esterna superi la massima temperatura operativa delle componenti di un computer. Raramente il danno è permanente, ma errori di bit possono manifestarsi con la comparsa sullo schermo di un carattere al posto di un altro, ad esempio B al posto di A, o, se il bit fa parte di un importante puntatore, può darsi che il programma si blocchi (in gergo: "si inchiodi"): ciò comporta, necessariamente, il "reset" dell'apparecchio, con perdita (o danneggiamento) del lavoro fatto. Le conseguenze potrebbero essere disastrose per cui sono stati studiati diversi metodi che permettono di individuare gli errori. Il più semplice è il cosiddetto "controllo di parità" (vedere a pag. 253). Un metodo alternativo impiega una "somma di controllo" (checksum), e viene frequentemente usato nell'archiviazione di dati su nastri o su dischi magnetici.

Generalmente, i dati vengono trattati dai computer in blocchi di 128 byte di cui l'ultimo viene usato come checksum. Questo byte contiene un valore ottenuto sommando tutti gli altri byte (ciascuno dei quali ha un valore compreso tra 0 e 255) modulo 256, ossia il resto che si ottiene dividendo la somma totale per 256. Ecco un esempio:

Dati: 114,67,83...(altri 121 valori)...36,154,198
Somma di questi 127 valori = 16673
Totale diviso per 256 = 65, resto 33
Quindi, checksum = 33

La somma dei valori dei byte (16673) è uguale a 65 gruppi di 256 più un resto di 33, valore che viene scritto nel centoventottesimo byte come "somma di controllo". Quando il computer legge nuovamente il blocco, ricalcola la somma dei valori e, se il risultato è diverso dal checksum, deduce che è avvenuto un errore durante la fase di scrittura o di lettura. Con i sistemi di controllo di parità e di checksum, il computer non può individuare quale bit sia danneggiato. Se l'errore si verifica durante uno scambio di dati tra computer, allora il computer ricevente può chiedere la ritrasmissione di un blocco di byte, ma nel caso di un errore di registrazione, non c'è alcun modo di recuperare i dati danneggiati. Un valido sistema per l'individuazione e la correzione automatica degli errori impiega i codici Hamming (dal nome del loro inventore R.W. Hamming dei Laboratori Bell Telephone). Tutti i sistemi di correzione degli errori lavorano sul principio della "ridondanza". I linguaggi umani contengono un alto grado di ridondanza: se un errore di battitura compare in un testo o una qualsiasi interferenza si sovrappone alle parole di una conversazione telefonica, è spesso possibile intuire le parole perdute considerando il contesto della frase. Talvolta si usano frasi con ridondanza supplementare (ad esempio, le parole alfa, bravo, Charlie al posto delle lettere "a", "b", "c" impiegate dai radioamatori).



OR esclusivo

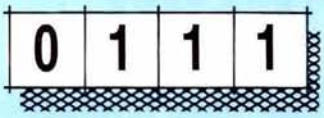
Una semplice porta di OR esclusivo possiede due ingressi e un'uscita. Se tutti gli ingressi sono a livello 0 logico, allora l'uscita sarà 0. Se uno dei due ingressi è a 1, l'uscita sarà 1. Ma, se entrambi gli ingressi sono a 1, l'uscita sarà 0. Questa ultima condizione differenzia la porta OR dalla porta XOR (cioè OR esclusivo). L'operazione può essere rappresentata in una tavola della verità. Quando una XOR possiede più di due ingressi, l'uscita sarà 1 se c'è un numero dispari di 1 all'ingresso. Tramite questi meccanismi vengono creati i bit di parità e di verifica degli errori

Supponiamo di immettere nel nostro computer una parola di "x" bit di lunghezza, composta da "y" bit di dati reali e da "z" bit ridondanti (cioè x = y + z). Quando abbiamo spiegato il controllo di parità, avevamo una parola di 8 bit, per cui y sarebbe pari a 7 e z ad 1. Per i codici Hamming, invece, 7 dovrà essere proporzionalmente più grande. Poniamo adesso che un errore di un singolo bit possa verificarsi in uno qualsiasi dei bit "x" (i bit "z" ridondanti sono naturalmente soggetti ad errore come lo sono i bit dati "y"). Se la probabilità di un errore in una parola è, diciamo, di uno su un milione, allora la probabilità di due errori in una parola è di uno su mille miliardi, per cui questa eventualità risulta trascurabile.
Quando il dato viene ricevuto dal sistema, ci saranno x+1 probabilità: cioè, o non ci saranno errori, o il primo bit dati conterrà un errore e così via fino all'xesimo bit. Ora, con z bit ridondanti, noi possiamo rappresentare 2 elevato alla z situazioni. Quindi, perché la parola sia a prova d'errore, cioè a prova di bit error, avremo la formula:

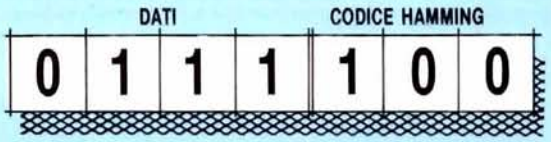
$$2^z > = y+z+1$$

Se y vale sette (per i codici ASCII), allora z dovrà essere quattro. Se y è quattro, come nell'esempio qui a fianco, allora z dovrà essere tre. Tuttavia se y vale 16, allora z dovrà essere aumentato solo fino a 5. Ne consegue che i codici Hamming sono molto più efficienti per parole lunghe che per quelle corte. In un codice Hamming, ciascuno dei bit ridondanti agisce come una verifica di "parità pari" su una differente combinazione di bit nella parola. Se si verifica un errore di bit durante la trasmissione, allora uno o più bit di controllo risulteranno errati e la combinazione di questi ultimi indicherà il bit danneggiato nella parola (vedere esempio). Basta quindi che il software del computer ricevente inverta il valore in esso contenuto, cioè riportarlo a 0 se è 1 o viceversa.
La chiave per comprendere il modo in cui lavorano i codici Hamming, è la differente combinazione di bit su cui ciascun bit Hamming agisce come verifica di parità. Il numero totale di bit è effettivamente diviso in insiemi differenti ma sovrapposti, costruiti in modo che nessuna coppia di bit appaia nella stessa combinazione di insiemi.
Il computer ricevente compie delle verifiche di parità sugli stessi insiemi sui quali ha operato l'apparecchio trasmittente per creare il codice Hamming. Se uno qualunque dei bit, inclusi i bit Hamming, è stato danneggiato nella trasmissione, allora uno o più di questi insiemi si dimostrerà inesatto alla verifica di parità. La combinazione delle verifiche fallite indicherà un unico bit e cioè quello errato.
Alcuni computer usano i codici Hamming anche per le loro operazioni interne di memoria. In alcuni casi è addirittura possibile rimuovere un intero chip di RAM e vedere il computer continuare il suo lavoro.

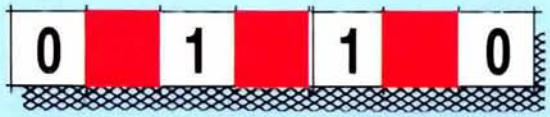
COME OPERA UN CODICE HAMMING



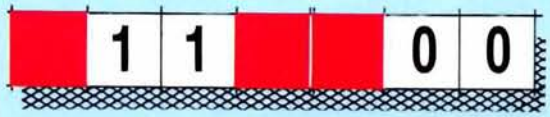
Supponiamo di voler inviare questi quattro bit dati.



Ad essi dobbiamo aggiungere un codice Hamming a 3 bit, un unico insieme di bit generato dal computer per verificare le seguenti condizioni:



Osservando solo 4 dei 7 bit, deve comparire un numero pari di 1 visibili



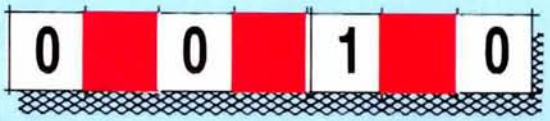
Similmente, escludendo questi 4, ci deve essere un numero pari di 1



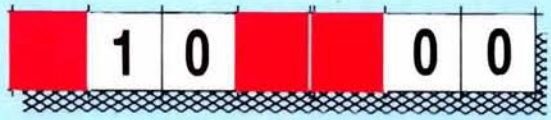
Anche in questo insieme di bit ci deve essere numero pari di 1. La definizione dei 3 bit, determinanti queste condizioni, richiede che il computer risolva tre equazioni simultaneamente



Ma supponiamo che, durante la trasmissione, il terzo bit da sinistra venga danneggiato, cioè che sia cambiato da 1 a 0



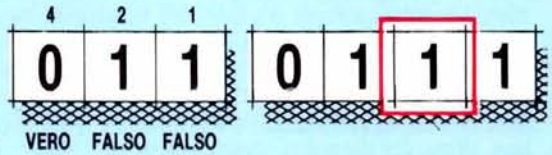
Se il computer ricevente esegue la prima di queste 3 verifiche sui dati, ottiene un risultato errato perché è presente un numero dispari di 1 visibili. Questo ci dice che c'è stato un errore ma ancora non sappiamo quale bit sia stato danneggiato



Ugualmente, la seconda verifica produce un risultato errato



Tuttavia la terza verifica risulta corretta: infatti è ancora visibile un numero pari di 1



È la combinazione delle verifiche corrette e di quelle errate che indica il bit danneggiato. Se noi simboleggiamo una verifica errata con 1 ed una corretta con 0, allora, scrivendo i risultati in ordine inverso, otterremo il numero binario equivalente a 3: esso indica che il terzo bit è quello danneggiato e che deve essere riportato da 0 a 1

Questa tecnica funziona anche se ad esser danneggiato è uno dei bit del codice Hamming. Se tutte e tre le verifiche risultano errate, dando una combinazione di 111, ciò significherà che il bit più a destra è stato danneggiato; se invece tutte e tre risulteranno corrette non ci sarà stato alcun errore. Questo tipo di codice di correzione fallisce solo se si è verificato più di un errore nei sette bit

Norbert Wiener

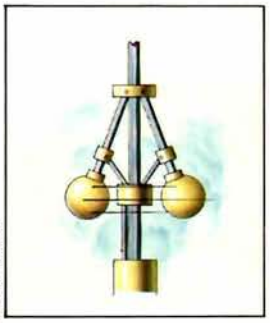
L'uomo che ha sviluppato i principi della cibernetica



Norbert Wiener nacque nel 1894 nel Missouri, USA. Dopo aver conseguito il diploma in matematica a 14 anni ed il dottorato in logica a 18, si recò a studiare a Göttingen, in Germania, con David Hilbert.
Per molti anni ha lavorato al Massachusetts Institute of Technology approfondendo le sue conoscenze di fisica probabilistica e concentrandosi nello studio statistico del moto di particelle immerse in un liquido (fenomeno noto come "moto Browniano").
Quando scoppiò la II Guerra Mondiale, Wiener mise le proprie conoscenze al servizio del governo degli Stati Uniti e cominciò a lavorare su problemi matematici riguardanti tecnologie di puntamento contro un bersaglio mobile. Lo sviluppo di sistemi di controllo per puntamento automatico, i suoi studi nel campo della fisica probabilistica ed i suoi vasti interessi, che si estendevano dal campo filosofico alla neurologia, formarono la base del suo libro *Cybernetics*, pubblicato nel 1948.
La cibernetica è lo studio dei controlli di autoregolazione inseriti in sistemi stabili, siano essi meccanici, elettrici o biologici. Wiener capì che l'informazione, intesa come dato quantitativo, era importante quanto l'energia o la materia: un filo di rame, ad esempio, può essere studiato riferendosi sia all'energia che può trasportare sia alla quantità d'informazioni che può fluire in esso. La rivoluzione informatica si basa, in parte, sulla seguente teoria: il potere in futuro non sarà più legittimato dal possesso di territori, mezzi di produzione o interessi economici, ma dal controllo dell'informazione. Quindi il contributo che Wiener ha dato alla scienza dei computer non riguarda tanto lo sviluppo dell'hardware, quanto la creazione di un ambiente intellettuale al cui interno possano liberamente svilupparsi computer ed automi.
La parola "cibernetica" deriva da una parola greca che significa "governatore". Wiener studiò il governatore della macchina a vapore di James Watt, che regolava automaticamente la velocità della macchina e si persuase che, per lo sviluppo dei computer, questi dovevano arrivare ad imitare le capacità umane nel controllo delle proprie attività.
Il termostato di una caldaia è un semplice esempio di sistema di controllo: esso regola il riscaldamento di un ambiente intervenendo sulle variazioni di temperatura rispetto ad un livello ottimale. L'intervento umano si limita unicamente alla definizione di questo livello. Wiener chiamò questa capacità di autoregolazione ed autocontrollo "feedback negativo" (retroazione negativa): "feedback" perché il risultato (output) del sistema (il calore) influenza il comportamento futuro del sistema stesso e "negativo" perché le variazioni controllate dal termostato tendono sempre a riportare la temperatura al livello ottimale prefissato.
Un sistema di questo genere, ma capace anche di scegliere da sé la temperatura (ed altri fattori) viene chiamato sistema a "feedback positivo". Quando un automa può fare tutto ciò, fino a riprodursi da solo, allora esso si avvicina alla condizione umana.
Le teorie di Wiener sulla cibernetica potrebbero essere viste come una superscienza: una scienza delle scienze che ha incoraggiato ricerche in molte aree relative ai sistemi di controllo ed a quelli che riguardano l'informazione. Ogni cosa è informazione: i cambiamenti che noi avvertiamo nel mondo che ci circonda vengono registrati dalla vista, dall'udito o dagli altri sensi, strumenti che ci permettono di selezionare solo alcuni dati da una totalità che altrimenti ci sommergerebbe.
L'informazione può anche essere studiata in modo statistico, indipendentemente dai suoi contenuti. Per esempio, osservando la frequenza con cui ricorrono determinati simboli, è possibile interpretare molti tipi di codici. Nella lingua inglese la lettera "e" ricorre molto spesso e la lettera "t" è la seconda più frequentemente usata. Analizzando grandi campionature di un codice e confrontando i risultati con campionature tipo della lingua inglese, è possibile individuare le lettere chiave e quindi cominciare a decifrare il codice.
Wiener morì nel 1964, prima che iniziasse la rivoluzione dei microcomputer; tuttavia egli previde ed approfondì molti problemi che si sarebbero successivamente presentati in questa nuova tecnologia.

Velocità controllata

Wiener rimase affascinato dall'idea del "governatore" di pressione, che è uno dei migliori e più semplici esempi di feedback negativo. Due pesi sono collegati, mediante braccia imperniate, ad un albero rotante collegato a sua volta ad un volano della macchina a vapore. Quando la velocità della macchina cresce, i due pesi, spinti dalla forza centrifuga, si innalzano portandosi verso l'esterno. Questo movimento, grazie ad un apposito giunto, fa chiudere leggermente la valvola a farfalla della macchina. Ciò ha l'effetto di stabilizzarne la velocità ad un livello voluto dall'operatore. Lo stesso sistema viene ancora adoperato per regolare la velocità negli ascensori. I moderni computer possono svolgere compiti di controllo molto più sofisticati, ma il principio di base resta il medesimo



KEVIN JONES

La comunicazione di massa

Le trasmissioni via cavo non servono soltanto per i canali televisivi



IAN McKINNELL

I collegamenti "via cavo" offrono molti vantaggi, tra i quali la possibilità di allacciarsi direttamente da casa con numerosi canali televisivi. Ma vi è un'altra utilizzazione sulla quale, per adesso, poco è stato detto, ma che potrebbe rivoluzionare il nostro modo di comunicare con gli altri: sfruttando uno di questi canali è possibile formare una rete di collegamento privato tra più utenti di home computer. Abbiamo già esaminato due tecniche che permettono di utilizzare gli home computer come terminali di comunicazione: i sistemi tipo "viewdata" (vedere a pag. 264) che permettono di accedere a grandi banche di dati di carattere generale ed i Network "locali" (vedere a pag. 218).

Con un sistema di banche dati (quale il Mailbox della Prestel) è possibile lo scambio di messaggi tra abbonati ed ogni abbonato al Prestel può acquistare beni e fruire di servizi (la prenotazione di vacanze, ad esempio) direttamente da casa propria. Tuttavia, ambedue i sistemi citati hanno alcune limitazioni. Infatti, facendo parte di un Network, ci sarà possibile comunicare con qualsiasi altra stazione, ma le più ampie reti di comunicazione, gestite da microcomputer, riescono raramente ad estendersi geograficamente oltre i 2 o 3 chilometri. Con molta probabilità, quindi, la banca, il macellaio o la compagnia di assicurazioni, presso i quali ci serviamo, non faranno parte dello stesso Network. Il maggiore ostacolo alla creazione di reti di comunicazione su larga scala (per un'intera città, ad esempio) è costituito dal fatto che la potenza dei segnali inviati lungo le linee di trasmissione si indebolisce proporzionalmente alla distanza da percorrere. Il solo modo per eliminare questo inconveniente consiste nell'inserire sulla linea, ad intervalli regolari, dei ripetitori o strumenti di amplificazione del segnale.

Una seconda considerazione da fare riguarda la densità di traffico, ossia il volume d'informazioni da trasmettere lungo la linea. Questo fattore influisce sulla larghezza di banda (gamma delle frequenze) della trasmissione. Come regola generale, sono necessari due hertz di larghezza di banda per ciascun bit da trasmettere al secondo. Per una trasmissione a 300 baud (ossia 300 bit per secondo) occorrono 600 Hz; per una a 1200 baud saranno necessari 2,4 KHz. In una normale chiamata telefonica viene usato un minimo di 3KHz, per cui questa è la larghezza di banda nominale di una linea telefonica. Per la trasmissione di un'immagine televisiva a colori sono invece necessari ben 8 MHz: una frequenza 3000 volte superiore a quella per una chiamata telefonica. In altre parole, con la larghezza di banda necessaria per trasmettere una sola immagine televisiva si

Una città per il 2000

Milton Keynes, una città costruita secondo i più moderni criteri, è stata dotata fin dalla sua nascita di un adeguato sistema televisivo via cavo. 22000 appartamenti usufruiscono di sette canali televisivi (sei "in diretta" ed uno che trasmette ininterrottamente film) e di sei canali radiofonici VHF. Il passo successivo verso una rete informativa integrata prevede che le forniture di elettricità e gas vengano controllate centralmente e non in ciascun edificio

potrebbero trasmettere 3000 diverse conversazioni telefoniche.
La capacità di una linea di trasmissione è più facilmente misurabile in base al numero di conversazioni telefoniche che essa può inoltrare contemporaneamente. La massima portata dei cavi attualmente adoperati si aggira sulle 20000 conversazioni alla volta, ma è già stato sperimentato con buon successo un cavo coassiale da 500 MHz, dotato quindi di una capacità potenziale di trasmissione pari a 167000 conversazioni simultanee. Il diametro del cavo è di un centimetro: ma una singola fibra di vetro, più sottile di un capello, può trasportare fino a 10000 conversazioni telefoniche!
L'hardware, per questo tipo di comunicazioni, è già in commercio e sarebbe abbastanza semplice creare una rete in ogni piccola e grande città. Vediamo come sarebbe possibile organizzare una rete di comunicazione per computer basata sul Modello B Micro della BBC e sull'impiego del sistema Econet della Arcon. Quando abbiamo esaminato i Network locali, abbiamo visto che la stazione di lavoro più lontana può distare dal centro di controllo della rete 500 metri al massimo. Ciò significa che essa può coprire un'area dal diametro di circa un chilometro. Abbiamo inoltre visto come sia possibile accogliere nel sistema fino a 254 abbonati, lasciando una stazione dedicata al lavoro di gestione degli archivi (ovvero del disco fisso in comune) ed un'altra al controllo della stampante. Ma se noi riservassimo una terza stazione ad un canale di comunicazione con un'altra rete, allora, con semplici modifiche al software operativo, saremmo in grado di collegare assieme le due reti. Questo collegamento richiederebbe due macchine dedicate allo scambio di messaggi fra le reti, in comunicazione tramite le rispettive porte parallele.
Naturalmente questo è un modo piuttosto empirico di risolvere il problema del collegamento tra tutti i membri di una comunità, ma, se fosse accolto con favore, potrebbe essere sostituito da una rete di collegamento progettata appositamente. È comunque improbabile che questo progetto vada in porto dal momento che verrebbe richiesto separatamente da ogni gruppo di utenti che adotta lo stesso tipo di microcomputer. Una rete di questo tipo si dimostrerà veramente efficiente se sarà "trasparente", se cioè permetterà a microcomputer diversi di comunicare tra loro (per esempio, uno Spectrum con un Dragon 32). È necessario quindi un regolatore del sistema centrale che permetta la conversione tra i vari protocolli usati da differenti tipi di computer. Il regolatore deve anche essere il punto di collegamento tra il Prestel e gli altri sistemi viewdata, tra il sistema bancario e le società imprenditrici, tra il Sistema Sanitario ed altri servizi pubblici, insomma tra tutte le componenti meccanizzate della società senza alcuna preoccupazione di compatibilità tra gli hardware.
Come trovare i fondi necessari per installare e far funzionare un sistema di questo tipo? L'esperienza inglese ha dimostrato che questo è l'ostacolo maggiore. Il governo inglese, quando propose per la prima volta un progetto per la creazione di un sistema televisivo "via cavo", tese a sottolineare i vantaggi che un simile sistema avrebbe portato alla tecnologia industriale dell'informazione.
Come conseguenza, un certo numero di potenziali utenti avviò delle ricerche di mercato sulla sua utilizzazione. Anche una catena di supermarket a livello nazionale approntò addirittura un progetto pilota di "tele-acquisto" per le sue sedi, per verificare le reazioni del pubblico.
L'esperimento ha avuto una durata di meno di tre mesi e, qui come altrove, non ha incontrato un gradimento apprezzabile.
Quando il progetto di spesa per la televisione via cavo venne presentato al parlamento britannico, i risultati delle ricerche di mercato furono talmente deludenti, che le autorità compententi decisero una pausa di riflessione sull'installazione di sistemi computerizzati a due vie. In seguito, si dette maggior rilievo al settore ricreativo per computer, che a quello dell'informazione. Addirittura, si è

Trasmissioni via satellite

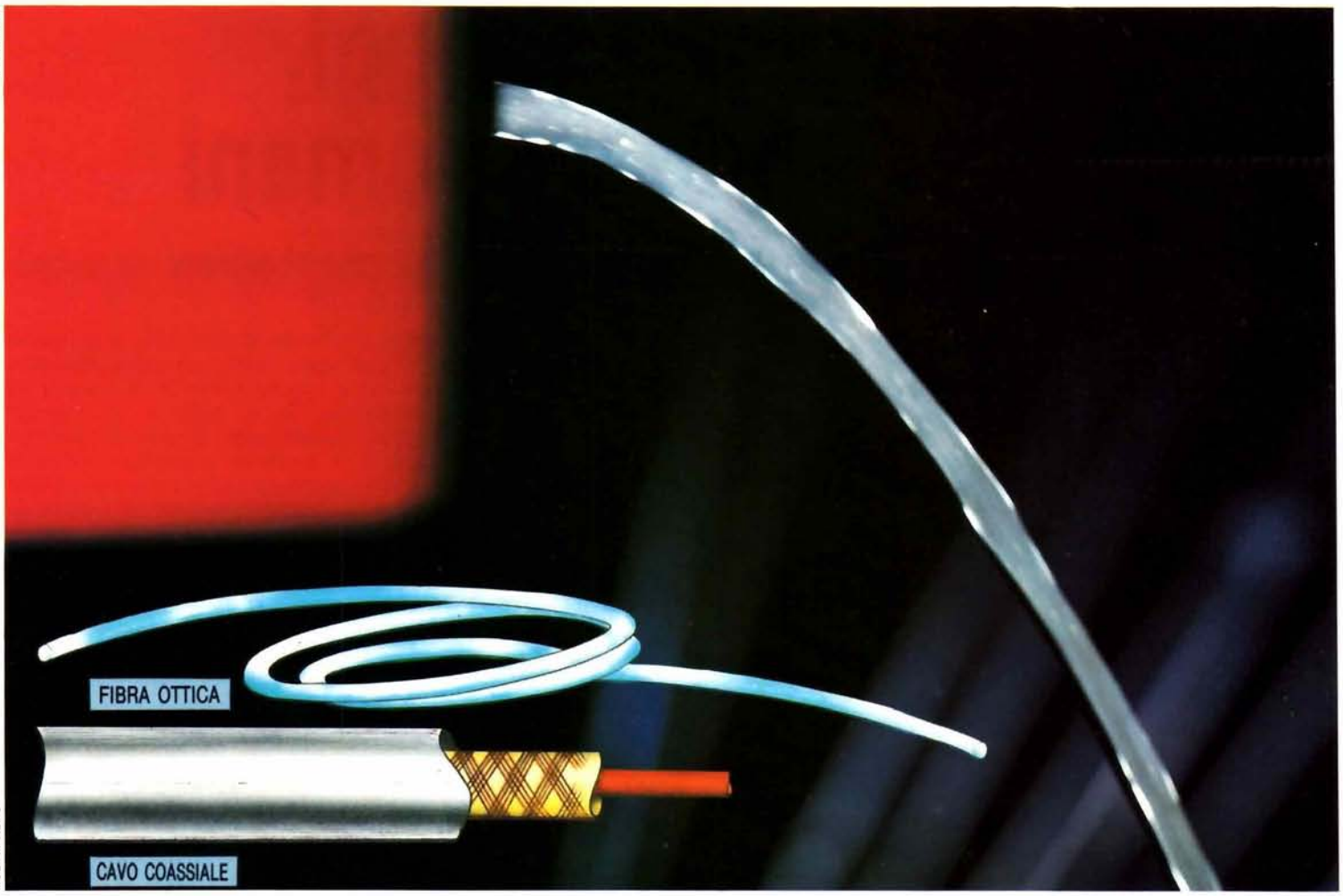
La MTV (una rete televisiva americana) impiega un sistema di trasmissione via satellite per inoltrare ai 1650 consociati i programmi che, successivamente, vengono inviati via cavo alle case dei tredici milioni e mezzo di abbonati in tutti gli Stati Uniti. La MTV trasmette 168 ore di programmi alla settimana, tutti in stereofonia. Ogni ora contiene otto minuti di pubblicità. Sebbene il servizio sia strettamente unidirezionale, la compagnia incoraggia lo sviluppo di uno scambio di opinioni col pubblico mediante numeri telefonici a chiamata gratuita. Con l'introduzione di cavi di comunicazione bidirezionali, questo tipo di rapporto diretto col pubblico diverrà sempre più di uso comune



per concessione della MTV



NICK DALY



GARY MARSH

Impulsi veloci

Le fibre ottiche, prodotte per la prima volta nel 1966 nei laboratori della Standard Telecommunication ad Harlow, in Gran Bretagna, si basano su un fenomeno di riflessione interna totale. La luce, introdotta da una estremità, scorre all'interno della fibra di vetro senza subire alcun abbassamento di intensità, essendo riflessa dalle pareti interne. Un fenomeno simile accade in un prisma o sulla superficie d'un corso d'acqua. Per ottenere questo risultato, il grado di purezza del vetro usato per la fibra deve avvicinarsi al 100 per 100. La luce introdotta nella fibra viene generalmente prodotta da un laser infrarosso, poiché il segnale deve "pulsare" molto rapidamente. I cavi coassiali, comunemente usati per i collegamenti di un'antenna televisiva, sono composti da un singolo cavetto di rame, circondato da una sottile calza di rame rivestita da un tubo plastico

giunti a vietare espressamente agli operatori dei sistemi via cavo di offrire una qualsiasi forma di servizio telefonico, comprese le video conferenze, anche se previsioni a lungo termine confermano che le reti via cavo, alla fin fine, sostituiranno ogni altro servizio di telecomunicazioni.

Simile è l'esperienza di altri paesi. Negli USA, dove la televisione via cavo è in uso già da tempo, viene fatto tuttora scarso uso di questo sistema come mezzo di comunicazione digitale a due vie. Anche laddove è utilizzato, ciò avviene per scopi abbastanza banali. In un episodio (molto pubblicizzato dai giornali), gli spettatori di una rappresentazione dell'*Amleto* di Shakespeare furono invitati ad esprimere, attraverso una rete via cavo, un giudizio sull'interpretazione degli attori. La maggior parte degli spettatori votò per l'interruzione della rappresentazione!

In Scandinavia ed in Olanda c'è stato un notevole sviluppo di comunità collegate mediante una rete di home computer. L'invio locale di posta elettronica e reti di computer che permettono di organizzare la sorveglianza dei bambini sono oggi molto usate in questi paesi; alcune comunità tengono anche referendum autogestiti su temi di interesse locale. Questi vantaggi di carattere sociale sono stati purtroppo largamente trascurati in quasi tutte le altre nazioni europee. Quando verranno messe in opera delle reti interattive via cavo, le organizzazioni che forniscono servizi (banche o gli stessi enti preposti alle telecomunicazioni) saranno incentivate a distribuire terminali a basso costo, o anche gratuitamente. Ciò si è già verificato in alcune zone della Francia, dove la rete telefonica nazionale ha sostituito i tradizionali elenchi telefonici con un servizio del tipo viewdata.



© BBC STILLS

Un'immagine vale mille parole?

È difficile stabilire il numero di bit necessari per creare un'immagine televisiva a colori ma, considerando che un'ampiezza di banda di 9 MHz permette di trasmettere quattro milioni e mezzo di baud e che ogni immagine completa viene trasmessa 25 volte al secondo, un semplice calcolo aritmetico ci dà 180000 bit per inquadratura. Mille parole, d'altra parte, utilizzano circa 60000 bit

Memorie di oggi, memorie di domani

I primi tipi di memorie per computer ed i più recenti sviluppi nel campo

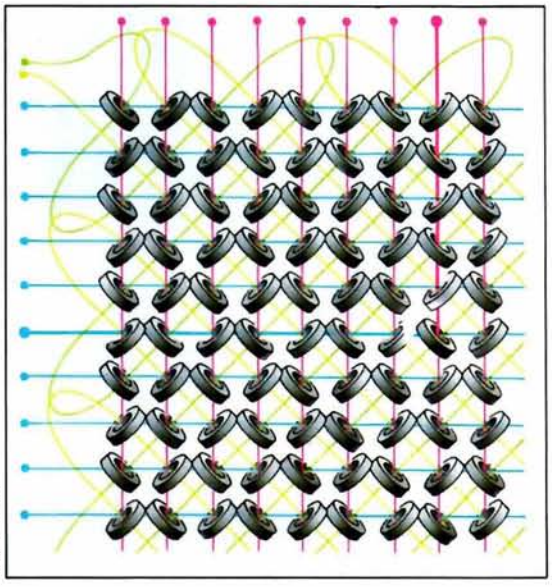
Ogni computer necessita di una certa capacità d'immagazzinare informazioni perché, anche nelle operazioni più semplici, è necessario memorizzare, magari, un risultato intermedio, da usare nella successiva fase di calcolo. Fin dalla nascita dei computer, le dimensioni della memoria sono costantemente aumentate e questo ha permesso di affrontare problemi di più vasta portata, precedentemente insolubili.

Gli scienziati, già nei primi studi sull'elettricità, hanno cercato il modo di utilizzarla per immagazzinare informazioni. L'elettricità può essere pensata o come un flusso d'elettroni o come un'onda in movimento ma, in entrambi gli aspetti, la caratteristica predominante è il moto. L'impossibilità di trattenere qualche cosa che per sua natura è sempre in movimento, ha portato all'adozione di metodi indiretti di memorizzazione. Una batteria, ad esempio, immagazzina energia in forma chimica.

Durante la II Guerra Mondiale si fecero studi approfonditi sul radar per riuscire a separare i segnali riflessi da un oggetto in movimento da quelli rimandati da oggetti fissi come gli alberi. Fu inventato un apparecchio chiamato "linea di ritardo al mercurio", in grado d'immagazzinare, temporaneamente, gli impulsi trasportati dalle onde radio, permettendo di confrontarli con le onde riflesse captate nei passaggi successivi del radar. Questo procedimento consentiva l'eliminazione delle immagini statiche.

I NUCLEI DELLA MEMORIA

I "tori" (minuscoli anelli di ferro) vengono attraversati da una rete di sottilissimi fili elettrici, disposti a formare una matrice. Ogni anello può essere magnetizzato applicando una corrente al filo orizzontale e verticale che lo attraversano. La linea di lettura (il terzo filo) individua i cambiamenti di campo magnetico avvenuti in un anello, che indicano se è stato memorizzato un 1 o uno 0



KEVIN JONES

Una linea di ritardo al mercurio si compone di un tubo di vetro, lungo un metro, riempito di mercurio e con un cristallo di quarzo ad ogni estremità. Se si applica un segnale elettrico ad un'estremità, il cristallo di quarzo entra in vibrazione e produce un'onda fisica che si propaga verso l'altra estremità del tubo dove viene "rivelata" dal secondo cristallo che la riconverte in un segnale elettrico. L'onda impiega circa 660 microsecondi per attraversare il tubo ma, facendo rimbalzare indietro il segnale attraverso il mercurio, gli impulsi possono essere immagazzinati per diversi minuti prima che il segnale subisca una distorsione.

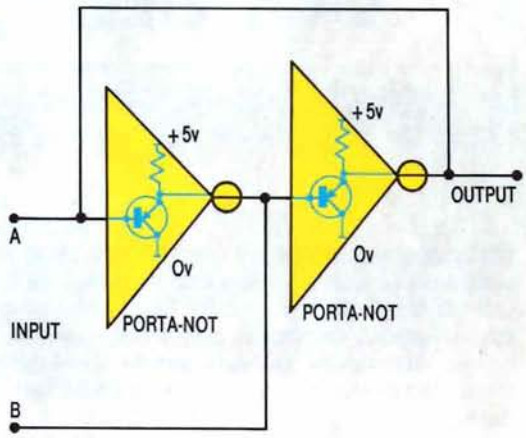
Con un temporizzatore, che regoli il treno d'impulsi a 500.000 cicli al secondo, è possibile immagazzinare 350 bit in un tubo d'un metro. Le dimensioni della memoria potrebbero essere aumentate allungando il tubo ma questo andrebbe a discapito della velocità d'accesso. Apparecchi di questo genere erano ingombranti, costosi e richiedevano tecnologie di precisione.

Il professor F. C. Williams, dell'Università di Manchester (GB), inventò un secondo sistema per la conservazione dei dati basata sul possibile uso dell'energia statica che si genera sulla faccia interna di un tubo a raggi catodici (uno schermo televisivo) come mezzo d'immagazzinamento. Un cannone elettronico genera una configurazione di cariche statiche sullo schermo, rilevabile da una griglia metallica posta in prossimità della superficie esterna dello schermo. Seguendo questo metodo, nel 1947 era possibile memorizzare, per un certo numero di ore, 2048 bit di informazioni su di un singolo schermo. Nonostante la velocità d'accesso fosse alta, le cariche statiche dovevano essere "rinfrescate" ogni 30 microsecondi, altrimenti sarebbero diminuite d'intensità fino a scomparire.

L'uso del nastro magnetico fu sperimentato per la prima volta con il LEO in Gran Bretagna (vedere a pag. 320) e sull'UNIVAC (Universal Automatic Computer) negli USA verso la fine degli anni '40. Questa fu la prima tecnica che permise di memorizzare grandi quantità d'informazioni a basso costo e con affidabilità. Otto bit, organizzati in una precisa sequenza, venivano memorizzati sul nastro, con una densità di 2360 sequenze al centimetro. Con nastri lunghi una sessantina di metri fu possibile realizzare memorie permanenti capaci di almeno un megabyte. Tuttavia, anche con le più veloci unità di lettura,

il tempo di accesso ai dati posti nella parte centrale del nastro era di qualche secondo. Di conseguenza, questo metodo fu usato principalmente per i file sequenziali impiegati, ad esempio, nella gestione dei libri paga.
La lettura di un nastro poteva essere velocizzata dotando di un cuscinetto d'aria la testina di lettura che demodulava il segnale magnetico: questo accorgimento riduceva anche l'usura per attrito e le lacerazioni del nastro.
Una memoria di grande capacità, ad alta velocità d'accesso e basso costo per bit fu progettata al Massachussets Institute of Technology intorno al 1950. Il "cuore" della memoria usava dei "tori" (anelli) d'acciaio inseriti in una griglia di sottilissimi fili elettrici (vedere illustrazione a pag. 304). Inviando una corrente su un filo collegato ad un anello, il metallo veniva magnetizzato. Ci sono due possibili direzioni del campo magnetico dell'anello ed esse vengono utilizzate per rappresentare i due stati binari, 0 ed 1. La direzione del campo magnetico può essere invertita applicando un'appropriata corrente. Ma, poiché la corrente elettrica origina un campo magnetico, si verifica anche l'evento opposto: quando la direzione del campo viene invertita, si produce una piccola, ma rilevabile, corrente elettrica. Il campo si inverte solo quando viene fornita una corrente d'intensità superiore ad una data soglia. Applicando quindi ad un filo verticale della griglia una corrente appena superiore alla metà del valore soglia e ad uno orizzontale l'altra metà, soltanto il toro collegato ad entrambi i fili riceverà una corrente sufficiente ad invertire il suo campo magnetico.
Il diametro dei singoli anelli è stato successivamente ridotto da circa 4 millimetri ad alcuni centesimi di millimetro riducendo così anche l'energia necessaria ad inserire dati in una cella di memoria. L'informazione presente in una data cella viene letta cercando di invertire il suo campo magnetico. Analizzando l'eventuale corrente indotta, è possibile dedurne lo stato magnetico originario. Tuttavia, questo metodo di lettura è "distruttivo", e, dopo ogni lettura, i dati devono essere nuovamente memorizzati.
Con l'avvento dei circuiti a larga scala d'integrazione, è stato recuperato uno dei primi metodi di memorizzazione dati, usato sin dai tempi dell'ENIAC, cioè il "flip-flop". Questo è un particolare circuito, detto "bistabile", poiché si stabilizza in uno dei due possibili stati (0 od 1). Si compone di due interruttori elettronici collegati in serie e l'uscita di ciascuno dei due è collegata con l'ingresso dell'altro. È quindi possibile intrappolare un segnale nel flip-flop finché se ne ha bisogno.
Ciascun bit necessita di due interruttori ed i primi flip-flop a valvole erano costosi, poco affidabili e si bruciavano facilmente; tuttavia, con l'introduzione dei circuiti integrati, composti da centinaia di migliaia di interruttori a transistor inseriti in un singolo chip, il flip-flop è divenuto nuovamente un circuito di grande importanza.

FLIP-FLOP



I flip-flop vengono impiegati nella maggior parte delle memorie a semiconduttori, ed il loro funzionamento si basa su due invertitori (o porte NOT). Nell'illustrazione, i circuiti a transistori equivalenti sono rappresentati in blu. L'uscita di ciascun invertitore è collegata con l'ingresso dell'altro. Se A è posto a livello logico 1, l'uscita sarà 1. Se l'ingresso B è posto a 1, allora l'uscita sarà 0. Questo circuito viene spesso chiamato "bistabile", in quanto si stabilizza su una delle due condizioni logiche possibili

KEVIN JONES

I chip di RAM "statica", utilizzati in quasi tutti i primi piccoli computer, contengono migliaia di circuiti bistabili, uno per ogni bit. La maggior parte dei computer più recenti, invece, utilizza RAM "dinamiche". La differenza è scarsamente apprezzabile dall'utente, ma una RAM dinamica è più veloce e, soprattutto, meno costosa; è paragonabile ad una serie di condensatori, ciascuno dei quali può immagazzinare una carica elettrica. L'unico inconveniente è che le cariche così immagazzinate tendono a disperdersi, per cui risulta necessario "rinfrescare" le memorie migliaia di volte al secondo, operazione svolta da speciali circuiti incorporati nel chip.
Il più recente sviluppo, in materia di memorie, è costituito dalle memorie a "bolle" e a laser. Queste ultime sono simili ai dischi musicali, ma vengono lette da un fascio di luce laser. In una memoria a bolle vengono create, elettrostaticamente, delle piccole "protuberanze" o bolle all'interno di un chip. La presenza o l'assenza di una bolla rappresenta, rispettivamente, le condizioni binarie 1 o 0. La densità delle bolle, potenzialmente, può essere talmente alta, che questo genere di memorie potrebbe, in un prossimo futuro, sostituire le attuali cassette ed i sistemi a disco.
Tra non molto saranno disponibili anche memorie "criogeniche". A temperature vicine allo zero assoluto la resistenza elettrica in un mezzo tende a scomparire per dar luogo al fenomeno chiamato "superconduttività": diventa allora possibile immagazzinare una carica equivalente a quella di un singolo elettrone. Le cariche su cui operano i più moderni tipi di memorie a chip sono molto piccole, tuttavia ciascuna di esse è equivalente a qualche milione di elettroni.
Grazie alla superconduttività ottenuta raffreddando i materiali che compongono queste memorie fino a temperature criogeniche (in bagni di elio liquido), la capacità e la velocità delle memorie supereranno di gran lunga qualsiasi risultato ottenuto con i sistemi attuali.

Biotecnologia

Sebbene la microelettronica abbia raggiunto un incredibile livello di miniaturizzazione, ogni operazione eseguita in un semiconduttore implica ancora un flusso di molte migliaia di elettroni. Gli scienziati stanno già sperimentando memorie che lavorano a livello molecolare con singoli elettroni; questo tipo di progetti si avvicina più alla biotecnologia che alla microelettronica. È noto, ad esempio, che nella struttura di una molecola di DNA è immagazzinato il codice genetico che controlla lo sviluppo del nostro corpo. Dovranno comunque passare molti anni prima che l'uomo sia capace di riprodurre questo meccanismo biologico in forme utilizzabili



I "Bulletin Board"

Le "bacheche" computerizzate per lo scambio di notizie tra computer

Un home computer, di per sé, ha capacità piuttosto limitate di comunicazione, ma tramite il telefono è possibile farlo "dialogare" con un'ampia gamma di altri computer, siano questi parte di un potente centro di elaborazione universitaria o di una grande banca dati pubblica (come il Prestel) oppure, più semplicemente, gli home computer dei vicini di casa.

ELENCO TELEFONICO PER COMPUTER
(tutti a 300 baud salvo diversa indicazione)

Servizi di "bacheca"	orario	prefisso - numero
Forum 80 Hull	1900-2300 Lu - Ve	0482 859169
CBBS North East	1430-0100	0207 543555
Liverpool Mailbox	24 ore al giorno	051 4288924
TBBS Blandford	24 ore al giorno	02584 54494
TBBS London	0900-0100	01 348 9400
Mailbox80 Stourport	1800-0800	0384 635336

Servizi commerciali	orario	prefisso - numero
Rewtel	24 ore	0227 232628
Maptel	24 ore	0702 552941
Distel	24 ore	01-679 1888
Estelle	0700-1800	0279 443511
Estelle (1200/75)	0700-1800	0279 441188
Prestel (1200/75)	24 ore	dipende dall'area
Micronet 800	24 ore	Prestel page *800

Usando il "Bulletin Board Services" (BBS), una specie di "bacheca" computerizzata, è possibile lasciare messaggi per altri utenti di home computer, pubblicizzare la vendita di beni o anche scambiare programmi. Un BBS è, in effetti, molto simile alle bacheche presenti, ad esempio, negli atri di molte facoltà universitarie, dove vengono affissi avvisi d'ogni genere. I servizi di "bacheca computerizzata" vengono, almeno per adesso, gestiti in gran parte da volontari, che, impiegando un normale home computer dotato di notevole memoria di massa (ad esempio, un disco fisso), offrono a chiunque possieda un computer ed un modem, un servizio di consultazione. La distanza tra gli apparecchi non è un problema: è possibile collegarsi con computer della stessa città, con altri al di fuori della regione o, addirittura, anche all'estero. L'unico ostacolo rimane, ovviamente, la bolletta telefonica!
Affinché il nostro home computer possa "conversare" via telefono con uno di questi centri, è necessario l'acquisto di un apposito dispositivo, che permetta il collegamento. Un apparecchio di questo tipo è il modem (vedere a pag. 108), che, solitamente, sfrutta la porta seriale (RS232) del computer. Alcuni home computer, però (quali il Vic-20 e lo Spectrum), non sono dotati di una porta seriale ed occorre quindi acquistare anche una scheda d'interfaccia seriale da applicare al computer. Il modem viene collegato alla linea telefonica direttamente (mediante una presa esterna al telefono) oppure usando un accoppiatore acustico. Le due velocità di trasmissione più comuni sono: 300 baud e 1200/75 baud. Servizi commerciali quali il Prestel usano una più elevata velocità (tipicamente: 1200/75, ossia 1200 baud in direzione dell'utente e 75 da utente a unità centrale), mentre il BBS (in quanto utilizzato soprattutto da home computer) usa 300 baud. Naturalmente è preferibile acquistare un modem che permetta di scegliere tra le due velocità. Recentemente il prezzo dei modem è calato notevolmente ed è possibile acquistarne uno da 300 baud, con alimentazione a pile, per una cifra che si aggira sulle 200.000 lire.
Per far funzionare un modem è inoltre necessario disporre di un software che permetta al computer di svolgere le funzioni di terminale. In commercio si trovano due tipi di software di questo genere: il "dumpb" e lo "smart". Un terminale smart può caricare programmi e ricevere messaggi provenienti da altri computer. Un terminale dumpb non è in grado di svolgere queste operazioni, ma è facile da far funzionare ed è forse maggiormente adatto alle esigenze degli utenti meno esperti. Questi programmi sono adatti sia agli home computer più "vecchi", come l'Apple II ed il Tandy TRS80, che ad alcuni dei modelli più nuovi: BTERM è infatti un programma per il Modello B della BBC. In certi casi, addirittura, un software di questo tipo è fornito gratuitamente. Comunque, non è molto complicato approntare un proprio programma di comunicazione, qualora sia difficile reperirlo già pronto. Lo scopo del programma è quello di inviare i caratteri battuti sulla tastiera alla porta RS232 e di visualizzare sullo schermo i dati ricevuti attraverso la stessa porta.
Per comprendere il funzionamento dei modem, esaminiamo il sistema di comunicazione detto BBS. Innanzi tutto è necessario conoscere il numero telefonico e le caratteristiche tecniche (come la velocità baud) del computer che si vuol chiamare. Dopo aver caricato il software di comunicazione, si compone il numero telefonico. Se il computer chiamato è "libero", si udrà una nota acuta sull'auricolare: questo è il suono della "por-

TELECOM GOLD

Posta elettronica

Il Telecom Gold è un sistema di posta elettronica offerto dalla British Telecom, una versione molto sofisticata del BBS, indirizzata principalmente al mercato commerciale. Gli utenti registrati possono affittare una casella postale nella quale altri abbonati possono depositare messaggi, che verranno successivamente letti dal computer dell'intestatario della casella. Usando il Telecom Gold è possibile inserirsi nelle reti telefoniche di altri paesi con il vantaggio che tutte le differenze di carattere tecnico tra i vari utenti vengono automaticamente uniformate

per concessione della BRITISH TELECOM



La bacheca computerizzata
Una delle più divertenti applicazioni dei modem è quella che permette di consultare una "bacheca" elettronica. È questa una versione computerizzata della tradizionale bacheca da parete. Su di essa è possibile "appuntare" messaggi a disposizione di tutti o solo di quelli che conoscono la giusta parola chiave. Vi si possono annunciare riunioni o vendite di componenti per computer di seconda mano. È anche possibile scambiare con altri utenti giochi o altri programmi

tante". Se si sta usando un accoppiatore acustico, dovremo appoggiare il ricevitore sulle apposite coppette gommate. Con un modem del tipo a collegamento diretto si dovrà invece abbassare l'interruttore che porta la scritta LINE o DATA e che sostituisce il ricevitore. Entrambe le procedure permettono di collegare il computer. Per prima cosa apparirà sullo schermo un qualche tipo di saluto, inviato dal computer posto all'altro capo della linea. Quindi apparirà una richiesta d'identificazione o parola chiave. Se il servizio è privato e non si è abbonati, molto probabilmente non si procederà oltre, a meno che non si abbia buona intuizione o molta pazienza! Un sistema privato, quindi, permetterà un accesso limitato agli "ospiti" che non fanno parte degli abbonati, cosicché vale la pena di tentare con poche parole come NEWUSER, GUEST o HELP.

Il BBS invece è aperto a tutti ed è gratuito (salvo, ovviamente, la chiamata telefonica). Si inserisce quindi, come richiesto, il proprio nome e la città, e, una volta accettati, si dovranno comunicare le caratteristiche del nostro computer, ad esempio le dimensioni dello schermo o la marca. Questi dati permettono all'unità di elaborazione centrale di identificarci nelle successive chiamate e di mettere il sistema in condizione di dialogare con il nostro computer.

Vengono quindi fornite alcune informazioni di sistema (come gli orari del servizio, le caratteristiche tecniche e la durata massima, in genere 30 minuti, della nostra chiamata). A questo punto viene presentato il menu principale, che consiste in una mezza dozzina di voci tra le quali è possibile selezionare quella voluta con la semplice pressione di un tasto. Ad esempio, per registrarci come "Nuovo Utente", dovremo battere la lettera N; per chiudere la chiamata, batteremo la lettera G per "Goodbye". Dal menu principale si possono selezionare dei sottomenu, sino a raggiungere l'argomento desiderato. La struttura del BBS è paragonabile ad un albero: il menu principale corrisponde al tronco da cui si originano i rami (i sotto menu). Anche il Prestel e la maggior parte delle altre banche dati hanno una struttura simile.

La sezione "Nuovo Utente" è per coloro che vogliono iscriversi al sistema BBS: vengono richiesti il nome e l'indirizzo ed è possibile scegliere una propria parola chiave da usare nelle chiamate successive. La voce "Informazioni" fornisce dettagliate notizie tecniche sul sistema, mentre la sezione "Servizi" comunica il tempo di durata della chiamata e fornisce chiarimenti su tutti gli altri servizi offerti. La sezione "Comunicati" contiene avvisi pubblici inseriti nel sistema da altri utenti ed è a disposizione di tutti. La sezione "Messaggi" permette di leggere messaggi a carattere privato inviati da altri utenti. È quindi possibile mandare messaggi privati a uno o più utenti che hanno qualcosa in comune (per esempio, a tutti i possessori di un Modello B della BBC). Queste caratteristiche rappresentano forse l'aspetto più interessante di questi servizi: ad esse è probabilmente dovuto il crescente interesse che tali sistemi suscitano.

Un servizio leggermente diverso, chiamato Rewtel, dispone di una banca dati specializzata per componenti elettroniche e relative informazioni: permette ai soli abbonati di acquistare direttamente da casa uno o più articoli. Diversamente dal BBS, occorre inserire delle parole chiave per giungere all'argomento che ci interessa. Ad esempio, se si ha bisogno di un aiuto mentre ci troviamo nella sezione "Acquisti", dovremo battere le parole HELP REWSHOP. Il Rewtel possiede anche caratteristiche simili a quelle del BBS: inserendo la parola chiave CHALK è infatti possibile inviare un messaggio. I non abbonati possono usare questo servizio ed il sistema sarà a loro disposizione per otto minuti. Il Distel ed il Maptelm sono due banche dati simili, principalmente destinate ad acquisti di materiale elettronico e attrezzature per computer. Il vantaggio offerto da questi sistemi commerciali è che essi sono in funzione 24 ore su 24 e non sono così frequentemente occupati quanto il BBS.

Fino a poco tempo fa solo le grosse unità di elaborazione utilizzavano le reti telefoniche. Solo recentemente, con la sensibile diminuzione del prezzo dei modem e dei computer, questo tipo di comunicazione si è diffuso enormemente. Il modem sta diventando un accessorio indispensabile per i computer, quanto la stampante e le memorie di massa.

Un redattore ineccepibile

I diversi tipi di programmi per la stesura di testi al computer

Nella redazione di un testo, sia esso una lettera o il listato di un programma, è facilissimo commettere errori di battitura. Ciò accade con le normali macchine da scrivere, ma, specialmente se l'operatore non è un buon dattilografo, anche usando la tastiera di un computer. Si è reso quindi necessario introdurre un qualche strumento che faciliti tutte queste operazioni: i cosiddetti programmi "editor" (letteralmente: "redattori").

A seconda del tipo di computer, l'editor può essere un programma a sé stante, oppure essere interno al sistema operativo contenuto nella memoria ROM.

Una forma più sofisticata di editor è il "word processor", nel quale le funzioni sono appositamente progettate per facilitare al massimo l'elaborazione di testi, offrendo, tra l'altro, la possibilità di cancellare frasi, di spostare interi paragrafi da una parte all'altra del testo, di cambiare una parola o una frase con una nuova, tutte le volte che essa compare nel testo, di ottenere in modo automatico l'impaginazione del testo, ecc.

Esistono due tipi di editor: lo "screen editor" ed il "line editor". Il primo è considerevolmente più flessibile e facile da usare, ma il secondo, più economico sia in termini di memoria che di denaro, è quello maggiormente diffuso sugli home computer. I primi line editor furono sviluppati quando ancora tutte le operazioni di elaborazione venivano effettuate impiegando telescriventi o terminali collegati ad un computer centralizzato. Le telescriventi possedevano una memoria d'accumulo limitata, che poteva contenere solo una linea di 80 caratteri (ossia 80 byte). Nei "line editor", il programmatore, dovendo apportare una modifica ad una delle linee di programma, è costretto a riscrivere la linea per intero. In alcuni casi, è disponibile uno speciale comando per modificare il contenuto di una particolare linea: EDIT 120, ad esempio, visualizza sullo schermo la linea 120, permettendo di eseguire, sia pure con certe limitazioni, le necessarie correzioni.

Gli editor di molti home computer funzionano ancora come se avessero una memoria di accumulo di una sola linea, mentre in effetti, l'intero contenuto del video è registrato in memoria: ogni locazione di carattere corrisponde ad un byte di memoria. Gli screen editor, invece, sono molto più efficienti: in essi è possibile spostare con facilità il cursore sul testo visualizzato, apportare le modifiche occorrenti e memorizzare il nuovo testo.

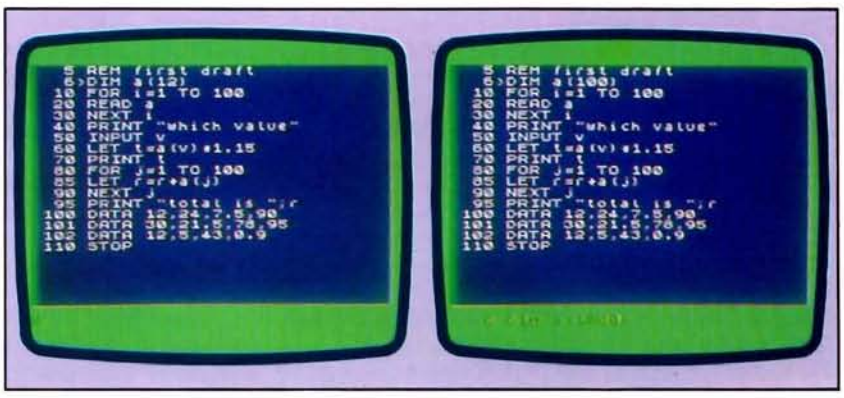
Per quanto concerne la programmazione, gli editor possono essere incorporati nel linguaggio adottato (come generalmente accade negli interpreti BASIC) oppure essere programmi indipendenti, il cui risultato viene successivamente elaborato da un compilatore. Nel primo caso, ogni volta che viene battuto il tasto RETURN, l'editor trasferisce l'intera linea sulla quale si trova il cursore all'interprete. Se essa contiene solamente dei comandi, viene eseguita immediatamente. Se, invece, è preceduta da un numero di linea, essa viene opportunamente inserita nel listato del programma, in attesa che esso venga eseguito integralmente. Spesso la tastiera è dotata di quattro tasti (contrassegnati da altrettante frecce), che permettono di spostare il cursore in qualsiasi punto del testo visualizzato e quindi di inserire, cancellare o riscrivere qualsiasi carattere.

Per una maggiore velocità di esecuzione, gli screen editor vengono, in genere, scritti in codice macchina. I migliori permettono lo scorrimento di un listato sia verso l'alto che verso il basso, l'inserimento e la cancellazione di intere linee di programma come di singoli caratteri. Alcuni possiedono persino comandi simili a quelli dei word processor, che permettono di trovare e sostituire una particolare stringa di caratteri tutte le volte che essa compare nel testo. Gli editor stanno diventando sempre più sofisticati e facili da usare ad ogni nuova generazione di computer. Con l'introduzione dei "topi" (vedere a pag. 296) e di software che imita il modo di procedere tipicamente umano nel selezionare e rimettere assieme le parti di un testo, si ridurrà sempre più il tempo necessario alla redazione di un documento ed alla sua stesura definitiva.

Le facilitazioni offerte dall'editor di un computer possono far risparmiare molto tempo nello sviluppo di un lungo programma. Occorre comunque sottolineare che un buon programmatore verifica sempre il funzionamento del proprio programma "sulla carta", prima di batterne il listato sul computer.

Linea dopo linea

Il funzionamento dell'editor di cui è dotato lo Spectrum della Sinclair è una via di mezzo tra quello dei "line editor" e quello degli "screen editor". Per apportare modifiche ad una particolare linea di programma, si può spostare il cursore (rappresentato col simbolo ">") sulla linea voluta, usando gli appositi tasti per il movimento del cursore



IAN McKINNELL

Lo Sharp MZ-711

Anatomia di un microcomputer a diffusione europea: tecnologie a confronto

Lo Sharp MZ-711 è un apparecchio interessante, con alcune caratteristiche piuttosto inconsuete. La sua costruzione rispecchia l'elevato standard nipponico nella progettazione in generale. I colori, sullo schermo, sono buoni, molto brillanti ed omogenei, e, a parte una lieve sfuocatura quando si impiegano particolari combinazioni di colori, i testi riprodotti sono apprezzabili per chiarezza.
Alcuni aspetti di questo apparecchio, però, lasciano un poco perplessi. La tastiera, ad esempio, sarebbe assai soddisfacente se il modo di funzionamento non fosse l'esatto opposto di quello che normalmente ci aspetteremmo: le lettere minuscole si ottengono premendo i tasti SHIFT, mentre normalmente avviene il contrario. Sebbene ciò sia indiscutibilmente utile per la programmazione BASIC, tuttavia, nella scrittura di testi ciò può confondere. Il normale modo di funzionamento si può ottenere premendo contemporaneamente i tasti CONTROL ed E. Per ripristinare il modo originario, invece, si premono i tasti CONTROL ed F.

L'unità di registrazione a microcassette per lo Sharp MZ-711 (venduta separatamente) viene alloggiata in un'apposita nicchia ricavata nell'involucro del computer, rispettando la linea estetica dell'insieme. Pur trattandosi di un "optional", l'unità risulta indispensabile per conservare dati e programmi. Il sistema è lento, ma affidabile e tutti i nastri sembrano funzionare regolarmente al primo tentativo. Tuttavia il sistema operativo che gestisce l'unità a nastri sembra avere diversi punti deboli: è possibile utilizzare solo nomi molto rudimentali per i file archiviati, non c'è alcun controllo del motore del registratore, che, invece, poteva essere facilmente previsto. Ciò limita fortemente l'uso dei nastri, al punto che conviene archiviare un solo file per ciascun lato della cassetta. I limiti di questo sistema sono sorprendenti, considerata l'esperienza della Sharp, particolarmente nel campo dell'alta fedeltà, nel quale le piastre di registrazione controllate da soleniodi sono diffusissime.
La stampante/plotter incorporata (anch'essa



CHRIS STEVENS

Tastiera

La tastiera dello Sharp MZ-711 dà un aspetto professionale alla macchina. I tasti sono spaziati correttamente, hanno una "corsa" completa, sono di dimensioni adeguate e adatti alla battitura di testi. Cinque tasti di funzione programmabili, da usare assieme ai tasti SHIFT, permettono di utilizzare con facilità dieci diverse funzioni. I simboli grafici sono sistemati in sequenza logica, mentre gli altri tasti sono disposti nel formato QWERTY

un "extra", che si inserisce nel contenitore dell'apparecchio) possiede un meccanismo di scrittura a penne colorate estremamente delicato e vulnerabile. Le penne dovrebbero essere estratte dal supporto e protette in un'apposita custodia, allorché si prevede di non usare la stampante per un lungo periodo di tempo. Infatti, l'inchiostro potrebbe seccarsi, rendendo la qualità di stampa piuttosto scadente.

L'interprete BASIC viene caricato da nastro (non è contenuto nella memoria ROM) e ciò offre l'innegabile vantaggio di poter usare altri linguaggi.

La documentazione tecnica, di livello decisamente elevato, è corredata da mappe di memoria e diagrammi di circuiti, da un listato in linguaggio Assembler del software di sistema incorporato e da altri dettagli tecnici. Anche se mancano i diagrammi di sincronizzazione, la documentazione risulta migliore di quella che accompagna la maggior parte degli altri computer in commercio.

I fabbricanti dello Sharp MZ-711 dimostrano un'incomprensibile tendenza a sottovalutare alcune caratteristiche più inconsuete dell'apparecchio. Un esempio è rappresentato dal programma dimostrativo: mentre nel manuale si legge che la macchina è capace di produrre 8 colori, il programma è in grado di produrre una gamma di colori ben 8 volte maggiore. Il programma, mediante l'istruzione POKE, deposita opportuni valori nella locazione di memoria 40960 e pare che ciò rappresenti la chiave del mistero. L'effettiva gamma di colori è pertanto difficile da determinare con esattezza, ma può essere divertente sbizzarrirsi alla ricerca di nuovi effetti.



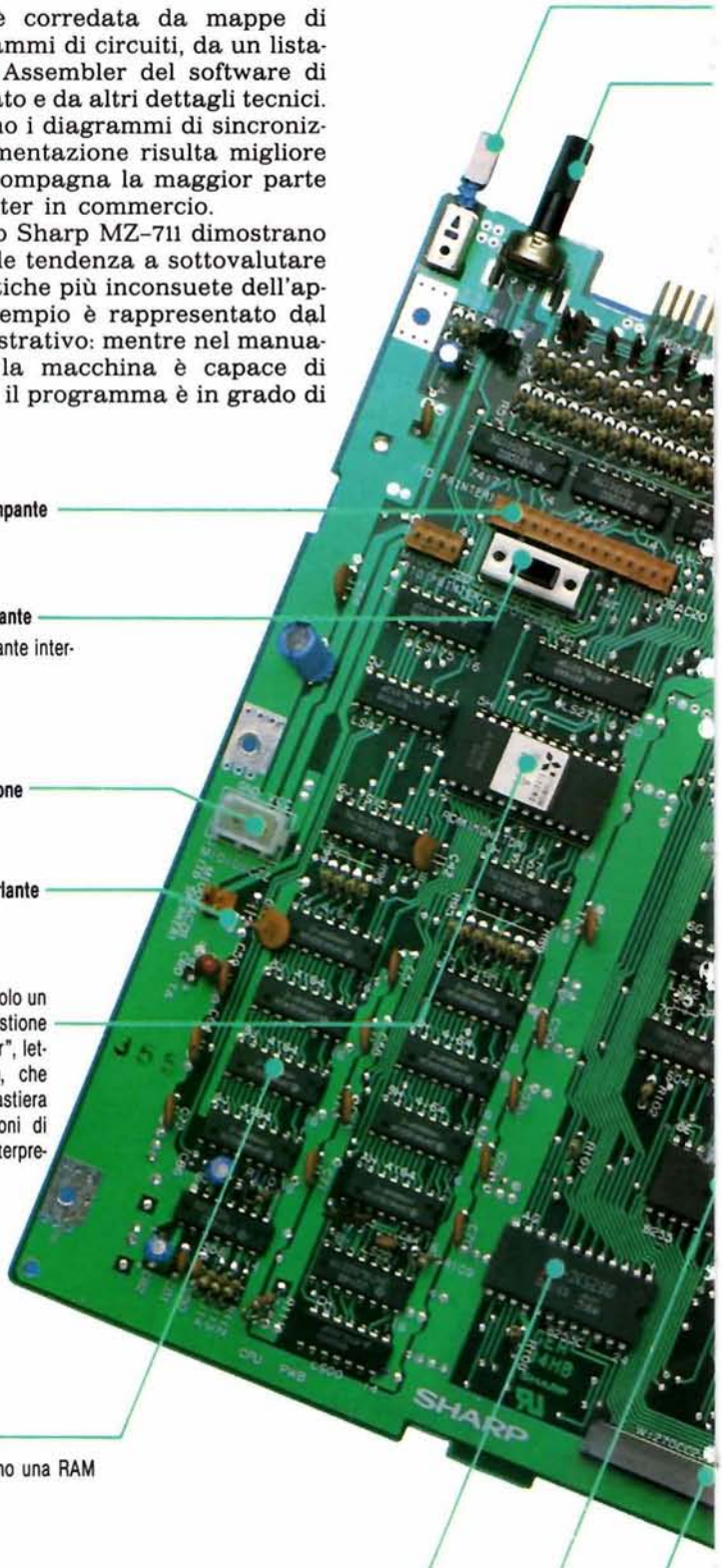
Registratore a cassette

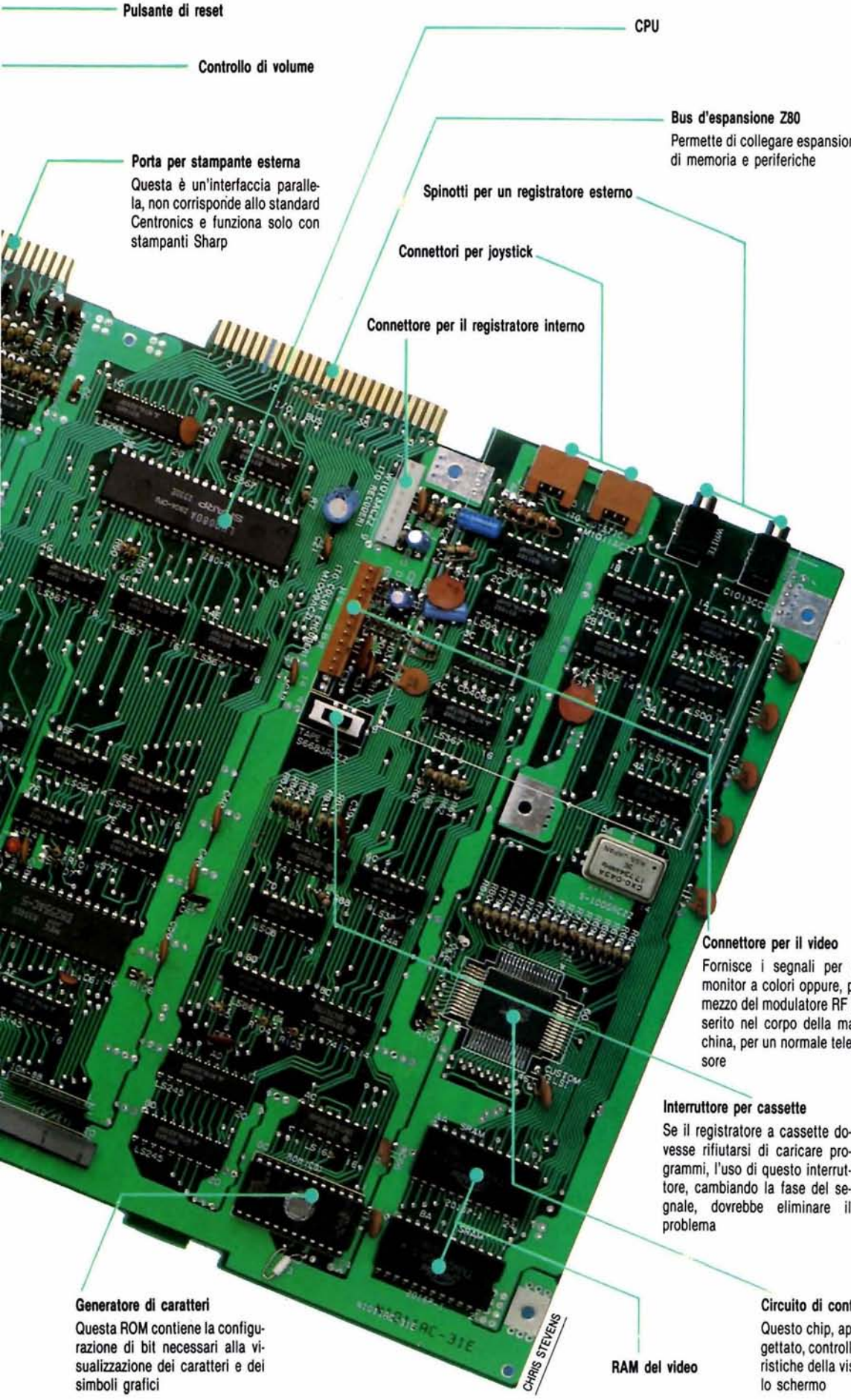
Il registratore a cassette dell'MZ-711 è un optional extra nonostante ne sia previsto l'inserimento nel corpo principale della macchina. Il sistema operativo non ha modo, tuttavia, di controllare il motore e questo fatto ne limita le applicazioni



Stampante/plotter

Come il registratore a cassette, la stampante/plotter è un extra da inserire nel corpo principale. Usa quattro penne a sfera miniaturizzate per stampare testi o grafica su un nastro di carta largo 11,5 cm. Tre modi di funzionamento permettono altrettante diverse dimensioni del testo (a 26, 40 o 80 colonne per riga)





SHARP MZ-711

DIMENSIONI

mm. 440 x 305 x 86

PESO

—

CPU

Z80A

CLOCK

4 MHz

MEMORIA

64 Kbyte di RAM, 12 Kbyte di ROM

VIDEO

24 linee di 40 caratteri, 50 x 80 pixel in grafica a bassa risoluzione con 8 colori dichiarati (la gamma effettiva sembra essere superiore). 127 caratteri predefiniti, 127 caratteri definibili dall'utente

INTERFACCE

Connettore per TV, per registratore a cassette, per stampante parallela, bus di sistema, porta seriale RS232

LINGUAGGI FORNITI

BASIC (su cassetta)

ALTRI LINGUAGGI DISPONIBILI

FORTH, PASCAL, FORTRAN, ASSEMBLER, DAFMOD TOOLKIT

ACCESSORI COMPRESI

Cavo di connessione per TV e per registratore a cassette, un nastro con il BASIC ed un programma dimostrativo, 2 manuali

TASTIERA

59 tasti formato QWERTY, 5 tasti funzione, 4 per il posizionamento del cursore

DOCUMENTAZIONE

La documentazione è molto completa e ricca di informazioni tecniche, raramente riportate sui manuali di altri apparecchi. Anche dal punto di vista operativo, i manuali sono di buon livello

Semplici suoni...

Effetti musicali sullo Spectrum

Lo Spectrum è uno dei più diffusi home computer e possiede un'eccellente grafica a colori, oltre a notevoli espansioni di memoria a basso costo. Purtroppo, l'economicità dell'apparecchio ha costretto a sacrificare alcune caratteristiche: in particolare, l'insolita tastiera, l'uso di un BASIC non standard e l'assai scarsa possibilità di generare effetti audio. Lo Spectrum, disponendo di un unico oscillatore, produce pertanto effetti musicali qualitativamente insoddisfacenti. È possibile controllare la durata di una nota e la sua altezza, ma non c'è modo di modificarne il tono o l'inviluppo (vedere a pag. 276). Un altro inconveniente è che il piccolissimo altoparlante piezo-elettrico incorporato emette soltanto degli striduli "beep". Per quanto lo Spectrum sia dotato di un segnale d'uscita alternativo che, attraverso le prese "mic" ed "ear", può essere inviato ad un impianto hi-fi esterno, questo sistema pare produrre suoni di qualità insoddisfacente. In compenso, il BASIC dello Spectrum possiede comandi molto semplici (ad esempio, BEEP e PAUSE) che permettono una facile comprensione dei principi che stanno alla base della generazione dei suoni con i computer. La macchina possiede un'estensione di frequenza di addirittura 10 ottave, assai imponente.

Prima di produrre un qualsiasi suono con lo Spectrum, è necessario: o collegare il computer ad un impianto hi-fi, o rendere udibile il "beep" interno mediante il comando:

```
POKE 23609,100
```

Questo comando, aumenta il volume del "click", udibile quando viene battuto un tasto.

Controllo del suono

Per insegnare al computer ad emettere un dato suono, occorre usare il comando BEEP nel seguente modo:

```
BEEP d,n
```

dove "d" rappresenta la durata di un suono ed "n" la sua altezza. Il tempo di durata sarà compreso tra 0,00125 e 10 secondi. L'altezza del suono viene espressa come il numero di semitoni a partire dal do medio (considerato come valore 0) ed è compresa tra —60 e 69. Ad esempio, la seguente linea di programma fa suonare la nota la a 440Hz, che è il nono semitono a partire dal do medio, per la durata di un secondo:

```
BEEP 9,5
```

Per far suonare un'intera sequenza di suoni separati tra loro da precise pause, possiamo usare il comando PAUSE:

```
PAUSE ms
```

dove "ms" rappresenta il tempo espresso in unità di 0,001 secondi (millisecondi). Per emettere tutti i suoni della scala in chiave di do

Immagini primarie

Le capacità grafiche del Vic-20

Come altri computer della Commodore, quali il Commodore 64 ed il PET, il Vic-20 è un apparecchio ben costruito, ma il set di istruzioni del BASIC non dispone di comandi grafici speciali: è quindi necessario avere un'ottima conoscenza del funzionamento interno della macchina oppure acquistare uno dei programmi progettati appositamente per facilitare la creazione di immagini grafiche. Comunque, gli apparecchi Commodore sono dotati di un'ampia gamma di caratteri speciali che, con un po' d'ingegnosità, possono essere impiegati, come in un gioco ad incastri, per ottenere interessanti risultati.
Il Vic-20 dispone di sedici colori e ogni quadrato sullo schermo, equivalente ad un carattere, può contenere fino a quattro colori. Lo schermo possiede 23 righe di 22 colonne ciascuna ed ogni cella di carattere si compone di 8×8 pixel: i caratteri possono anche essere visualizzati con un formato di 16×8. Sul Vic-20 è possibile ottenere una grafica ad alta risoluzione, ma un utente medio troverà una certa difficoltà nel suo impiego.

Capacità in bassa risoluzione

Sono disponibili caratteri alfabetici maiuscoli e minuscoli e più di 60 caratteri grafici speciali. Sulla maggior parte dei tasti del Vic-20, in alto, si trovano due piccoli quadrati, ciascuno dei quali ha una particolare configurazione (simboli delle carte da gioco, piccoli riquadri quadrettati, cerchi e numerosi altri simboli). Ogni carattere può comparire sul video in modo inverso (cioè nero su

maggiore (do, re, mi, fa, sol, la, si, do) a partire dal do centrale, composta da note della durata di mezzo secondo e pause tra i suoni di un quarto di secondo, occorre usare il seguente formato:

```
10 FOR I = 1 TO 8
20 READ N
30 BEEP .5,N
40 PAUSE 250
50 NEXT I
60 DATA 0,2,4,5
70 DATA F,9,11,12
```

Questo programma rappresenta un ottimo esempio del formato da impiegare per suonare brani musicali. Un'ottava si estende dalla nota fondamentale (in questo caso il do centrale) fino alla nota successiva con lo stesso nome (il do immediatamente successivo in ordine ascendente), attraversando 12 semitoni.



Do maggiore

Scale e pianoforti

Impiegando l'istruzione INKEY$, la tastiera dello Spectrum può essere usata in modo simile a quella di un pianoforte:

```
10 REM ★★★★★★★★★★★
20 REM ★OTTAVA DI PIANOFORTE★
30 REM ★★★★★★★★★★★
40 IF INKEY$="Q" THEN BEEP0,1
50 IF INKEY$="2" THEN BEEP1,1
60 IF INKEY$="W" THEN BEEP1,2
70 IF INKEY$="3" THEN BEEP1,3
80 IF INKEY$="E" THEN BEEP1,4
90 IF INKEY$="R" THEN BEEP1,5
100 IF INKEY$="5" THEN BEEP1,6
110 IF INKEY$="T" THEN BEEP1,7
120 IF INKEY$="6" THEN BEEP1,8
130 IF INKEY$="Y" THEN BEEP1,9
140 IF INKEY$="7" THEN BEEP1,10
150 IF INKEY$="U" THEN BEEP1,11
160 IF INKEY$="I" THEN BEEP1,12
170 GOTO 40
```

È possibile migliorare un programma come questo e trasformare la tastiera del computer in uno strumento musicale ancor più funzionale. Anche se lo Spectrum, come abbiamo già detto, è scarsamente adatto alla produzione di musica di buona qualità, può essere validamente impiegato per scopi didattici. Inoltre, sono già in commercio alcuni accessori per lo Spectrum, particolarmente progettati per la generazione di effetti musicali.

bianco invece che bianco su nero) e questo aumenta ancora le possibilità grafiche. Con pazienza ed immaginazione è quindi possibile produrre immagini di alta qualità.
I caratteri possono essere visualizzati sia mediante la generica istruzione PRINT, sia tramite l'istruzione POKE, depositando opportuni codici direttamente nelle memorie video e colore del Vic-20. La versione Commodore dell'istruzione PRINT è estremamente efficace e permette di definire il colore di ogni singolo carattere. Anche il movimento del cursore può essere controllato mediante la PRINT, permettendo di produrre facilmente immagini in movimento. La visualizzazione di caratteri mediante la POKE non risulta veloce quanto come con l'istruzione PRINT, ma può essere utile in particolari circostanze.

Capacità in alta risoluzione

Si può produrre grafica ad alta risoluzione con un Vic-20 non dotato di espansioni di memoria, ma in tal caso si riesce ad utilizzare appena metà dello schermo. Il procedimento che permette di ottenere immagini ad alta risoluzione è conosciuto come "bit-mapping" e consente di controllare ciascun pixel all'interno di una definita zona del video. Ogni cella di carattere, appartenente alla griglia di 23 linee per 22 colonne, sullo schermo, è composta da 64 pixel ordinati in una matrice di 8 × 8 punti.
Nella matrice che definisce ogni carattere, esiste una relazione matematica tra le coordinate (x, y) stabilite per un pixel ed il bit corrispondente. Questa può essere utilizzata, assieme ad una combinazione di comandi POKE, per creare immagini composte da singoli pixel.

Cartuccia di super espansione

Creare immagini grafiche ad alta risoluzione con la tecnica del "bit-mapping" può essere un processo lungo e difficile. È possibile eliminare alcuni inconvenienti acquistando la cartuccia di super espansione della Commodore: essa permette l'uso dell'alta risoluzione ed aggiunge alcuni comandi BASIC per il colore e la musica. I comandi comprendono: GRAPHIC per selezionare il livello di visualizzazione, POINT per "accendere" un singolo pixel sullo schermo e PAINT, per colorare le immagini.
Ci sono solo due "nei" nella super espansione: non è disponibile un comando UNPOINT (per cancellare un singolo pixel) e non è possibile colorare l'area adiacente ad una linea diagonale senza danneggiare la linea stessa.



Il coniglio

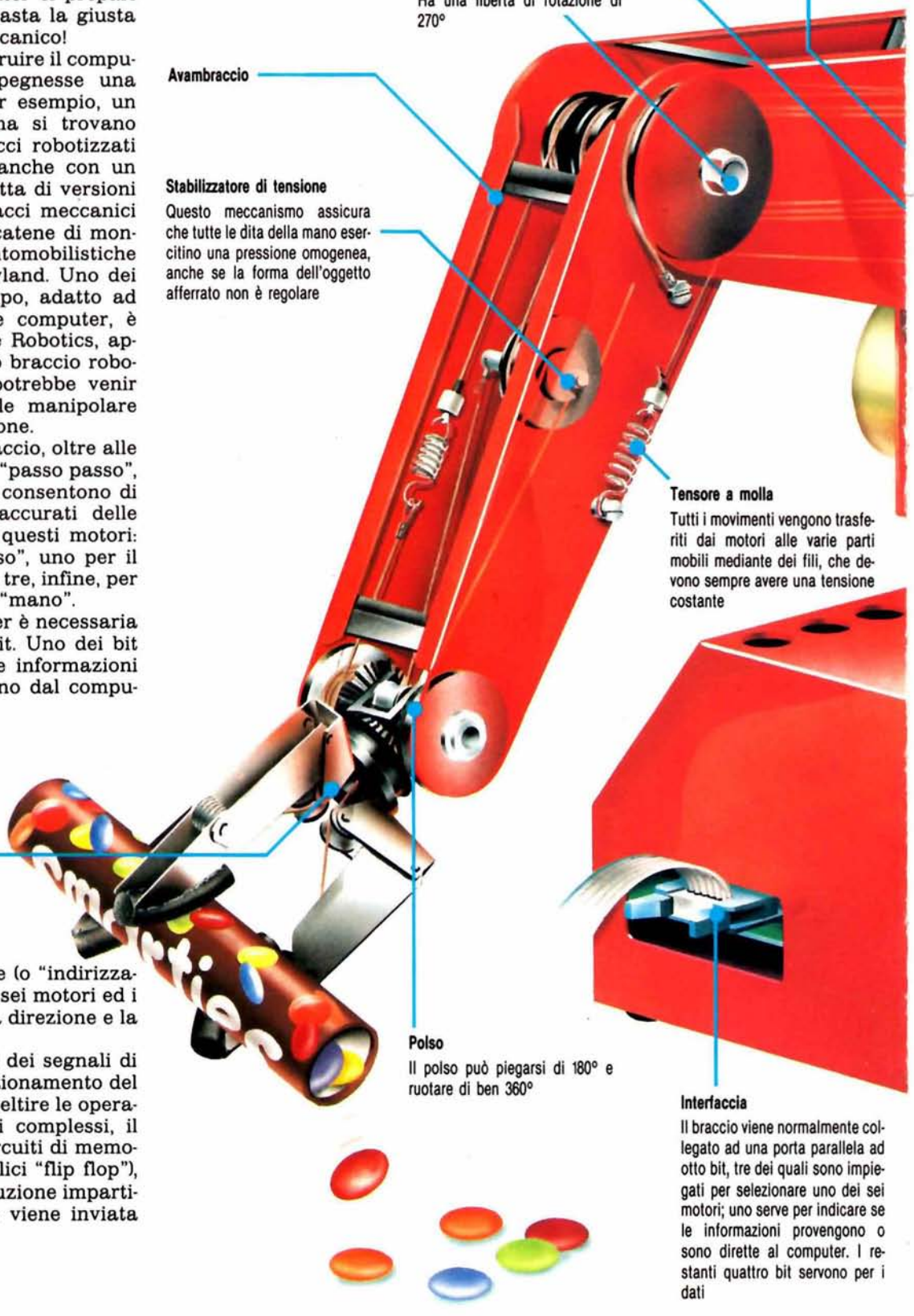
Questo programma, per il Vic-20, dimostra la versatilità dello speciale set di caratteri della Commodore. L'immagine del coniglio è costruita esclusivamente con essi. Per capire come il programma sia concepito, occorre farlo girare il e quindi spostare il cursore sui vari caratteri visualizzati sullo schermo

Il braccio e la mente

Anche un home computer può essere dotato di un braccio meccanico robotizzato

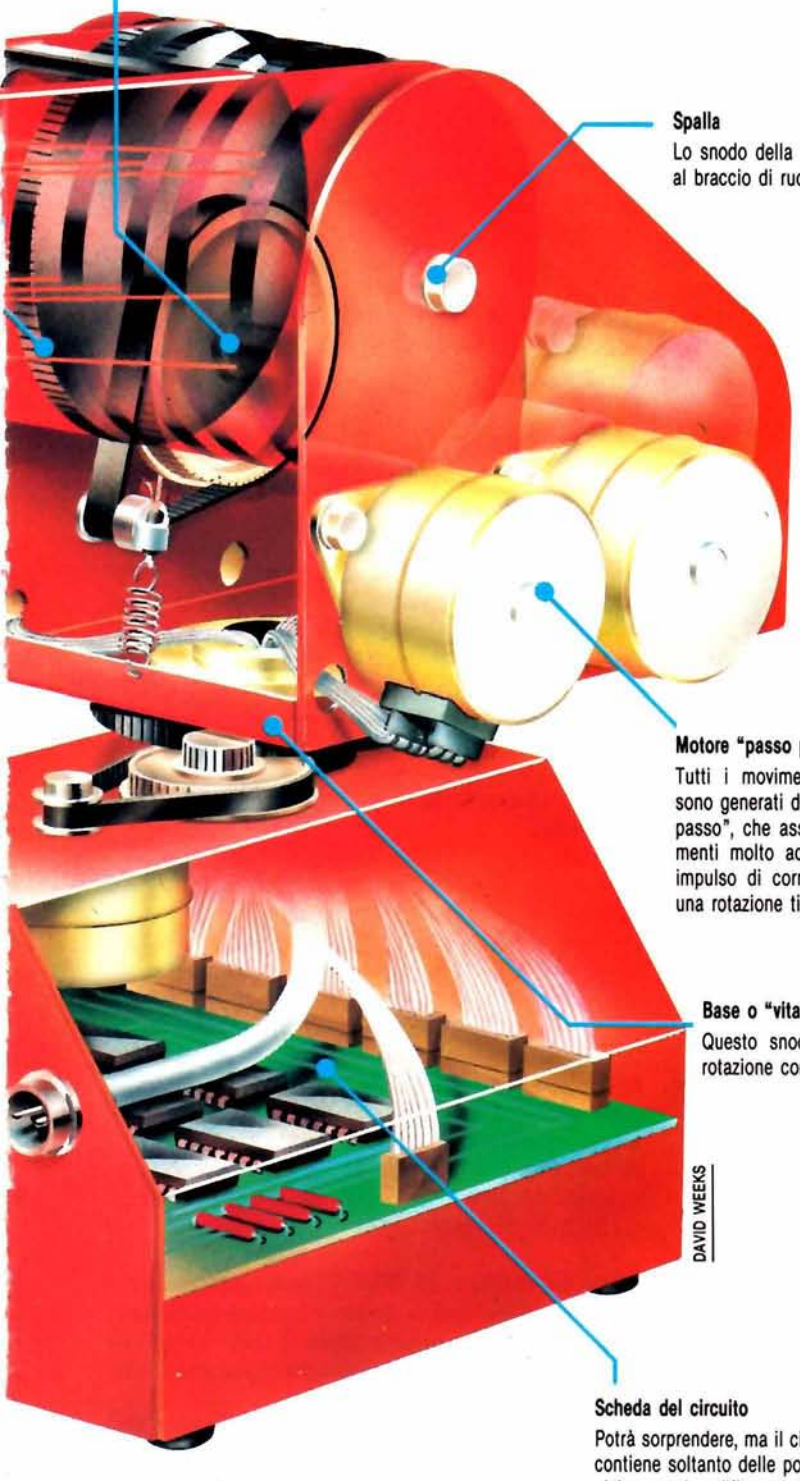
Vogliamo che il nostro computer ci prepari un caffè? Nessun problema: basta la giusta interfaccia ed un braccio meccanico!
Finora, potevamo tutt'al più istruire il computer affinché accendesse o spegnesse una qualsiasi apparecchiatura (per esempio, un bollitore d'acqua elettrico), ma si trovano adesso in commercio dei bracci robotizzati economici e facili da usare, anche con un piccolo home computer. Si tratta di versioni molto semplificate di quei bracci meccanici impiegati, per esempio, nelle catene di montaggio di grandi industrie automobilistiche come la FIAT o la British Leyland. Uno dei primi apparecchi di questo tipo, adatto ad essere collegato ad un home computer, è stato l'"Armdroid" della Colne Robotics, apparso nel 1981. Sebbene questo braccio robotizzato non sia mobile (ma potrebbe venir montato su ruote), è possibile manipolare oggetti con un'elevata precisione.
I principali componenti del braccio, oltre alle parti metalliche, sono i motori "passo passo", che, controllati dal computer, consentono di ottenere spostamenti molto accurati delle varie parti. Sono usati sei di questi motori: uno per la rotazione del "polso", uno per il "gomito", uno per la "spalla", e tre, infine, per controllare i movimenti della "mano".
Per il collegamento al computer è necessaria una porta parallela ad otto bit. Uno dei bit viene usato per indicare se le informazioni sono dirette oppure provengono dal computer.

Tre bit servono per specificare (o "indirizzare", per dirla in gergo) uno dei sei motori ed i restanti quattro bit indicano la direzione e la velocità dei movimenti.
Inoltre, vengono anche inviati dei segnali di clock, per sincronizzare il funzionamento del braccio con il computer. Per sveltire le operazioni e permettere movimenti complessi, il braccio è dotato di appositi circuiti di memorizzazione (dei "latch" o semplici "flip flop"), che mantengono attiva un'istruzione impartita ad uno dei motori, mentre viene inviata quella relativa ad un altro.



Meccanismo ad ingranaggi
Composto da cinghie dentate di gomma e da ingranaggi di opportune dimensioni, il meccanismo permette movimenti accurati, con una tolleranza di circa 2 mm.

Per ottenere movimenti complessi, quali, ad esempio, gli spostamenti necessari per raccogliere un oggetto, è necessario suddividerli in una sequenza di movimenti semplici. Ogni motore dovrà essere programmato separatamente in modo che l'effetto risultante sia omogeneo e senza scatti inutili. In genere, i bracci robotizzati vengono venduti con appositi programmi che ne facilitano l'uso e che permettono al braccio di "imparare" alcuni movimenti ripetitivi.



Spalla
Lo snodo della spalla consente al braccio di ruotare di 180°

Motore "passo passo"
Tutti i movimenti del braccio sono generati da motori "passo passo", che assicurano spostamenti molto accurati. Ad ogni impulso di corrente, si ottiene una rotazione tipica di 7°

Base o "vita"
Questo snodo permette una rotazione completa di 360°

DAVID WEEKS

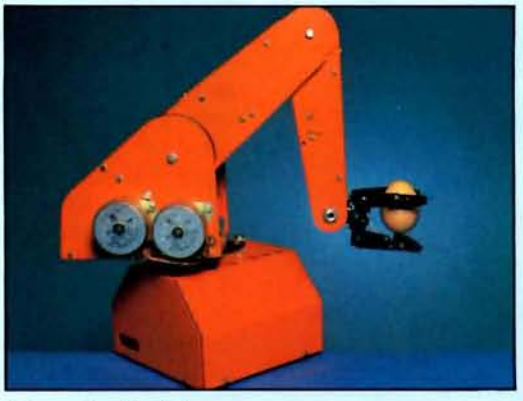
Scheda del circuito
Potrà sorprendere, ma il circuito contiene soltanto delle porte logiche, per decodificare i segnali emessi dal computer: non vengono usati né microprocessori, né RAM e neppure ROM

Spesso viene usato... un uovo per verificare la precisione dei movimenti ed il computer deve essere in grado di controllare molto accuratamente la presa della mano meccanica: se la presa non è sufficiente, l'uovo cade; al contrario, se è troppo forte, l'uovo si rompe. Diversi sistemi possono essere usati per convogliare le informazioni dal braccio meccanico al computer, ma uno dei più comuni è l'impiego di interruttori miniaturizzati (microswitch), che vengono montati in opportune posizioni, per controllare la corsa delle varie parti in movimento.

In alternativa ai microswitch, nei bracci meccanici più sofisticati e costosi vengono adoperati speciali sensori di pressione: certi materiali, nei quali la resistenza elettrica varia proporzionalmente alla pressione esercitata su di essi, si prestano egregiamente allo scopo.

Qualora non sia previsto alcun ritorno d'informazione (chiamato in inglese "feedback") dal braccio al computer, il sistema si dice "a ciclo aperto" o deterministico. Nell'esempio poc'anzi citato, un tale sistema causerebbe la rottura dell'uovo. Ma, se è prevista una qualche forma di feedback, che permetta al computer di aggiustare accuratamente i movimenti, allora il sistema è detto "a ciclo chiuso" o stocastico. Tornando all'esempio dell'uovo, grazie ai sensori di pressione esso viene afferrato saldamente, ma non abbastanza da romperlo.

Nei sistemi robotizzati più sofisticati sono montati svariati tipi di sensori: per misurare la luce, il calore o altri fattori variabili. L'insieme delle informazioni viene quindi elaborato dal computer per rilevare eventuali situazioni anomale di funzionamento.



IAN McKINNELL

Il linguaggio dei robot
È abbastanza semplice scrivere programmi per il controllo di braccia robotizzate. In BASIC, si tratta di far accettare dei semplici comandi, immessi da tastiera, di codificarli opportunamente e di passarli alla porta d'uscita, alla quale è collegato il braccio. In genere vengono usate delle istruzioni POKE o OUT. Quando si tratta di leggere le informazioni "di ritorno", invece, si impiega l'istruzione PEEK o INP. Se, però, la velocità di funzionamento è un fattore essenziale, occorre scrivere i programmi in linguaggio macchina. In alternativa è possibile, su certi computer, adoperare il FORTH, un linguaggio simile al BASIC, ma con la velocità del linguaggio macchina. In FORTH un gruppo di istruzioni può essere identificato con un nome, che viene poi impiegato nel resto del programma come se si trattasse di un singolo comando. Ad esempio, una completa operazione che consista nell'afferrare un oggetto, può essere chiamata semplicemente PRESA

Come usare i file

Le istruzioni necessarie alla memorizzazione di dati su nastro o su dischetto

La rubrica computerizzata, che stiamo sviluppando nel corso delle ultime lezioni, è paragonabile ad un "database" in versione ridotta, per quanto riguarda l'uso dei file (archivi).
Il termine "file" merita di essere definito in modo preciso, dato il suo larghissimo impiego in informatica.
In un sistema computerizzato, un file è un gruppo di informazioni conservato su supporto magnetico (nastri o dischetti). Ad ogni file è assegnato un nome identificativo, che serve sia all'operatore, sia al sistema operativo del computer per l'accesso al suo contenuto. Le informazioni che costituiscono un file possono essere le istruzioni di programma o i dati necessari al funzionamento di un programma.
Anche nel caso della nostra rubrica computerizzata, le informazioni memorizzate sono di due generi ben distinti: il programma ed i dati sui quali esso lavora.
I dati sono raggruppati in "record", ciascuno contenente informazioni di vario genere (nominativo, indirizzo, numero telefonico, ecc.), ma anche alcune informazioni riservate al sistema operativo e normalmente non utilizzate dall'utente del computer.
Queste informazioni "interne" possono consistere in speciali contatori, utili al sistema operativo per conoscere la posizione di un particolare dato, oppure "flag" (marcatori) per riconoscere un record "cancellato" dagli altri e così via.
Nel corso delle prime lezioni, abbiamo spesso utilizzato le istruzioni READ e DATA, facendo in modo che i dati necessari ad una particolare elaborazione fossero contenuti nello stesso programma.
Questo metodo può essere conveniente se i dati non sono soggetti a frequenti cambiamenti (per esempio, il numero dei giorni dei mesi). Se, al contrario, prevediamo di usare dati sempre diversi per l'elaborazione, il programma potrebbe prevedere l'impiego di opportune istruzioni INPUT.
A questo punto, però, sorge il problema di conservare i dati immessi, anche dopo aver spento l'apparecchio. Nella nostra rubrica computerizzata, infatti, sarebbe impensabile voler inserire i nominativi usando l'istruzione DATA: ad ogni nuovo nominativo occorrerebbe modificare il listato del programma e ciò non è affatto pratico.
Molto opportunamente, è previsto un sistema di conservazione dei dati (su nastri o dischetti magnetici) esternamente al programma, mediante l'impiego dei file. I dati vengono conservati con un formato simile a quello adoperato con le frasi DATA. Il programma è in grado di "aprire" uno o più file, leggere i dati da esso (solitamente trasferendoli in una matrice) e quindi di "chiudere" il file. Nel caso di una modifica ai dati, il programma "apre" il file, legge un particolare record, ne modifica il contenuto, riscrive il record ed, infine, richiude il file.
Nei sistemi che impiegano unità a dischetti, questo intero processo non richiede che pochi secondi, mentre in quelli che utilizzano unità a nastri, i tempi sono spesso molto più lunghi (talvolta è necessario riavvolgere il nastro ed aspettare che si sia riposizionato correttamente sulle informazioni da modificare).
Un altro vantaggio dei sistemi a dischi è la possibilità di aprire più file contemporaneamente, operazione che non è possibile fare coi nastri.
I file, dunque, sono un insieme di dati, conservati su "memorie di massa", utilizzabili da uno o più programmi.
Un word processor, ad esempio, potrebbe accedere ad un file contenente i nomi e gli indirizzi dei clienti per la creazione di lettere circolari personalizzate.
Le regole che governano l'uso dei file variano da una versione BASIC all'altra: i manuali operativi dedicano, in genere, qualche capitolo esclusivamente all'uso dei file e delle relative istruzioni OPEN e CLOSE. Riportiamo alcuni esempi, di carattere molto generale, per dare un'idea sull'argomento.

I formati dei file, in BASIC, sono solitamente due: "random" (ad accesso casuale) e sequenziali. In un file sequenziale, i dati vengono conservati, appunto, sequenzialmente: uno dopo l'altro.
Per accedere al trentaduesimo elemento, occorre leggere tutti quelli che lo precedono. In un file random, al contrario, l'accesso può avvenire in modo diretto: se ci occorre leggere (o scrivere) il trentaduesimo record, ciò viene fatto virtualmente ignorando tutti gli

altri, senza dover iniziare dal primo della lista. Per il momento, ci occuperemo dei file sequenziali, gli unici disponibili nei sistemi che impiegano unità a nastri.
Supponiamo di voler memorizzare le temperature medie giornaliere di una settimana (adoperiamo la notazione BASIC: il punto al posto della virgola):

LUNEDÌ	13.6
MARTEDÌ	9.6
MERCOLEDÌ	11.4
GIOVEDÌ	10.6
VENERDÌ	11.5
SABATO	11.1
DOMENICA	10.9

Per semplificare, impiegheremo soltanto variabili numeriche (Lunedì corrisponde ad 1, Martedì a 2, ecc.), cosicché i dati possono essere espressi nella forma:

```
1,13.6,2,9.6,3,11.4,4,10.6,5,11.5,6,11.1,7,10.9
```

Per conservare questi dati in un file sequenziale dovremo:

```
APRIRE (OPEN) il file
scrivere i dati nel file
CHIUDERE (CLOSE) il file
```

Ogni volta che apriamo un file, occorre specificare se intendiamo scrivere in esso (output) oppure leggere da esso (input). Nel Micro BBC, le rispettive istruzioni sono OPENOUT e OPEN. Nel BASIC della Microsoft, invece, le forme equivalenti sono: OPEN "O" ed OPEN "I", come riportato nel seguente frammento di programma:

```
100 OPEN "O", #1,"TEMP.DAT"
110 PRINT #1,1,13.6,2,9.6,3,11.4,4,10.6,5,11.5,6,11.1,7,10.9
120 CLOSE #1
```

Alla linea 100, l'istruzione OPEN, seguita da "O", indica al computer che intendiamo usare un file in scrittura; mentre #1 indica che, nelle operazioni di scrittura, identificheremo il file col numero 1.
Questa numerazione è puramente arbitraria, ma deve essere impiegata coerentemente nelle istruzioni PRINT# e INPUT#, relative, rispettivamente, alla scrittura ed alla lettura di dati dai file. Infine, il nome identificativo (o, se si vuole, "mnemonico") del file è quello racchiuso tra doppi apici: "TEMP.DAT".
Anche questo nome è arbitrario: abbiamo scelto TEMP, in quanto i dati si riferiscono a temperature e DAT per specificare che il file contiene dati.
Una versione completa, in BASIC Microsoft, del programma per la scrittura e successiva rilettura dei dati è il seguente:

```
100 OPEN "O", #1, "TEMP.DAT"
110 PRINT #1,1,13.6,2,9.6,3,11.4,4,10.6,5,11.5,6,11.1,7,10.9
120 CLOSE #1
130 REM IN QUESTO PUNTO POTREBBERO ESSERE
140 REM INSERITE ALTRE ISTRUZIONI
150 OPEN "I", #1, "TEMP.DAT"
160 FOR X = 1 TO 7
170 INPUT #1,GIORNO,TEMP
180 PRINT "GIORNO";GIORNO,TEMP
190 NEXT X
200 CLOSE #1
210 END
```

Il programma apre per la scrittura un file "TEMP.DAT" assegnandogli il numero identificativo 1, vi scrive i dati mediante la PRINT# e quindi chiude il file. Dopo di che, riapre il medesimo file (stesso nome e numero identificativo) e, mediante l'istruzione INPUT# inserita nel ciclo, legge i valori contenuti nel file e li visualizza sullo schermo.

Ma torniamo, a questo punto, alla nostra rubrica computerizzata ed, in particolare, allo sviluppo della parte INIZIALIZZAZIONE.
In primo luogo dobbiamo calcolare l'occupazione di memoria per ogni singolo record dell'archivio. L'uso di campi di lunghezza fissa, anche se comporta un sensibile spreco di memoria, facilita enormemente il lavoro di programmazione. Se assegniamo 40 caratteri a ciascun nominativo, essi occuperanno altrettanti byte di memoria, anche se molti di essi non conterranno altro che il codice "spazio", ossia, ai fini della visualizzazione, saranno vuoti. In quasi tutte le versioni di BASIC, comunque, quando una variabile alfanumerica viene DIMensionata, ad ogni suo elemento sono riservati ben 256 caratteri: il dimensionamento non riguarda la lunghezza degli elementi, ma soltanto il loro numero massimo.
Se il BASIC da noi impiegato consente l'uso di matrici alfanumeriche a più dimensioni non esistono grandi problemi, altrimenti il metodo più semplice consiste nell'usare una variabile alfanumerica per ogni campo del record.
In alternativa, ecco un trucchetto molto comodo per simulare una matrice multidimensionale.
Supponiamo di voler impiegare una matrice di tre righe e cinque colonne. Nei BASIC che permettono l'uso di matrici a più dimensioni si userebbe comodamente una DIM A(3,5), contenente, quindi, 15 elementi. Noi, invece, possiamo scrivere DIM A(15) e, ogni volta che dovremo accedere all'elemento A(R,C), usare la forma A((R−1)*5,+C).
Se impieghiamo una variabile per ogni campo del record, dobbiamo decidere come DIMensionare le matrici. Si potrebbe assegnare una dimensione fissa, ma ciò limiterebbe la quantità di nominativi archiviabili. Un sistema potrebbe essere quello di dimensionare le

matrici in base al numero di nominativi presenti nel database. Purtroppo, però, non tutte le versioni di BASIC consentono di dimensionare le matrici oltre un certo limite e, comunque, la quantità di memoria RAM necessaria potrebbe essere superiore a quella disponibile. Un breve programma per calcolare empiricamente la massima dimensione consentita dalla particolare versione di BASIC, per una variabile alfanumerica indicizzata, è il seguente:

```
10 READ D$
20 INPUT "DIMENSIONE DELLA VARIABILE";A
30 DIM N$(A)
40 FOR L = 1 TO A
50 LET N$(L) = D$
60 NEXT L
70 FOR L = 1 TO A
80 PRINT L,N$(L)
90 NEXT L
100 CLEAR
110 GO TO 10
120 DATA "IL MIO COMPUTER"
130 END
```

Ogni volta che il programma riparte, viene chiesta la dimensione da assegnare alla variabile N$(): occorre rispondere con valori che siano sempre crescenti, fino ad ottenere un errore del tipo OUT OF MEMORY (memoria insufficiente).
L'istruzione CLEAR, alla linea 100, serve per azzerare le dimensioni (ed il contenuto) di tutte le variabili ad ogni nuovo "giro". Senza di essa, otterremmo, al secondo passaggio, un errore del tipo "REDIMENSIONED ARRAY", ossia il BASIC ci avvertirebbe che tentiamo di dimensionare una matrice precedentemente dimensionata.
Anche se ogni elemento fosse composto da 40 caratteri e vi fossero 5 campi per ogni record e 256 elementi accantonati per ogni matrice, la memoria necessaria per contenere tutti i dati avrebbe dimensioni piuttosto notevoli: calcolando che è necessario un byte per ogni carattere, occorrerebbero 51.200 byte (5 × 40 × 256 byte).
Non è certo conveniente impostare il lavoro su queste premesse ed ecco perché vengono impiegati i file esterni, senz'altro preferibili ad interminabili frasi DATA, contenute nello stesso programma, e pertanto difficilmente modificabili.
L'impiego dei file non è molto difficile e, fatta un po' di pratica con semplici programmini di scrittura e lettura, non si trovano ostacoli di rilievo ad applicarne l'uso a programmi anche piuttosto complessi.
Per offrire materia di esempio, abbiamo sviluppato il semplice programma per l'archiviazione delle temperature in due versioni BASIC piuttosto diverse tra loro: quella dello Spectrum e quella del Micro BBC (ambedue diverse, a loro volta, dal BASIC della Microsoft).
Nel BASIC dello Spectrum, le istruzioni OPEN# e CLOSE# sono riservate all'impiego dei Microdrive (i micro dischetti della Sinclair); quando si usa l'unità a nastri occorre impiegare una variazione delle istruzioni LOAD e SAVE, che normalmente servono per "salvare" su nastro i programmi o il contenuto di variabili semplici. Per salvare il contenuto di variabili indicizzate, si impiega l'istruzione SAVE-DATA, nel formato:

```
SAVE nomefile DATA nomematrice ( )
```

"Nomefile", nel nostro esempio, corrisponde a TEMP.DAT, mentre "nomematrice" rappresenta il nome della variabile indicizzata che intendiamo archiviare, seguito dalle parentesi "vuote" ( ). Ovviamente, tale variabile deve essere stata in precedenza dimensionata e riempita con gli appropriati valori. Per rendere evidente la differenza fra il nome del file e quello della matrice, abbiamo chiamato il primo TEMPDAT ed il secondo C$:

```
10 DIM C$(14)
20 FOR X = 1 TO 14
30 READ C$(X)
40 NEXT X
50 DATA
60 SAVE "TEMPDAT" DATA C$( )
70 STOP
80 LOAD "TEMPDAT" DATA X$( )
90 FOR L = 1 TO 14 STEP 2
100 PRINT "GIORNO";C$(L,C$(L+1))
110 NEXT L
120 STOP
```

La linea 60 provvede all'archiviazione dei valori contenuti in C$ in un file chiamato "TEMPDAT". La linea 70 sospende l'esecuzione del programma, per consentire il riavvolgimento del nastro, dopodiché si può impartire il comando CONT (=CONTinua), che permette di riprendere l'esecuzione dalla linea 80, grazie alla quale i dati vengono letti dal nastro e trasferiti nella variabile C$.
Per quanto riguarda il BASIC del Micro BBC (uno tra i più sofisticati nel settore degli home computer), possiamo sfruttare la possibilità di definire delle "procedure", come si può notare dal seguente listato, dove compaiono anche le istruzioni OPENOUT (per aprire i file in scrittura) ed OPENIN (per aprire i file in lettura):

```
10 DIM C$(2,7)
20 FOR R = 1 TO 7
30    FOR C = 1 TO 2
40       READ C$(C,R)
50    NEXT C
60 NEXT R
70 DATA 1,13.6,2,9.6,3,11.4,4,10.6,5,11.5,6,11.1,7,10.9
```

```
80 INPUT "DIGITARE UNA 'S' PER SALVARE I DATI",K$
90 IF K$ <> "S" THEN GOTO 80
100 PROCSAVE : CLEAR : DIM C$(2,7)
110 INPUT "RIAVVOLGERE IL NASTRO E DIGITARE UNA 'L'",K$
120 IF K$ <> "L" THEN GOTO 110
130 PROCLOAD
140 PRINT "GIORNO TEMPERATURA"
150 FOR R = 1 TO 7
160 FOR C = 1 TO 2
170       PRINT C$(C,R);"          "
180    NEXT C : PRINT
190 NEXT R
200 END
300 DEF PROCSAVE
310 X = OPENOUT ("TEMPDAT")
320 FOR R = 1 TO 7
330     FOR C = 1 TO 2
340        PRINT #X,C$(C,R)
350     NEXT C
360 NEXT R
370 CLOSE #X
380 ENDPROC
400 DEF PROCLOAD
410 X = OPENIN ("TEMPDAT")
420 FOR R = 1 TO 7
430    FOR C = 1 TO 2
440       INPUT #X,C$(C,R)
450    NEXT C
460 NEXT R
470 CLOSE #X
480 ENDPROC
```

Uno dei vantaggi più evidenti, nel BASIC del Micro BBC, è che le istruzioni relative ai file funzionano, senza alcuna modifica, sia per le unità a nastri che per le unità a dischi. Ciò è particolarmente utile quando si desidera scambiare programmi con altri possessori dello stesso home computer.
In questa lezione abbiamo visto come i dati possono essere trasferiti dalla memoria del computer alle memorie di massa, e viceversa, introducendo il concetto di file. Abbiamo inoltre esaminato un aspetto della parte INIZIALIZZAZIONE (il dimensionamento delle matrici), che svilupperemo nella prossima lezione del corso. Vedremo quali e quante matrici occorrono per la nostra rubrica computerizzata ed a quale punto del programma sarà necessario trasferirne il contenuto sulle memorie di massa.

Esercizio

Si scriva un programma così strutturato:

```
INIZIO
  INIZIALIZZA
  IMMISSIONE DATI
  SCELTA
      Scrittura dei dati
      Lettura dei dati
      Uscita dal programma
FINE
```

INIZIALIZZA deve dimensionare le matrici impiegate nel programma ed eventualmente assegnare alle variabili i valori iniziali. I dati potrebbero essere, ad esempio, 15 nominativi da essere immessi con delle INPUT dalla tastiera.
Il comando SCELTA deve visualizzare sullo schermo un menu di opzioni, permettendo all'operatore di scegliere fra le tre possibilità suindicate.
Sarebbe utile introdurre, nella parte che riguarda l'uscita dal programma, l'uso di una variabile "di comodo", che provveda a salvare i dati se (e soltanto se), a seguito di una nuova immissione, si tentasse di uscire dal programma senza aver prima effettuato il salvataggio.

Basic & Basic

VARIABILI

Solo il Basic del Micro BBC, tra quelli presenti sulla maggior parte degli home computer, permette l'uso di nomi di variabili di una certa lunghezza (ad esempio: NOMINATIVO$). Lo Spectrum consente nomi "lunghi" per le variabili numeriche, ma nomi di una sola lettera per quelle alfanumeriche. Nel Dragon 32, nel Vic-20 e nel Commodore 64 è possibile usare nomi "lunghi", ma soltanto i primi due caratteri sono significativi, cosicché NOMINATIVO$ e NOTAZIONE$ si riferiscono alla stessa variabile alfanumerica, avendo i primi due caratteri identici. Nell'Oric-1 i nomi delle variabili sono composti, al massimo, da due caratteri (una lettera seguita da un numero), mentre il Lynx permette nomi di variabili di una sola lettera, sebbene distingua tra maiuscole e minuscole.

OPEN

Sul Dragon 32 occorre usare il seguente formato:

```
OPEN "O", #-1,"TEMPDAT"
PRINT #-1,1,13.6,2.9.6,3,11.4, ecc.
CLOSE #-1
```

e

CLOSE

```
OPEN "I", #-1,"TEMPDAT"
INPUT #-1,D,T
CLOSE #-1
```

PRINT#

Sul Commodore 64 e sul Vic-20 invece:

```
OPEN 1,1,2,"TEMPDAT"
PRINT #1,1,13.6,2,9 6,3,11.4, ecc.
CLOSE 1
```

e

```
OPEN 1,1,0,"TEMPDAT"
INPUT #1,D,T
CLOSE 1
```

LYNX

Né sul Lynx, né sull'Oric-1 sono disponibili, attualmente, istruzioni per il trattamento di file su nastri. Presto saranno disponibili le opportune espansioni.

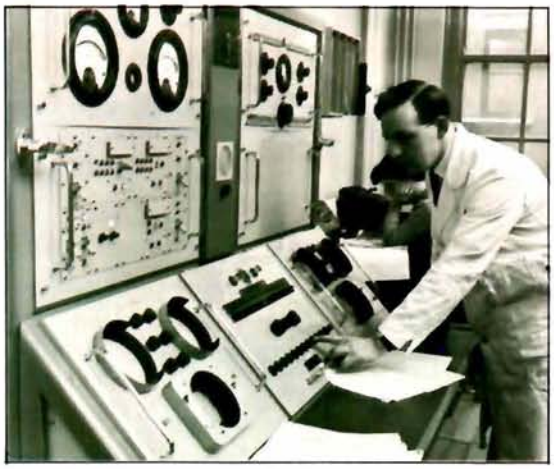
ORIC-1

Lyons: pasticcini ed informatica

La catena di sale da tè che ha creato il primo computer commerciale

L'ufficio elettronico

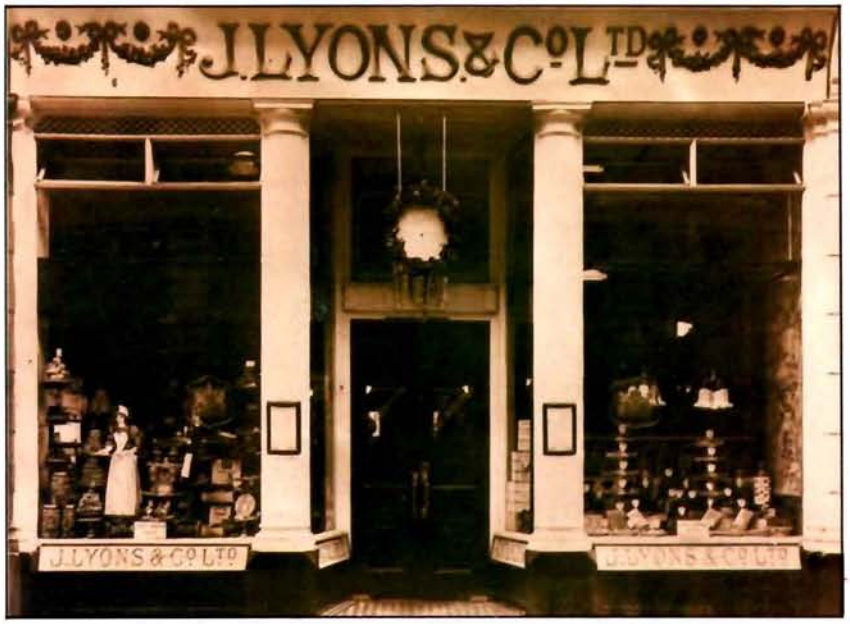
Contrariamente ai computer precedenti, adatti per impieghi militari o universitari, il LEO 1 era progettato per eseguire pochi calcoli, ma su enormi quantità di dati



La sala da tè elettronica

Sembra quasi impossibile che una tradizionale saletta da tè inglese sia stata la sede di una rivoluzionaria applicazione commerciale, eppure è stato proprio grazie ai particolari problemi di gestione contabile della catena di negozi J. Lyons, che sono stati sviluppati i primi computer per applicazioni commerciali

Nel 1947 venne presa una decisione piuttosto pionieristica: tentare la costruzione di un computer che potesse automatizzare la contabilità di un ufficio. Si sarebbe trattato del primo computer al mondo, destinato ad attività commerciali. Questa storica iniziativa venne intrapresa, fatto abbastanza sorprendente, da una ditta decisamente estranea all'informatica: la J. Lyons, una compagnia inglese che gestisce una grande catena di negozi di pasticceria e tè.



Gli affari della Lyons implicavano un gran numero di piccole transazioni e, per cercare di rendere remunerativa un'attività commerciale di questo tipo, era necessario tenere continuamente aggiornati i libri contabili.
La Lyons aveva già la fama di essere una ditta al passo coi tempi: adottò le prime calcolatrici meccaniche fin dal 1896 e nel 1930 sperimentava l'impiego di microfilm per archiviare le proprie transazioni.
Periodicamente, alcuni impiegati della Lyons venivano inviati all'estero per apprendere nuovi metodi di lavoro e, nel 1947, due di essi vennero mandati negli Stati Uniti, per indagare su di un nuovo apparecchio: il cervello elettronico. La sola scoperta significativa, però, fu che un computer era in fase di realizzazione molto più vicino a casa: addirittura a Cambridge.
La Lyons commissionò immediatamente uno studio sulla possibilità di farsi costruire un computer a proprio uso e consumo. In termini economici, l'analisi portò a questi risultati: il costo della realizzazione si sarebbe aggirato sulle 100.00 sterline, ma avrebbe consentito un risparmio di ben 50.000 sterline all'anno. Nell'ottobre del 1947 venne dato il via al progetto. L'impresa non era di immediata realizzazione, in quanto il computer dell'Università di Cambridge era appena allo stadio sperimentale. La Lyons versò un contributo iniziale di circa 3.000 sterline all'Università di Cambridge per finanziare in parte quello che sarebbe divenuto l'EDSAC (Electronic Delay Storage Automatic Computer, il calcolatore automatico con immagazzinamento elettronico). Nel 1949 l'EDSAC svolse il suo primo lavoro: calcolare una tabella di numeri primi. La Lyons, nel frattempo, preparò una serie di annotazioni sulle procedure contabili da far svolgere al computer e quest'analisi particolareggiata risultò estremamente utile in fase di progetto dei primi programmi per il nuovo computer. Fin dal principio, però, ci si accorse che mentre l'EDSAC era concepito per eseguire un'enorme massa di calcoli, partendo da un input numerico relativamente modesto, un computer per applicazioni commerciali avrebbe dovuto gestire un tipo di elaborazione totalmente all'opposto: poche operazioni di calcolo, ma enormi quantità di dati da elaborare.
Nacque così, il 9 febbraio 1954, il LEO (il Lyons Electronic Office, ossia l'ufficio elettronico Lyons) inizialmente impiegato per calcolare le paghe settimanali di oltre 1700 dipendenti. In un solo secondo e mezzo, era capace di fare le stesse operazioni che richiedevano otto minuti manualmente.
Il LEO ebbe un gran successo, e la Lyons si accorse ben presto della necessità di disporre di più apparecchi del genere. Anche altre ditte cominciarono ad interessarsi al LEO, e la Leo Computers, la ditta produttrice, riuscì a collocare molti apparecchi, con prestazioni sempre migliori, presso diverse industrie, enti governativi ed organizzazioni commerciali. Nel 1963 la ditta venne acquistata dalla English Electric Company.

Il computer ergonomico

Gli apparecchi del prossimo futuro terranno in gran conto anche la comodità dell'operatore

Nel progetto di apparecchi e strumenti, due elementi fanno parte del "design": quello estetico, che riguarda la bellezza della forma e quello ergonomico, che concerne i rapporti che intercorrono tra l'utilizzatore dell'apparecchio ed il suo ambiente di lavoro. Anche l'occhio vuole la sua parte, ed una certa gradevolezza estetica può contribuire ad alleggerire il lavoro.

Similmente, anche l'ambiente di lavoro non dovrebbe essere né scomodo né, peggio ancora, alienante.

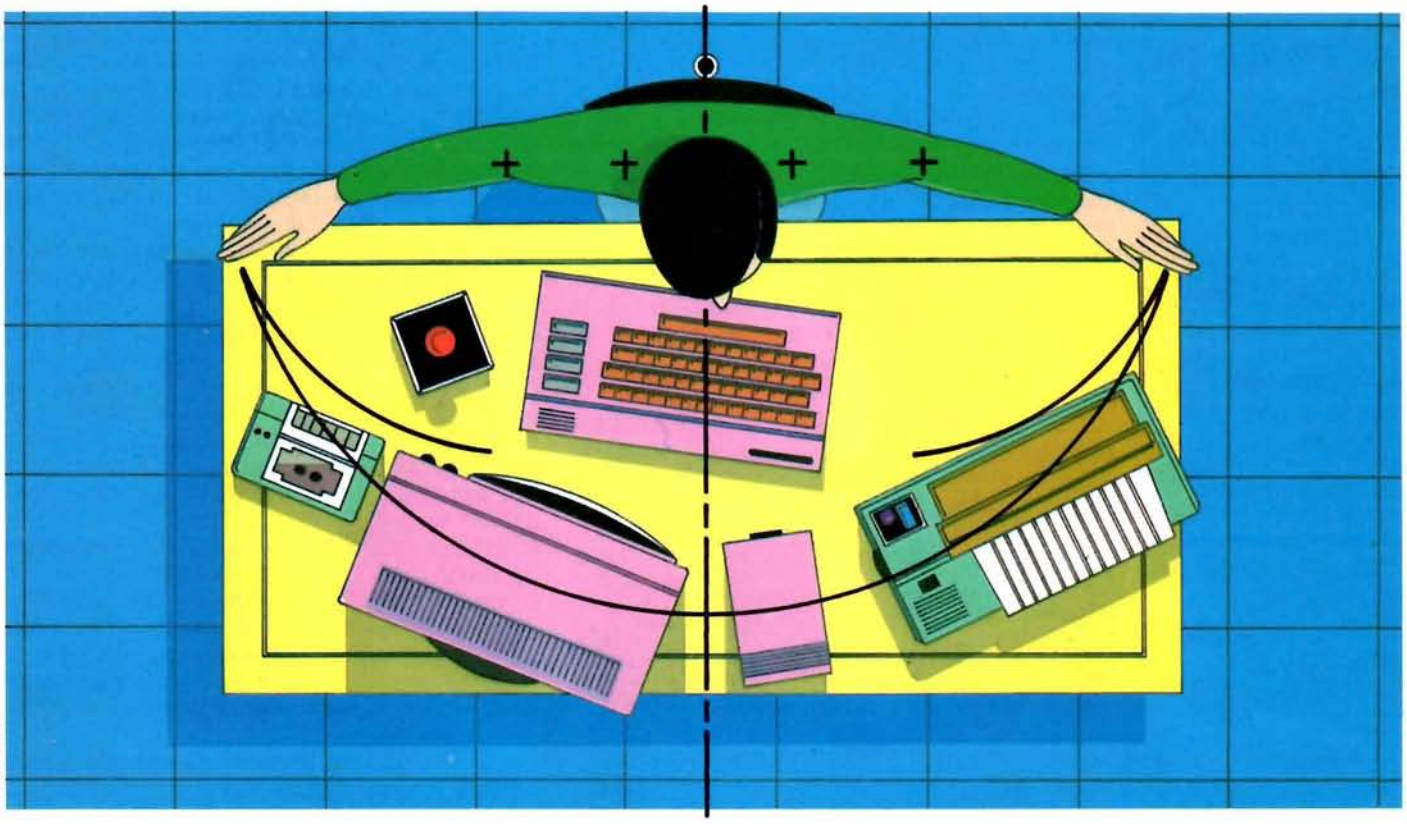
Probabilmente, nella scelta di un computer, si tiene più conto del prezzo e delle prestazioni tecniche, che non delle sue caratteristiche ergonomiche, ma, poiché si dovranno trascorrere diverse ore del giorno seduti di fronte alla macchina, può giovare qualche considerazione sull'ambiente fisico col quale verrà usata la macchina.

Innanzi tutto, il lavoro dovrebbe svolgersi in un ambiente comodo: una scrivania di giuste dimensioni, con un piano di lavoro adeguato alle necessità dell'operatore. La programmazione di un computer è già abbastanza complicata di per sé ed è piuttosto insensato renderla ancora più difficile lavorando in condizioni del tutto inadeguate. Si possono seguire varie strade per creare un ambiente di lavoro più confortevole. Partiamo dallo schermo, che deve consentire una facile lettura delle informazioni. Usando un normale televisore, anziché un apposito monitor, dovremo sopportare fastidiosi fenomeni di riflessione e non sempre è possibile applicare gli speciali filtri oggi in commercio per ridurre al minimo questo inconveniente che a lungo andare è dannoso alla vista. Tuttavia, è possibile migliorare la qualità dell'immagine televisiva applicando un filtro direttamente sul video: esistono filtri semplicemente colorati oppure anche polarizzati, che eliminano in parte i fenomeni di riflessione. Questi filtri permettono di ottenere un buon contrasto anche a basso livello di luminosità, riducendo così l'affaticamento degli occhi.

Anche il livello di luminosità dell'ambiente esterno ha una certa importanza: la migliore fonte di luce artificiale è una lampada da tavolo a bassa intensità, che illumini la tastiera e gli appunti di lavoro, ma escluda il video dal cono di luce. La distanza del video dagli occhi dovrebbe essere di circa un braccio, ossia di circa 70-80 centimetri. L'inclinazione del video dovrebbe essere aggiustabile, in modo che il piano dello schermo abbia

Il linguaggio del corpo

Il corpo umano può avere diverse forme e dimensioni, ma le proporzioni tra le varie parti di esso restano per lo più costanti, come ben sanno gli studenti di disegno figurativo. È proprio grazie a questa costanza di proporzioni, sulla quale si fonda lo studio dell'ergonomia, che vengono stabilite delle regole generali di progettazione degli ambienti di lavoro. Nel caso di un computer, il video dovrebbe trovarsi alla distanza di un braccio dall'operatore per ridurre al minimo lo sforzo di "accomodazione" degli occhi, quando questi vengono spostati dal video agli appunti sulla scrivania e viceversa. Anche la posizione e la forma della tastiera andrebbero definite in base a principi ergonomici



ROY INGRAM

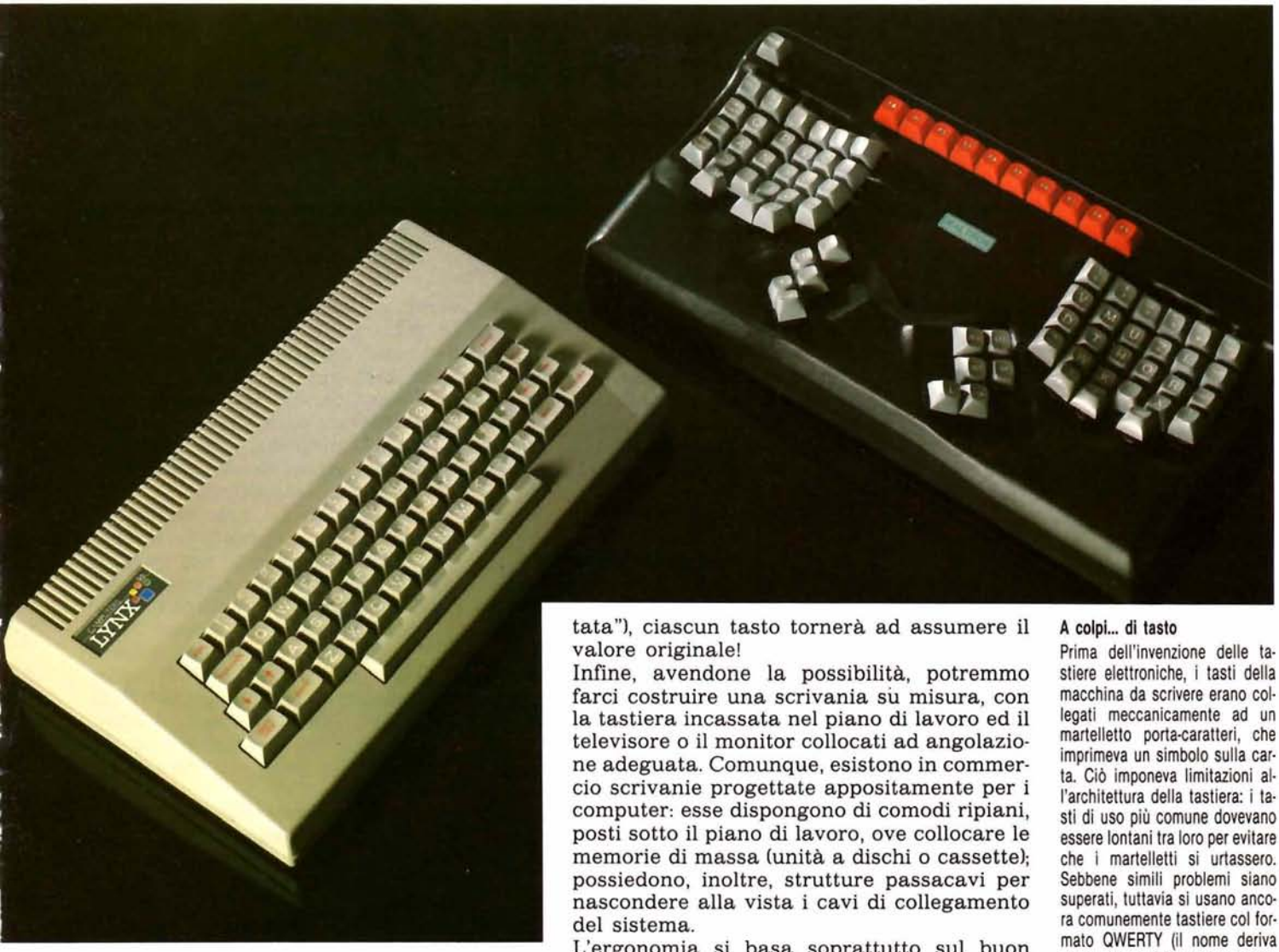
un'angolazione di 90 gradi rispetto alla linea, ideale, che congiunge gli occhi dell'operatore al centro dello schermo. Ciò si può ottenere, eventualmente, inserendo uno o due libri sotto la parte anteriore del televisore o del monitor. In queste condizioni, però, può crearsi un nuovo problema: la riflessione sullo schermo della sagoma stessa dell'operatore. Un filtro a superficie opaca elimina quasi del tutto questo inconveniente.
Dopo aver reso confortevole l'uso dello schermo, consideriamo ora la conformazione fisica e le caratteristiche della tastiera.
Il fattore più importante è l'altezza dei tasti rispetto al piano di lavoro ed i rapporti d'angolazione tra le varie file di tasti. In condizioni ideali, la tastiera dovrebbe essere collocata su di un piano di lavoro leggermente più basso rispetto agli avambracci ed ai polsi dell'operatore ed, inoltre, dovrebbe avere un'inclinazione regolabile. Sfortunatamente, ben pochi home computer possiedono una tastiera dal profilo basso: fanno eccezione la serie ZX della Sinclair, l'Oric-1 e pochi altri, ma quasi tutti impiegano tasti a strati di membrane, al posto dei normali tasti a molla. Le tastiere di questo genere, dette "multistrati", non sono particolarmente adatte per chi è abituato alla tastiera di una macchina da scrivere. Infatti, spesso esse presentano una spaziatura tra i tasti molto ridotta e non offrono sufficiente conferma tattile sulla pressione dei vari tasti. L'insieme di questi difetti rende estremamente faticoso l'inserimento di lunghi programmi o l'elaborazione di testi.
Nell'Oric-1 e nello Spectrum parte del problema è stato risolto con l'emissione di un breve segnale acustico di conferma, quando viene premuto un tasto. Molte ditte hanno commercializzato, per i computer della Sinclair, delle tastiere alternative, di dimensioni standard e con tasti a molla, ma i modelli migliori, che conservano la caratteristica dei tasti "multifunzione", sono piuttosto costosi.
La migliore disposizione dei tasti è stata argomento di interminabili discussioni: quando furono fabbricate le prime macchine da scrivere, nel XIX secolo, le loro tastiere erano molto diverse da quelle attuali.
Generalmente, i tasti dei caratteri più frequentemente usati si trovavano raggruppati nella parte centrale della tastiera. Quando fu introdotto il "cestello", nel 1870, i fabbricanti si accorsero che, anche ai dattilografi meno veloci, accadeva di provocare la sovrapposizione (e conseguente incastro) dei martelletti porta-carattere. Ciò si verificava più frequentemente con parole come "ten" nelle quali le lettere più comunemente usate nella lingua inglese (che erano state posizionate per comodità l'una vicina all'altra sulla tastiera), venivano battute in rapida successione. Venne adottata la soluzione di allontanare il più possibile tra loro, nel cestello di battitura, quelle particolari lettere. Da quella scelta nacque la nuova tastiera standard QWERTY che fu progettata da Scholes e Gliden negli USA. Se, ancora oggi, le tastiere elettroniche conservano questa configurazione, è perché il suo formato è divenuto uno standard, di fatto universalmente accettato, ma ormai superato e, malgrado tutto, impossibile da cambiare.
Sono stati fatti alcuni tentativi per sviluppare tastiere di forme diverse: nel 1977 Lillian G. Malt sfruttò la flessibilità offerta da sistemi elettronici d'avanguardia per disegnare una tastiera ergonomica, chiamata Maltron, assai più comoda da usare rispetto a quella standard. Essa velocizza di molto il lavoro di battitura e permette di dattilografare senza sforzo 300 e più parole al minuto. Questo nuovo modello non ha, però, ottenuto gran successo, in un mercato troppo abituato alla tastiera formato QWERTY.
Una delle comodità di questa tastiera, del resto comune alla maggior parte dei microcomputer, è la possibilità di essere separata dall'unità centrale del sistema.
Nella maggior parte degli home computer, l'unità di visualizzazione e l'involucro del computer, che comprende anche la tastiera, sono separati e quest'ultimo è abbastanza piccolo da poter essere spostato comodamente sul piano di lavoro. Ciò non è vero per la maggior parte dei micro progettati in passato per uso commerciale. Negli ultimi tempi, comunque, vengono progettate tastiere poco ingrombranti, collegabili al microcomputer mediante un "cordone ombelicale". Il PC Junior della IBM costituisce un ulteriore passo in avanti: il collegamento tra la tastiera ed il microcomputer è simile a quello usato nei telecomandi per televisione e funziona a raggi infrarossi, senza alcun cavo.
Poiché l'ergonomia non è una scienza esatta (è, infatti, lo studio delle relazioni intercorrenti tra l'operatore ed il suo ambiente di lavoro, e queste tendono a cambiare rapidamente nel tempo), non è possibile stabilire delle regole fisse.
L'obiettivo sarà dunque quello di progettare ambienti che risultino confortevoli per un lungo periodo. Questo richiede una rielaborazione di apparecchi e di ambienti in modo che ogni energia possa essere dedicata al lavoro da compiere senza che sia necessario cambiare ripetutamente posizione o stancarsi eccessivamente.
Parlando del Lisa della Apple (vedere a pag. 261) abbiamo visto che esistono delle alterna-



MARTIN BURKE

Macchina stenografica

Quando è necessario verbalizzare un discorso "dal vivo", viene spesso impiegato un apparecchio chiamato Palantype, che permette una notevole velocità di battitura, usando una versione stenografica dell'ortografia fonetica



tive all'uso della tastiera, quando si lavora con software controllato da menu. Può essere interessante sperimentare versioni più economiche di questi programmi, che sfruttino un normale joystick e valutarne, quindi, i vantaggi. Naturalmente sarà necessario scrivere alcuni piccoli programmi di prova, ma ciò non dovrebbe costituire un ostacolo.

In alternativa, se il computer permette di ridefinire il valore associato a ciascun tasto, si può tentare di riprogrammare la tastiera, applicando sui tasti opportune etichette, che indichino i nuovi valori attribuiti. In questo caso, si potrebbero leggere (con delle PEEK) i valori degli 8 byte che definiscono un particolare carattere, trasferirli in una matrice di 8 elementi, modificarne i valori ed, infine, reinserirli (con delle POKE) nuovamente in memoria. È possibile inserire gli 8 byte che definiscono il nuovo carattere direttamente nelle locazioni occupate dai precedenti ma, adottando questa procedura, può esser necessario "salvare" la serie completa dei valori originali in una matrice temporanea, prima di depositare i nuovi. Inoltre, è utile archiviare (su nastro o su dischetto) il programma che effettua questa operazione, altrimenti, ogni volta che la macchina viene spenta (o "resettata"), ciascun tasto tornerà ad assumere il valore originale!

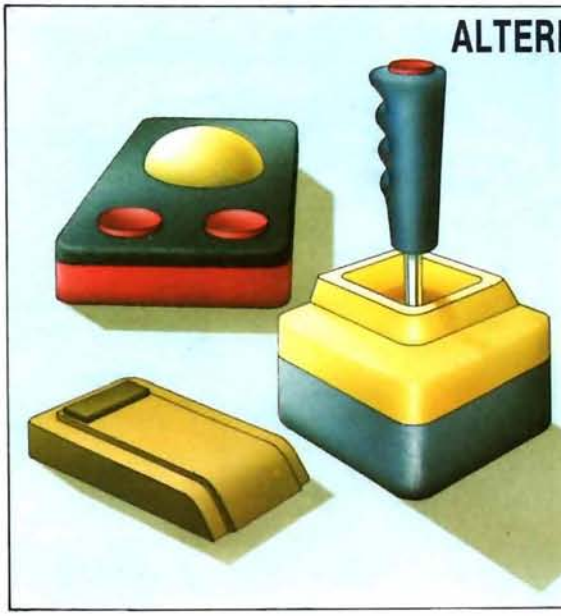
Infine, avendone la possibilità, potremmo farci costruire una scrivania su misura, con la tastiera incassata nel piano di lavoro ed il televisore o il monitor collocati ad angolazione adeguata. Comunque, esistono in commercio scrivanie progettate appositamente per i computer: esse dispongono di comodi ripiani, posti sotto il piano di lavoro, ove collocare le memorie di massa (unità a dischi o cassette); possiedono, inoltre, strutture passacavi per nascondere alla vista i cavi di collegamento del sistema.

L'ergonomia si basa soprattutto sul buon senso: talvolta, una scelta iniziale ponderata può evitare una lunga serie di fastidi in futuro (affaticamento della vista, dolori dovuti a posizioni di lavoro scomode e via dicendo).

A colpi... di tasto

Prima dell'invenzione delle tastiere elettroniche, i tasti della macchina da scrivere erano collegati meccanicamente ad un martelletto porta-caratteri, che imprimeva un simbolo sulla carta. Ciò imponeva limitazioni all'architettura della tastiera: i tasti di uso più comune dovevano essere lontani tra loro per evitare che i martelletti si urtassero. Sebbene simili problemi siano superati, tuttavia si usano ancora comunemente tastiere col formato QWERTY (il nome deriva dalle prime 5 lettere della tastiera); e quelle, come la Maltron, nella quale i tasti sono disposti con criteri più funzionali, non hanno ottenuto alcun successo

ALTERNATIVE FUTURE



Molti progettisti di computer, se potessero, farebbero volentieri a meno delle tastiere. I più moderni tipi di micro, dotati di un'ampia capacità di memoria e di una maggior velocità di elaborazione, fanno oggi uso di altre unità come joystick, "sfere" o topi, abbinati ad un software molto sofisticato

KEVIN JONES

Tracce e settori

Un sistema operativo deve sempre sapere dove trovare i dati memorizzati sui dischetti

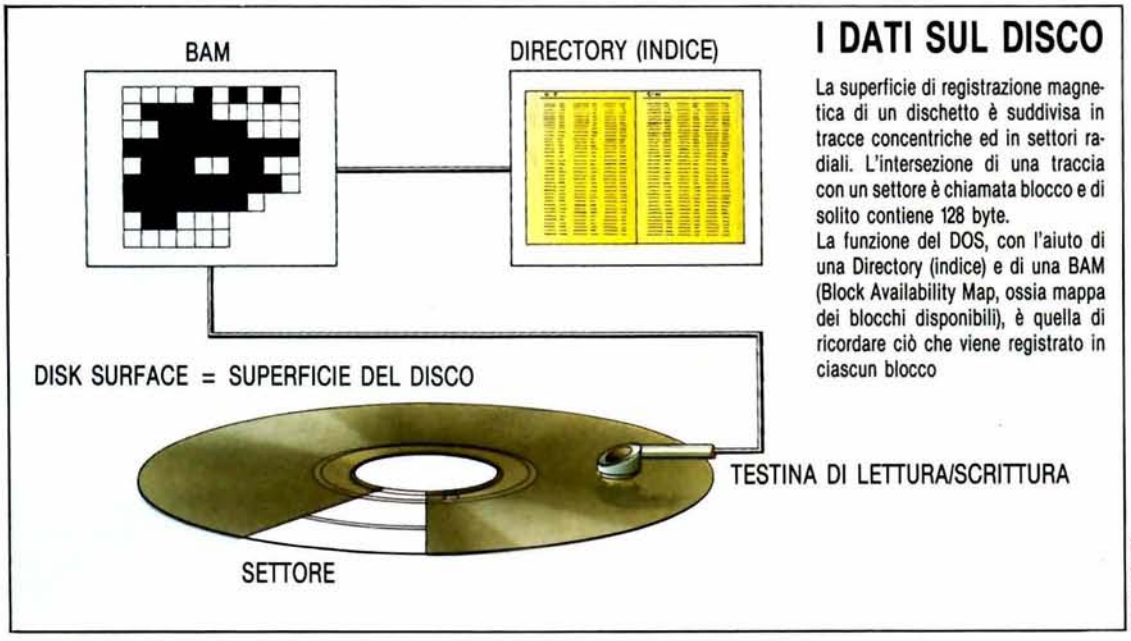
I programmi applicativi (ad esempio una Contabilità Generale, un "word processor", o anche un gioco) non rappresentano che la superficie "esterna" del software di un computer. Infatti, esiste tutta una serie di programmi, generalmente non accessibili all'operatore, necessari al controllo delle funzioni interne. L'insieme di questi programmi interni è chiamato Sistema Operativo (SO), e, nella maggior parte degli home computer, è contenuto nelle memorie ROM. Nei computer dotati di unità a dischi, gran parte del SO è impegnata nella gestione di questa unità ed il gruppo di routine che svolge questa funzione prende il nome di Sistema Operativo per Disco (Disk Operating System o, brevemente, DOS). Queste tre lettere sono spesso presenti nei nomi dei prodotti di varie case produttrici: un sistema operativo della Microsoft è, ad esempio, chiamato MSDOS. Il DOS viene fornito in tre forme diverse. Può essere contenuto in una parte della ROM all'interno del computer, come, ad esempio, nello Spectrum della Sinclair, che possiede al suo interno i comandi per gestire i Microdrive (un apparecchio simile ad un'unità a dischi).

Nella seconda forma, il DOS è contenuto nella ROM all'interno della stessa unità a dischetti. Questa configurazione è utilizzabile solo quando si tratta di un'unità a dischi "intelligente" (come, ad esempio, quella della Commodore), dotata, cioè, di un proprio microprocessore, di ROM e di RAM. Questi apparecchi sono molto più costosi da fabbricare, rispetto alle unità normali, ma offrono considerevoli vantaggi. In primo luogo, non occupano la memoria del computer (sempre preziosa per l'utente) ed, in secondo luogo, possono eseguire complesse operazioni sui dischi mentre, simultaneamente, il computer lavora con i programmi applicativi.

Nella terza forma, il DOS si trova nella RAM del computer. Questa configurazione è la più diffusa nei sistemi gestionali, nei quali le unità a dischi sono inserite all'interno del computer e c'è una notevole quantità di RAM disponibile (tipicamente, di 128 Kbyte). Per il fabbricante, questo tipo di sistemi offre il vantaggio di eliminare la necessità di creare nuove serie complete di ROM, qualora il DOS venga modificato, anche di poco. Inoltre, l'utente ha la possibilità di scegliere tra più Sistemi Operativi in grado di funzionare sullo stesso hardware.

A questo punto potrebbe sorgere un dubbio: se il DOS, che serve al computer per "dialogare" con l'unità a dischi, è memorizzato sul dischetto, come è possibile trasferirlo dal disco alla RAM interna al computer? Un programma non può caricarsi in RAM da solo!

È ovvio che deve esserci un piccolo programma, nelle ROM, eseguito ogni qual volta la macchina viene accesa. Questo programma

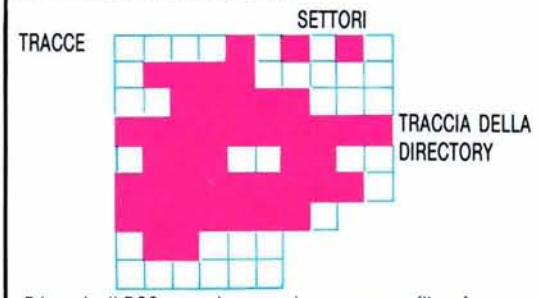


I DATI SUL DISCO

La superficie di registrazione magnetica di un dischetto è suddivisa in tracce concentriche ed in settori radiali. L'intersezione di una traccia con un settore è chiamata blocco e di solito contiene 128 byte.

La funzione del DOS, con l'aiuto di una Directory (indice) e di una BAM (Block Availability Map, ossia mappa dei blocchi disponibili), è quella di ricordare ciò che viene registrato in ciascun blocco

LA DISTRIBUZIONE DEI DATI



Prima che il DOS possa immagazzinare un nuovo file e fare una nuova registrazione nella Directory, deve consultare la BAM o Lista dei Settori Liberi. È questa una sezione della memoria in cui ciascun bit corrisponde ad un blocco del dischetto. Il numero binario 1 indica che il blocco è occupato, il numero 0 indica che è libero e può essere impiegato (in figura i quadrati rossi e bianchi). Notare che le tracce più interne (alla base della mappa) hanno meno settori perché sul dischetto sono molto più corte

si chiama "bootstrap" (letteralmente "laccio da scarpone") ed è una versione molto elementare di DOS. La funzione del bootstrap è proprio quella di cercare il DOS sul disco e di trasferirlo, un byte alla volta, nella RAM. Terminato il trasferimento, il controllo passa al DOS principale. Questo processo di accensione del computer e di caricamento del DOS viene chiamato "booting-up" (una specie di "calcio d'inizio"). Al suo termine compare sul video una frase di saluto e quindi un "prompt", ossia un simbolo che indica la disponibilità del computer a ricevere uno o più comandi dall'utente.

Abbiamo già visto (a pagina 114) che un dischetto è suddiviso in anelli concentrici chiamati tracce, divisi, a loro volta, in settori. L'intersezione di una traccia con un settore, chiamata blocco, contiene, solitamente, 128 byte di dati ed è il più piccolo insieme che l'unità a dischetti può leggere o scrivere.

Qualunque forma il DOS assuma in un sistema, il suo compito principale è quello di gestire le locazioni o indirizzi contenuti nei dischetti.

Per facilitare questa operazione, il DOS gestisce una "Directory" (indice) del dischetto. Spesso, questa è situata nella traccia centrale del disco, in quanto viene continuamente consultata ed una simile posizione minimizza i tempi necessari allo spostamento della testina di lettura/scrittura. La velocità operativa di un disco dipende, perciò, molto più dal tempo impiegato dalla testina a spostarsi da una traccia all'altra che dalla velocità di rotazione del disco stesso.

La directory è una lista di tutti i file (siano essi programmi o dati) presenti sul dischetto. In essa sono presenti anche delle indicazioni che si riferiscono al nome ed al tipo di file ed alla lista dei blocchi (ciascuno specificato da traccia e settore) nei quali un particolare file è stato memorizzato. Talvolta vengono conservate anche altre informazioni, quali, ad esempio, la data in cui è stata effettuata l'ultima copia di sicurezza del file, oppure una lista degli utenti che possono accedere ad un particolare file.

Quando è necessario archiviare un nuovo file, il DOS deve innanzi tutto consultare una tabella, chiamata Lista dei Settori Liberi o Mappa dei Blocchi Disponibili (Block Availability Map o, brevemente, BAM). In questa tabella, a ciascun blocco del disco corrisponde un bit. Se il blocco è "occupato" il valore del bit è 1, altrimenti è 0. In alcuni computer (ovviamente, dotati di unità a dischetti) esiste uno speciale programma, che visualizza la BAM, permettendo di osservarne i cambiamenti durante l'archiviazione di un file. Quando l'operatore impartisce un comando per la cancellazione di un file, il DOS non spreca tempo a cancellare effettivamente il contenuto dei blocchi utilizzati per quel file, ma interviene semplicemente sulla BAM riportando a 0 i bit interessati. Ciò basta ad indicare che i dati contenuti in quei blocchi non interessano più e sono disponibili per nuove scritture.

Un'altra caratteristica di questo sistema è che i file non vengono immagazzinati in blocchi consecutivi, come invece ci si aspetterebbe. Supponiamo, ad esempio, che una traccia sia suddivisa in 12 settori, numerati da 1 a 12 in senso orario. I primi 128 byte di un programma possono venir registrati nel settore 1, i 128 successivi nel settore 7, i seguenti 128 nel settore 2 e così via. Ciò avviene perché è necessario un piccolo intervallo di tempo per il trasferimento di un blocco di dati contenuti nella memoria al dischetto. Se il DOS dovesse scrivere in settori consecutivi, dovrebbe attendere una rotazione completa del dischetto tra due registrazioni successive: ciò provocherebbe un rallentamento enorme sia nelle operazioni di scrittura che di lettura. Inoltre, c'è un altro inconveniente: a lungo andare, un dischetto sul quale i file vengono continuamente scritti, cancellati e riscritti, finisce con l'avere una BAM simile ad un pezzo di groviera, piena di buchi. Pertanto, i nuovi file devono essere distribuiti riempiendo questi spazi sparsi.

Un Sistema Operativo per Dischi svolge molte altre funzioni, compresa la formattazione di nuovi dischetti (cioè la creazione di tracce, settori e di una Directory vuota su un disco "vergine"), la produzione di copie di sicurezza ed il riordinamento di dischetti pieni. Le versioni di DOS più sofisticate dispongono anche di un assortimento di programmi interni, per la gestione delle informazioni (vedere a pag. 204).



CHRIS STEVENS

I disk-drive intelligenti

Alcune unità a dischetti sono dotate di un proprio microprocessore, di una ROM e di una RAM. Vengono allora chiamate unità "intelligenti" ed il DOS è incorporato nella ROM. Quando invece vengono usate unità "non intelligenti", il DOS viene immagazzinato all'interno del computer

Oltre i limiti

Anche un economico computer può evolversi in un sistema piuttosto sofisticato grazie a buoni accessori

Dal punto di vista economico, lo ZX81 della Sinclair, anche in versione base, offre forse il miglior rapporto costo/prestazioni rispetto a qualsiasi altro computer della medesima categoria. Comunque, esiste un numero sorprendente di unità aggiuntive, da collegare a questo apparecchio, che lo rendono notevolmente sofisticato. Queste unità permettono, tra l'altro, di ottenere una grafica a colori ad alta risoluzione, una buona sintetizzazione vocale, e una maggiore capacità di comunicazione. Dato il suo costo decisamente basso, questo computer presenta alcuni inconvenienti, che possono essere in gran parte eliminati con accessori quali tastiere standard professionali, estensioni di memoria RAM, e dispositivi di controllo programmabili per joystick.



Cartuccia RAM

Lo ZX81, nella versione di base, possiede solo 1 Kbyte di RAM, di cui 123 byte sono occupati da variabili di sistema. Di conseguenza, si avverte subito la necessità di un'espansione di memoria. La cartuccia RAM della Sinclair, mostrata in figura, viene venduta nel solo formato di 16 Kbyte, ma esistono altre cartucce, come la Cheetah, che contengono fino a 64 Kbyte

Accoppiatori acustici

Ci sono due tipi di modulatori/demodulatori (modem): quelli a collegamento diretto sulla linea telefonica e gli accoppiatori acustici come il Micro-Mite 60, qui mostrato, che utilizza la cornetta telefonica. I modem a collegamento diretto sono generalmente un po' più costosi, modulano e demodulano segnali elettronici che rappresentano, in forma binaria, le informazioni inviate o ricevute. Gli accoppiatori acustici, invece, trasformano gli 0 e gli 1 binari in segnali acustici, e viceversa, a seconda che si tratti di trasmissione o di ricezione di informazioni

Forth su ROM

I micro ZX della Sinclair dispongono di una versione BASIC piuttosto particolare. Non è possibile utilizzare una diversa versione di BASIC, ma si può cambiare completamente linguaggio scegliendo, ad esempio, il Forth. Ci sono due strade per raggiungere questo risultato: una consiste nel caricare in RAM il nuovo linguaggio a partire da una cassetta, il che comporta un costante ritorno al linguaggio BASIC quando il computer viene acceso o resettato; la seconda consiste nel sostituire la ROM del BASIC con un'altra ROM. La ROM del Forth ha grandi potenzialità: permette infatti di far girare sullo stesso computer 10 o più programmi contemporaneamente. Questa caratteristica può essere adeguatamente sfruttata soltanto in applicazioni di controllo, quando cioè più periferiche debbono essere programmate separatamente

E inoltre...

Oltre alle unità già viste, ci sono altri accessori che aumentano le capacità di lavoro dello ZX81, ad esempio: schede video a colori che permettono di ottenere, sullo schermo, ben 16 colori; generatori di suoni a 3 timbri vocali programmabili; porte bidirezionali per gestire fino a 16 unità di input/output e così via. Lo ZX81 è un home computer adatto per giocare ed imparare a programmare, ma può "crescere" ed esaltare al massimo le potenzialità del suo microprocessore, uno Z80

MARCUS WILSON-SMITH



Sintetizzatore vocale

Un altro accessorio interessante è il "Sweet Talker", un sintetizzatore di parole prodotto dalla Cheetah. Utilizza un sistema allofonico e per questo è più facile programmarlo rispetto ad apparecchi che utilizzano i fonemi (gli allofoni sono gruppi di suoni simili ai fonemi). Unità di questo tipo sono disponibili sia per lo ZX81 che per altri home computer

Hebot

La "tartaruga" Hebot, disponibile in scatola di montaggio o pronta per l'uso, è uno dei più sofisticati robot per home computer. Viene venduta completa di software e può essere dotata di svariati accessori, tra i quali dei fotosensori, che permettono al robot di seguire un percorso prestabilito seguendo delle "tracce" fatte incollando sul pavimento dei nastri adesivi riflettenti

Tastiera

La tastiera a membrana "multistrato" è forse la caratteristica meno positiva dello ZX81. Per questa ragione molte case produttrici offrono in alternativa tastiere convenzionali di dimensioni standard. La Mapsoft ZX81, qui in figura, prodotta dalla Maplin Electronics, è disponibile in scatola di montaggio o pronta all'uso. Oltre alla normale gamma di caratteri, la tastiera Mapsoft possiede tre tasti extra. Un'alternativa meno costosa può essere l'applicazione di una tastiera di adeguate dimensioni direttamente su quella dello ZX81. Essa permetterà di individuare più facilmente i tasti ma non produrrà altri effetti

Joystick e dispositivo di controllo

Poiché molti ZX81 vengono usati soprattutto con programmi di giochi, è abbastanza strano che la Sinclair non abbia prodotto dei joystick e relativi dispositivi di controllo. Tuttavia, in commercio si trova una vasta gamma di questi apparecchi, sia del tipo non programmabile (i movimenti indotti dal joystick sono predeterminati), sia del tipo programmabile (è possibile selezionare quali tasti il joystick deve simulare). Il modello in figura, prodotto dalla AGF Hardware, può essere programmato cambiando la disposizione dei cavetti di collegamento; altri tipi possono essere programmati direttamente dal computer. Lo ZX81 può funzionare con qualsiasi joystick ad interruttore come pure con "sfere guida"

Stampante ZX

Questa stampante usa carta rivestita da una sottilissima lamina d'alluminio sensibile all'elettricità. La testina non stampa in modo convenzionale, bensì asporta la lamina d'alluminio portando alla luce la superficie scura sottostante. La stampa è abbastanza veloce, ma le dimensioni e le particolari caratteristiche di questo tipo di stampa creano qualche inconveniente. Comunque, usando un'apposita scheda d'interfaccia, è possibile collegare stampanti di tipo normale. Sono inoltre disponibili schede per la porta RS232 e per interfacce Centronics

Una marcia in più

Consigli per migliorare l'efficienza e la velocità dei programmi in BASIC

Il BASIC, nonostante venga da più parti criticato, è pur sempre un linguaggio versatile ed un ottimo strumento didattico. Si possono scrivere programmi di ogni genere in BASIC, a condizione di avere sufficiente memoria a disposizione e che i tempi di esecuzione non siano particolarmente limitati. La lentezza del BASIC, soprattutto se si adopera un interprete e non un compilatore (vedere a pag. 184), si evidenzia soprattutto quando esso deve interpretare ed eseguire più volte una stessa sequenza di operazioni.
Le operazioni di riordino (vedere a pag. 286), per esempio, costituiscono un processo altamente ripetitivo, che viene generalmente ottenuto ricorrendo all'impiego di cicli FOR-NEXT nidificati. Dovendo riordinare 100 elementi, il programma, probabilmente, sarà costretto ad una serie valutabile tra 2500 e 5000 iterazioni. Un riordino, in BASIC, è sempre e comunque lento, ma si possono usare alcuni accorgimenti che possono influire anche notevolmente sui tempi d'esecuzione. Se una particolare sequenza d'istruzioni è ripetuta 5000 volte, e, modificandone l'ordine, si guadagna un centesimo di secondo a "giro", il risparmio totale di tempo è sull'ordine dei 50 secondi: tutt'altro che disprezzabile!
Per meglio dimostrare la differenza tra una codifica buona ed una meno buona, dovremo disporre di un meccanismo di temporizzazione e di un programma di prova. Chi possiede un computer Commodore, può utilizzare l'orologio incorporato (accessibile mediante le variabili TI$ e TI). Qualora non sia disponibile un orologio interno all'apparecchio, occorrerà un cronometro esterno. In tal caso, è conveniente far emettere un sonoro "bip" dal programma, quando l'esecuzione inizia e quando termina. La base per un programma di prova può essere:

```
1000 L = 500
2000 PRINT "* * INIZIO * *
2050 (inserire qui il "bip" sonoro)
2100 TI$ = "000000"
2200 FOR K = 1 TO L
 :
(serie di istruzioni da provare)
 :
2900 NEXT K : T9 = TI
2950 (inserire qui il "bip" sonoro)
3000 PRINT "* * FINE * *
3100 PRINT "Tempo richiesto: ";(T9/60)" secondi"
```

Le linee 2100 e 3100 valgono per i possessori di Commodore: con altri apparecchi occorre sostituire le equivalenti istruzioni per l'azzeramento e la lettura dell'eventuale orologio interno. Subito dopo la linea 2200 e fino alla 2900 viene inserita la sequenza d'istruzioni da cronometrare. Si noti che tutte le prove si riferiranno ad un numero di iterazioni L, fissato nella linea 1000. Si può modificare questo valore a piacimento, ma è consigliabile effettuare tutta una serie di prove con uno stesso valore di L.
Qui di seguito, riportiamo alcune regole generali, che permettono di scrivere programmi BASIC più efficienti:

1. Evitare complessi calcoli nei cicli

L'elevazione a potenza (x^3 significa x elevato alla terza, per esempio) ed altre funzioni matematiche (seno, coseno e tangente, per esempio) sono particolarmente lente. La moltiplicazione e la divisione sono più lente dell'addizione e della sottrazione, ma anche la più veloce di queste operazioni (la somma) è relativamente lenta.
Si provi ad inserire, nel programma di prova, queste linee:

```
900 Z = 1.1
2300 X = Z↑3
```

e si lanci l'esecuzione. Un risultato tipico, con 500 iterazioni, è 27.95 secondi. Ora si sostituisca la linea 2300 con:

```
2300 X = Z*Z*Z
```

Un lancio del programma farà subito notare che, nonostante il risultato dell'operazione sia identico, il tempo impiegato, questa volta, è di circa 3 secondi e mezzo!
Ci si può spingere a ricercare il livello oltre il quale non conviene più adottare una simile sostituzione all'elevazione a potenza: in genere, questo si aggira sulla diciottesima potenza (ossia x elevato alla 18). Tuttavia, non è sempre possibile usare questo stratagemma, in quanto per ottenere X elevato alla 2,3 una moltiplicazione multipla non basta, mentre l'elevazione a potenza del BASIC prevede esponenti reali ed anche negativi.
Si provi ad usare il programma di prova per altre funzioni matematiche.

2. Si usino delle variabili piuttosto che delle costanti numeriche

Ogni volta che il BASIC incontra una costante numerica (7.280, per esempio), viene sprecato un sacco di tempo nella conversione del numero in una forma adatta al computer. Si inserisca, nel programma di prova:

```
2300 X = X+7280
```

Questa forma richiede circa 4,63 secondi, con 500 iterazioni, ma la forma:

```
900 C = 7280
2300 X = X + C
```

impiega circa metà tempo, a parità d'iterazioni.

3. Se devono essere impiegate istruzioni GOTO, si cerchi di saltare "in avanti" nel programma, mai "all'indietro"

Lo stesso vale per le GOSUB. Quando il BASIC incontra una GOTO o una GOSUB, l'interprete confronta il numero di linea richiesto con quello della linea in corso d'esecuzione. Se quest'ultimo è inferiore a quello richiesto, la ricerca parte dal numero di linea "corrente" in avanti, ma, in caso contrario, il BASIC spesso riparte dalla prima linea del programma. Ne consegue che la collocazione migliore, per le subroutine, è all'inizio o alla fine di un programma. Si provi ad inserire 56 linee contenenti la sola istruzione REM all'inizio del programma di prova e quindi si aggiunga:

```
2300 GOTO 2400
2400 GOTO 2500
2500 GOTO 2900
```

Ciò richiede circa 2,33 secondi (500 iterazioni). Ma

```
2300 GOTO 2500
2400 GOTO 2900
2500 GOTO 2400
```

richiede circa il doppio del tempo.

4. Si inizializzino le variabili in ordine di frequenza

I nomi delle variabili sono conservati dall'interprete nell'ordine col quale compaiono, per la prima volta, all'interno del programma. Quindi, accedere all'ultima variabile nell'elenco richiede tempi più lunghi. Per la stessa ragione conviene riutilizzare nomi di variabili già definite precedentemente nel programma, ma il cui valore non è più utile, piuttosto che definire una nuova variabile.
Inoltre, conviene definire per prime le variabili adoperate all'interno di cicli nidificati, eventualmente con un valore fittizio. Ad esempio:

```
1000 L = 500 : C = 7280 : X = 0 : Z = 1.1
2300 A = 0
```

richiede 2.2 secondi per 500 iterazioni, mentre:

```
1000 A = 0 : L = 500 : C = 7280 : X = 0 : Z = 1.1
2300 A = 0
```

richiede 2.06 secondi.

5. Si eviti l'uso di stringhe

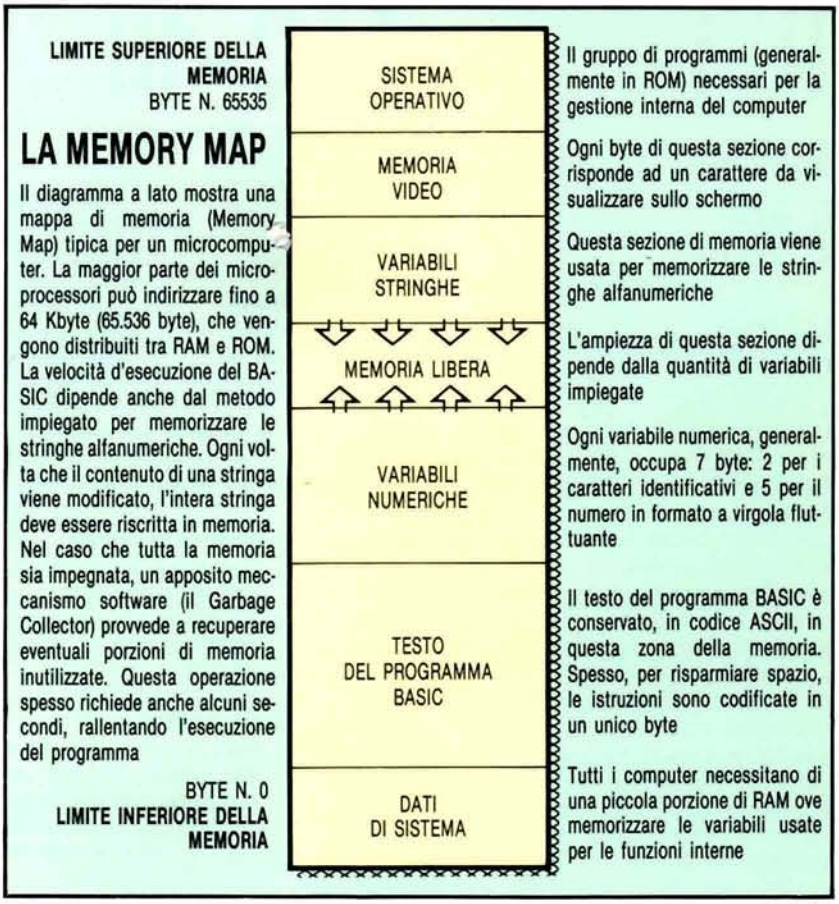
Le operazioni necessarie al trattamento delle stringhe (variabili alfanumeriche) consumano grandi quantità di memoria, ed un meccanismo interno al BASIC, il Garbage Collector (letteralmente: il raccoglitore di rifiuti), viene periodicamente usato dall'interprete per riordinare il contenuto della memoria riservata alle stringhe. Più sono le stringhe, maggiore è il tempo necessario all'operazione.
È difficile portare un esempio concreto e di applicazione generica, poiché il sistema di gestione della memoria varia considerevolmente da computer a computer. Il metodo da seguire, comunque, consiste nel riempire la maggior parte della memoria con dati (ad esempio una grossa matrice) e quindi eseguire alcune operazioni su variabili alfanumeriche. Un esempio potrebbe essere:

```
1000 L=500:DIM T$(L)
1100 FOR K=1 TO L
1200 T$(K)="A"+"B"
1300 PRINT K
1400 NEXT K
```

La DIM iniziale provvede a diminuire sensibilmente la quantità di memoria disponibile (all'occorrenza, se il valore di L è troppo basso, questo può essere aumentato). La PRINT, eseguita ad ogni iterazione, visualizza il valore del contatore del ciclo. Durante l'esecuzione, si possono notare delle pause, durante le quali entra in funzione il Garbage Collector. Il programma continua con:

```
2300 A$ = LEFT$(T$(L),1):B$ = A$ + RIGHT$(T$(L),1)
```

e questo richiede circa 30 secondi per 500 iterazioni. Eseguendo il programma di nuovo, ma senza riempire preventivamente la memoria, il ciclo impiega circa 9 secondi.



LIZ DIXON

Il Tandy MC-10

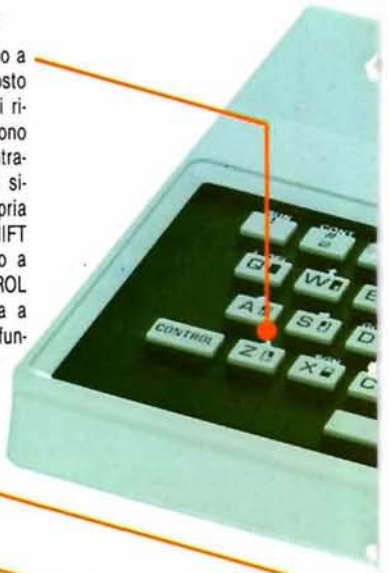
Anatomia di un microcomputer a diffusione europea: tecnologie a confronto

Il Tandy MC-10 è un piccolo e compatto computer dalle buone prestazioni, pur usando un chip poco sofisticato. La tastiera ha tasti a pulsante leggermente più grandi di quelli di altre tastiere della medesima categoria e possiede anche una vera e propria barra spaziatrice. Altre caratteristiche rendono questa macchina piuttosto facile da usare: è possibile, ad esempio, inserire comandi del linguaggio BASIC premendo contemporaneamente il tasto CONTROL ed il tasto di funzione richiesto. All'accensione, la macchina seleziona automaticamente il funzionamento "maiuscolo", ma è possibile alternare i caratteri "maiuscolo" o "minuscolo" con la pressione contemporanea dei tasti SHIFT e 0. Lo schermo è piuttosto piccolo, possiede solo 13 righe di 32 caratteri e permette l'uso di una grafica a bassa risoluzione e di media qualità. Le variazioni cromatiche sono limitate, ma la qualità del colore è molto buona. Sorprendentemente, il display non è capace di visualizzare le lettere maiuscole, che, sebbene accettate regolarmente da tastiera, compaiono sul video in maiuscolo con contrasto invertito. I testi possono essere in verde su fondo nero o viceversa; i caratteri grafici possono avere 9 differenti colori, ma le lettere o il fondo devono essere neri, per cui è impossibile ottenere un'immagine blu su fondo rosso, anche usando il funzionamento grafico!

Anche le capacità di suono mostrano alcuni limiti. Esiste un solo canale audio, che permette solo minime variazioni di tonalità e di durata. L'input e l'output previsti riguardano l'unità di registrazione a cassette (compreso un controllo a distanza), il televisore ed una porta seriale RS232.

Quest'ultima può essere usata come linea di trasmissione per comunicare con un altro computer o, in alternativa, per collegare una stampante. Inoltre, la si può utilizzare anche per realizzare una rete di collegamento con altri Tandy MC-10. Risulta evidente che l'apparecchio non è stato progettato per un uso essenzialmente ricreativo: non sono previsti connettori per joystick o paddle, né alcuna particolare grafica o chip di controllo del suono, presenti, invece, su gran parte dei computer della stessa categoria.

Tuttavia sono previste, in futuro, alcune possibilità d'espansione, dato che all'interno della macchina si può notare un "bus" di sistema (abbastanza misterioso in quanto non



La tastiera del Tandy MC-10
I tasti, di plastica rigida, sono a pulsante e funzionano piuttosto bene. I caratteri ed i simboli riportati su di essi, però, tendono a scomparire con l'uso. Contrariamente ad altri apparecchi simili, esiste una vera e propria barra spaziatrice. Il tasto SHIFT è unico e si trova in basso a destra, mentre il tasto CONTROL è piuttosto largo e si trova a sinistra in posizione molto funzionale. L'impressione generale è quella di una tastiera piuttosto confortevole

Pulsante di RESET
È grande e di color rosso chiaro, quindi facilmente individuabile. La sua posizione è tale da evitarne l'uso accidentale

Interfaccia per cassetta
I piedini 1 e 3 dello spinotto DIN a 5 piedini permettono il controllo a distanza. Il segnale di input è sul 4, l'output sul 5 e la "massa" sul 2

Interfaccia RS232
È uno spinotto DIN a 4 piedini. Il segnale di controllo è sul piedino 1, l'input dati è sul 2, la "massa" sul 3 e l'output dati sul 4

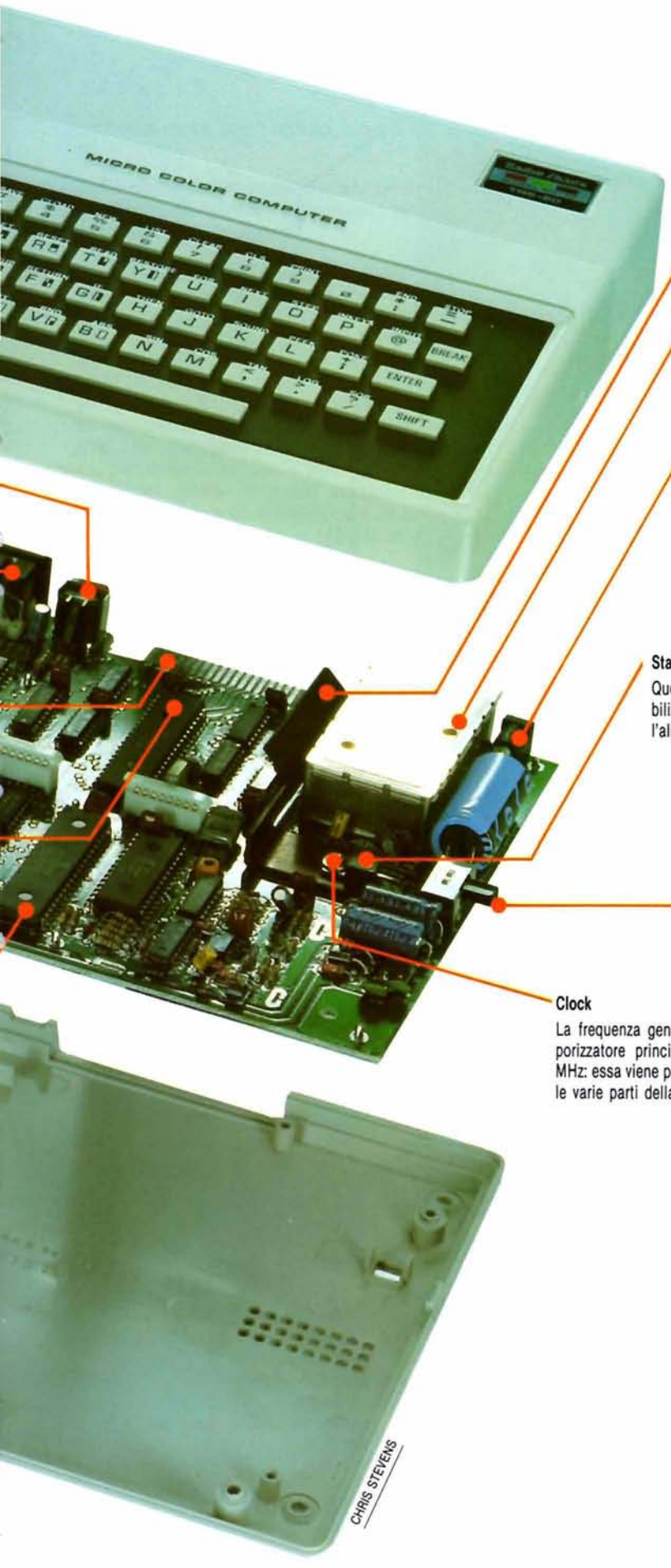
Bus di sistema
Non è descritto nel manuale ma verrà utilizzato, in futuro, con unità d'espansione non precisate. I canali sono comunque sufficienti per controllare anche apparecchiature complesse

ROM
Il chip della ROM è ben saldato alla scheda e, quindi, non è facilmente sostituibile con ROM alternative o di maggiori capacità. Il BASIC Microsoft occupa 8 Kbyte di ROM

La CPU
Il computer MC-10 è dotato di un microprocessore piuttosto insolito, il 6803. Questo processore fa parte di una vecchia famiglia, ma non è famoso quanto il 6502 o lo Z80. È comunque un buon chip a 8 bit con una discreta gamma d'istruzioni

RAM statica
I 4 Kbyte di RAM sono contenuti in questi 2 chip di RAM statica di 2 Kbyte x 8 bit e vengono in parte usati per la gestione del video e per alcune variabili di sistema

6847 VDP
Il video è controllato dal chip MC 6847, un Video Display Processor. Questo chip è identico a quello usato sul Dragon 32 e (almeno in teoria) può essere programmato per differenti formati video



Dissipatore
Il transistor stabilizzatore di potenza, un Triac, scalda molto durante il funzionamento ed il calore viene dissipato da questa larga piastra di metallo

Modulatore TV
Converte il flusso di dati prodotto dalla circuiteria del video in un segnale TV per il Canale 36. Non è prevista un'uscita audio ed, inoltre, questo è il solo output video disponibile: non esiste una presa per monitor

Spinotto di alimentazione
È una normale presa coassiale a basso voltaggio. Come tutti i computer di questo tipo, il Tandy MC-10 riceve corrente da un piccolo trasformatore a basso voltaggio collegato ad una presa di corrente

Stabilizzatore
Questo transistor di potenza stabilizza la tensione necessaria all'alimentazione dei circuiti

Interruttore d'accensione
Per il RESET del sistema esiste l'apposito pulsante, quindi questo interruttore viene esclusivamente impiegato per accendere o spegnere il computer

Clock
La frequenza generata dal temporizzatore principale è di 4,4 MHz: essa viene poi adattata per le varie parti della macchina

CHRIS STEVENS

TANDY MC-10

DIMENSIONI
mm. 210 x 178 x 51

PESO
—

CPU
6803

CLOCK
4,4 MHz

MEMORIA
4 Kbyte di RAM e 8 Kbyte di ROM

VIDEO
16 linee di 32 caratteri. 9 colori disponibili, 75 caratteri predefiniti

INTERFACCE
Connettore per TV, per registratore a cassette, porta seriale RS232, "bus" di sistema (attualmente non usato)

LINGUAGGI FORNITI
BASIC

ALTRI LINGUAGGI DISPONIBILI
Nessuno per il momento

ACCESSORI COMPRESI
Cavo di connessione per TV, un manuale operativo ed uno sul BASIC

TASTIERA
48 tasti del tipo a pulsante

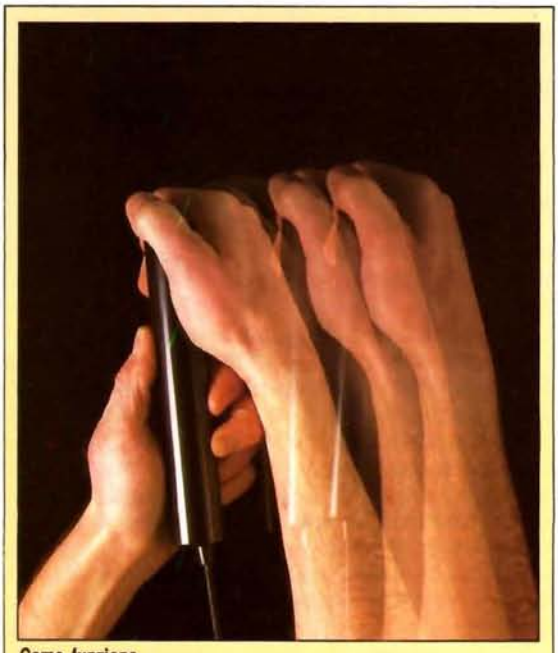
DOCUMENTAZIONE
La documentazione di accompagnamento è chiara e ben impostata, ma non fornisce molte informazioni tecniche. Si nota la mancanza di un indice, che faciliterebbe la ricerca di argomenti specifici. Una scheda riassuntiva sul BASIC permette, a chi ha un po' d'esperienza, di mettersi immediatamente al lavoro sul computer

documentato), che confluisce in un connettore, protetto da un cappuccio metallico. Il manuale descrive molto sommariamente questa struttura: «Il connettore è riservato a future espansioni di memoria». Non vengono fornite ulteriori informazioni sul tipo di espansioni che sarà possibile utilizzare su di esso. La documentazione dell'apparecchio MC-10 è quella che la Tandy solitamente mette a disposizione per le sue macchine: una serie di descrizioni piuttosto concise, ma, tutto sommato, un testo abbastanza omogeneo e privo di pause.
Come macchina a basso costo, merita una certa considerazione, anche se la memoria RAM di 4 Kbyte (parte della quale usata per funzioni interne) è piuttosto scarsa.

I joystick della nuova generazione

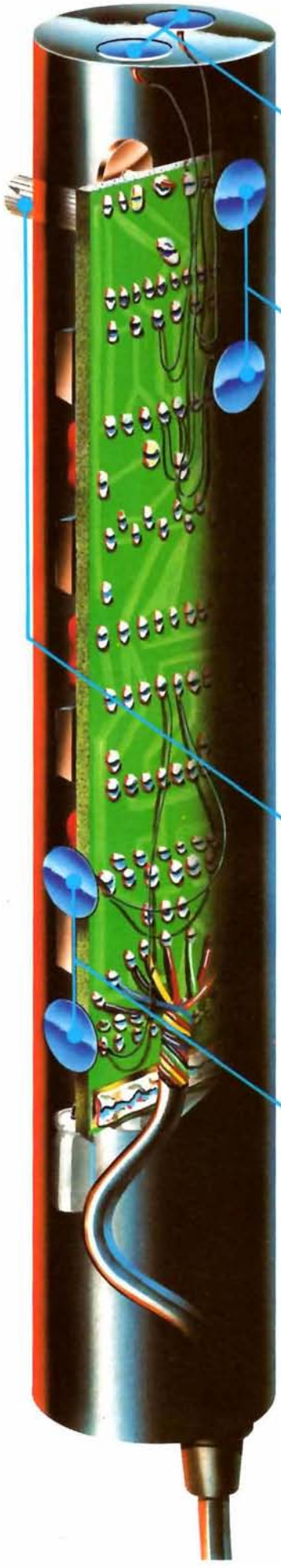
Le più recenti innovazioni tecniche applicate a questi utili accessori

L'industria dei personal computer è soggetta a rapidi sviluppi tecnologici. Le innovazioni non sono limitate esclusivamente alle unità centrali: anche le periferiche e gli accessori migliorano continuamente. Il principio di funzionamento del joystick meccanico è già stato esaminato in precedenza (vedere a pag. 56); adesso vedremo come funzionano due tipi di joystick fra i più sofisticati.
Un apparecchio chiamato "Le Stik" è stato il primo joystick analogico a non utilizzare i classici meccanismi di segnalazione. Il Le Stik si compone di una manopola sagomata, dotata di un bottone di fuoco posto nella sua parte superiore e di un "tasto di pausa" situato lateralmente. Diversamente dai joystick fissati su supporti da tavolo, il Le Stik viene tenuto in mano semplicemente orientato nella direzione desiderata: questi movi-



Come funziona
Il Trickstick sfrutta il "ronzio di rete" indotto sul corpo umano (ossia la radiazione elettromagnetica generata dall'impianto elettrico). Il nostro corpo si comporta come un'antenna ed i sensori posti sul cilindro rilevano con maggiore o minore intensità, a seconda della pressione esercitata su di essi dalle dita

IL TRICKSTICK



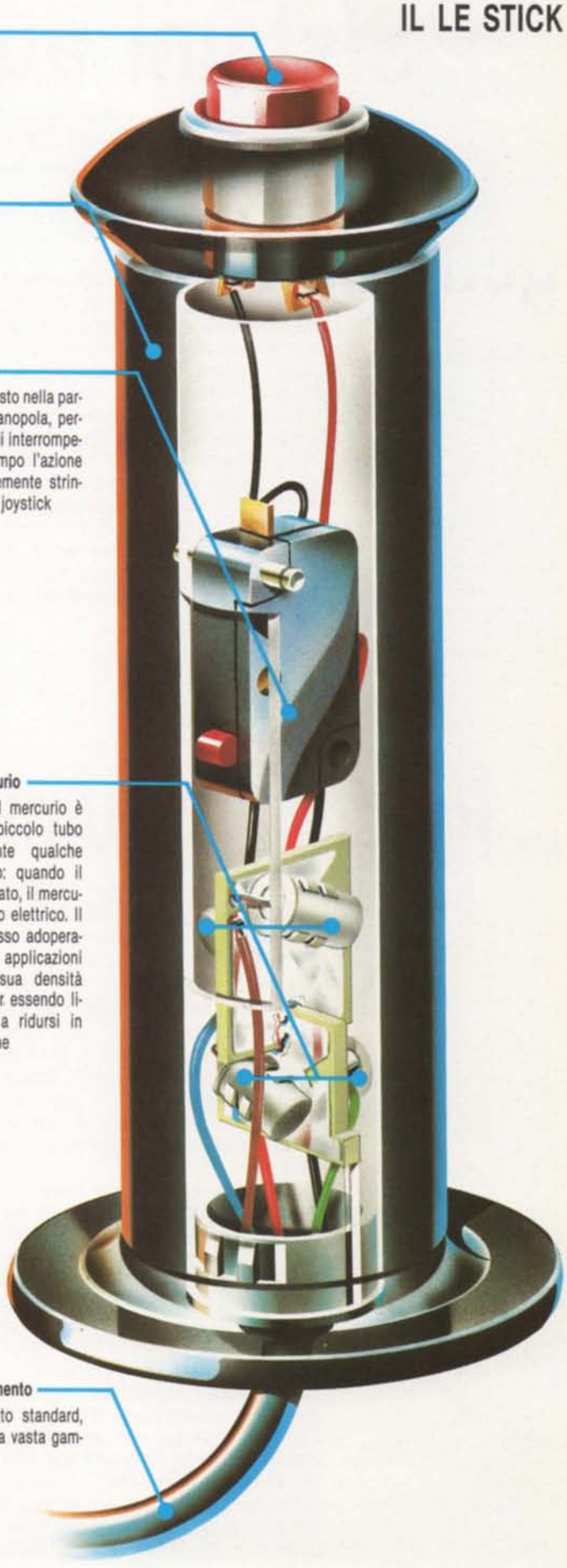
Controlli del movimento orizzontale
Spostando il pollice da un sensore all'altro è possibile controllare i movimenti in avanti e all'indietro

Controlli del movimento verticale
Il sensore superiore controlla i movimenti verso l'alto, quello inferiore i movimenti verso il basso

Controllo di sensibilità
Questa piccola manopola permette di "sintonizzare" il Trickstick alla sensibilità del corpo del giocatore rispetto al "ronzio di rete"

Bottoni di fuoco
Ciascun sensore genera un segnale indipendente: è quindi possibile lanciare bombe premendo un bottone e contemporaneamente azionare il cannone laser con l'altro

IL LE STICK



menti, rispetto alla verticale, provocano movimenti corrispondenti di una figura (ad esempio, un'astronave) rappresentata sullo schermo.
Il meccanismo all'interno del Le Stik si compone di 4 piccoli involucri cilindrici, chiusi ermeticamente e contenenti mercurio. Quando il joystick viene inclinato rispetto alla verticale, il mercurio fluisce nella stessa direzione e chiude uno o più contatti elettrici, proprio come se fosse stato premuto un interruttore. Riportando la manopola in posizione verticale, il mercurio fluisce all'indietro nei tubi riaprendo così il circuito. La risposta del sistema è notevolmente più veloce di quella indotta dai joystick di vecchia concezione. Anzi, spesso la sensibilità del sistema risulta addirittura eccessiva, se il gioco che si sta usando è stato programmato per joystick di tipo convenzionale.
Un altro procedimento molto interessante, per convertire i movimenti del joystick in segnali interpretabili da un computer, viene usato dal "Trickstick", prodotto dalla East London Robotics. Questo apparecchio è l'unico ad utilizzare un fenomeno d'induzione elettrica. Il principio è abbastanza semplice: si utilizza il corpo umano come antenna, per captare il "ronzio di rete" (l'innocua radiazione elettromagnetica prodotta in ogni casa dall'impianto elettrico). Il Trickstick si compone di un tubo sigillato, rivestito da un astuccio di plastica, che viene tenuto in posizione verticale con entrambe le mani. Sulla superficie del tubo sono visibili tre paia di sensori: il primo paio controlla i movimenti in avanti e all'indietro, il secondo quelli verso l'alto ed il basso, il terzo paio costituisce i bottoni di fuoco.
Il ronzio di rete, captato dal corpo, viene rilevato da questi sensori e trasmesso alla circuiteria interna al joystick, ove gli impulsi vengono convertiti in segnali che comunicano al computer le informazioni direzionali. A questo punto, i segnali vengono interpretati per visualizzare sullo schermo l'intero movimento. Premendo il pulsante con maggior energia si ottiene un segnale più forte e quindi una risposta più rapida da parte del computer. Il Trickstick quindi combina il controllo proporzionale caratteristico del joystick analogico al rapido e diretto controllo digitale, tipico di un'unità ad interruttori. Poiché ogni giocatore interagisce con i circuiti in modo diverso, occorre regolare la sensibilità del Trickstick e ciò si ottiene intervenendo su di una piccola manopola posta ad un'estremità del joystick.

Sistemi audio

Qualche suono sul Vic-20

Nell'ultimo articolo della serie Suono e Luce abbiamo esaminato il Vic-20 ed abbiamo imparato a controllare i 3 oscillatori della macchina intervenendo con il comando POKE sulle locazioni di memoria, a selezionare i livelli di volume ed a definire la durata di un suono. Abbiamo inoltre visto come fosse possibile determinare la durata delle pause tra i suoni usando dei cicli FOR...NEXT, o (con migliori risultati) sfruttando il temporizzatore del Vic-20, che scandisce il tempo in jiffy (sessantesimi di secondo). Il controllo di questi 3 elementi musicali, frequenza, volume e tempo, permette di creare sul Vic-20 semplici motivi musicali e quindi di produrre buoni effetti sonori.

Come suonare motivi musicali

Per creare un motivo musicale occorre, innanzitutto, annotarsi la sequenza delle note necessarie. Ad esempio, le prime note della canzone *Oh, I do like to be beside the seaside*. Esse possono essere selezionate, secondo il giusto ordine, nel seguente modo:

D#	E	F	D#	C	A#	G#	G	G#	D#	D#
Re#	Mi	Fa	Re#	Do	La#	Sol#	Sol	Sol#	Re#	Re#

Usando la procedura descritta a pag. 284, è possibile definire la durata dei suoni e delle pause mediante la variabile interna TI. Sarà quindi possibile far suonare al computer il motivo, inviando in esecuzione il seguente programma (notare l'uso di variabili per semplificare la selezione delle istruzioni POKE):

```
10 V = 36878
20 FOR I=1 TO 11
30 READ N: REM LETTURA DELLA NOTA
40 POKE V,7:P=TI:REM ★VOLUME★
50 IF TI-P < 15 THEN 50: REM ★PAUSA★
60 POKE V-3,N:D=TI: REM ★EMISSIONE NOTA★
70 IF TI-D < 20 THEN 70: REM ★DURATA★
80 POKE V-3,0: REM ★FINE EMISSIONE★
90 NEXT I
100 DATA 203,207,209,203: REM ★VALORI DELLE NOTE★
110 DATA 195,187,179,175
120 DATA 179,203,203
130 POKE V,0: REM ★AZZERAMENTO VOLUME★
140 END
```

... Programmi luminosi

Capacità grafiche del Micro BBC

Sul Micro BBC bastano poche righe di programmazione BASIC per realizzare effetti grafici di ottima qualità. Inoltre, le immagini compaiono sul video con estrema rapidità ed esistono molti comandi per l'alta risoluzione, ivi incluse le istruzioni per tracciare linee rette, punti e per disegnare e colorare triangoli.
Quest'ultima funzione permette di colorare qualsiasi immagine, sebbene non sia disponibile un comando PAINT, procedendo però per piccoli triangoli di video. Mancano comandi BASIC per tracciare cerchi ed ellissi e la possibilità di usare "sprite". Usando un programma in BASIC, è inoltre possibile intervenire su quella parte del Sistema Operativo che controlla la visualizzazione delle immagini, mediante l'apposita funzione VDU. È infatti possibile suddividere il video in "finestre" per testi e grafica: ciò permette all'utente di ripartire le immagini da visualizzare in sezioni separate per testi e grafica. Ciascuna finestra può essere caratterizzata da colori propri ed il suo contenuto potrà essere cancellato indipendentemente dalle altre.

Modi di visualizzazione

Il Micro della BBC possiede otto modi grafici, tre dei quali specifici per la visualizzazione di testi. A seconda del modo scelto, è possibile ottenere sul video righe di 20, 40 o 80 caratteri e, sempre in dipendenza dal modo selezionato, si potranno adoperare 2, 4 o 16 colori. Un'interessante caratteristica è che i 2 o 4 colori utilizzabili, non sono fissi, ma possono essere variati dal programmatore scegliendo tra i 16 a disposizione.
Il modo grafico 7 si differenzia da tutti gli altri perché non si basa sulla gamma di caratteri standard ASCII e sui codici ad essa associati. Infatti le immagini vengono create sul video utilizzando i caratteri Teletext. In questo modo di funzionamento non è consentito usare gli abituali comandi PLOT e DRAW.
La seguente tabella mostra i tipi di risoluzione e le possibilità di colore specifiche per ogni modo:

Modo	Testo	Grafica	Colori
0	80×32	640×256	2
1	40×32	320×256	4
2	20×32	160×256	16
3	80×25		2 (bianco e nero)
4	40×32	320×256	2
5	20×32	160×256	4
6	40×25		2 (bianco e nero)
7	40×25		Teletext

Questo programma suona semplicemente le note nella giusta sequenza mantenendo uguale il tempo di durata di suoni e pause. Di conseguenza si ottiene un motivo musicale piuttosto piatto e artificioso. Dopo vari tentativi sarà possibile scrivere programmi più complessi che permettano di variare i tempi di durata dei suoni e delle pause.

Effetti sonori

Usando due o tre oscillatori è possibile far suonare semplici accordi. Il programma precedente suonava l'accordo di re maggiore (fa#, la e re) iniziando con fa# ed assommando poi la e re ad intervalli di un secondo. L'accordo continuava per altri 2 secondi.

```
10 POKE 36878,7
20 POKE 36874,233:D=TI
30 IF TI-D <60 THEN 30
40 POKE 36875,219:D=TI
50 IF TI-D < 60 THEN 50
60 POKE 36875,147:D=TI
70 IF TI-D <120 THEN 70
80 POKE 36878,0: POKE 36874,0
90 POKE 36875,0: POKE 36876,0
100 END
```

È comunque possibile aggiungere un gruppo di nuove istruzioni che rendano il timbro di queste note più piacevole. Si potrà, ad esempio, variare il volume per la durata di una nota, aumentandolo o diminuendolo a seconda dei valori attribuiti ad una variabile.

```
100 V = 36878
110 FOR I = 1 TO 12
120 POKE V,I
130 NEXT I
140 POKE V,0
```

Questo gruppo d'istruzioni farà sì che il volume aumenti un "passo" alla volta fino ad un massimo di 12, laddove il campo totale è compreso tra 0 (spento) e 15 (volume massimo). Un effetto di "tremolo" si ottiene alternando, nella variabile, valori di alto e basso volume. Allo stesso modo si può variare la frequenza (effetto "vibrato") e provocare l'abbassamento di un suono cambiando nel seguente modo l'istruzione 120:

```
120 POKE V-3,203+1
```

Val la pena di tentare anche diverse combinazioni di rumore, frequenze d'oscillazioni e volume. Sia facendo musica, che corredando i giochi con effetti sonori, lo scopo è quello di ridurre il senso di noia provocato dalla monotona ripetizione di suoni sempre uguali.
Abbiamo visto come sia possibile, con un poco d'attenzione, ricavare semplici melodie ed accordi sfruttando la generazione di suoni sul Vic-20, nonostante esso sia privo di comandi specifici per questa funzione. Ciò implica la realizzazione di programmi relativamente lunghi, che impediscono all'interprete BASIC di elaborare il codice tra le note con sufficiente rapidità. La via più semplice per eliminare questo problema è acquistare uno dei numerosi pacchetti di software in commercio dotati di istruzioni di comando aggiuntive, che facilitano le creazioni musicali. La cartuccia Super Expander della Commodore dispone di un'ampia ed utile gamma di comandi per il suono e di procedure che facilitano l'archiviazione dei brani creati. Altri modelli, come il BBC micro, il Commodore 64, il Dragon 32 e l'Oric-1 sono più versatili in questo campo.

L'alta risoluzione video viene sempre definita a partire dall'angolo in basso a sinistra dello schermo, indipendentemente dal modo selezionato. I valori verticali sono compresi tra 0 e 1023, quelli orizzontali tra 0 e 1279. Questo metodo coerente di definizione video si dimostra particolarmente comodo quando si passa da un modo ad un altro. Se il cambiamento di modo avviene durante l'esecuzione di un programma, allora le immagini sullo schermo verranno automaticamente cancellate.
I colori dello sfondo, dei testi e della grafica vengono selezionati usando i comandi COLOUR e GCOL. Il Micro BBC segue una procedura logica ed interessante che mette l'utente in grado di scegliere un limitato gruppo di colori tra i 16 disponibili. Per meglio comprendere il suo funzionamento, faremo un esempio utilizzando il modo 0 che permette la scelta di due soli colori. I livelli logici di colore 0 ed 1 definiscono i due colori che sarà possibile avere in primo piano: il computer normalmente interpreta lo zero come nero e l'uno come bianco. Il comando COLOUR permette di scegliere il colore del testo in primo piano. COLOUR 1 permetterà di definire il livello logico di colore 1 come colore del testo; è altresì possibile annullare il colore scelto per il testo usando uno dei comandi VDU. VDU19 definisce il livello logico di colore. Per attribuire il numero 1 al colore verde (solitamente definito dal numero 2), bisogna procedere in questo modo:

```
VDU19, 1, 2, 0, 0, 0,
```

I tre 0 finali non hanno, per adesso, alcun significato pratico, ma verranno utilizzati per successive espansioni del sistema.
Al comando GCOL sono associati due numeri. Il primo serve a stabilire se il colore, selezionato dal secondo numero, verrà usato in primo piano o per il fondo.
Prossimamente, torneremo a parlare del Micro della BBC e ne illustreremo le possibilità di alta risoluzione e di definizione dei caratteri; analizzeremo inoltre in modo più approfondito il gruppo dei comandi VDU.

Scambi e ricerche

Vediamo alcuni aspetti del trattamento dei dati nel nostro database elementare

Abbiamo terminato la scorsa lezione con un esercizio: la stesura di un piccolo programma per la gestione di dati. Vediamo quali passi siano necessari per l'inserimento di un record nell'archivio. Proseguiremo poi con lo sviluppo della subroutine INIZIALIZZAZIONE del programma principale. Supponiamo che, nel record, vi siano i seguenti campi, e che ciascuno di essi corrisponda ad una matrice:

CAMPO	MATRICE
1 NOME	CAMNOM$
2 NOME MODIFICATO	CAMMOD$
3 VIA	CAMVIA$
4 LOCALITÀ	CAMLOC$
5 NAZIONE	CAMNAZ$
6 NUMERO TELEFONICO	CAMTEL$
7 INDICE	CAMIND$

Il significato dei vari campi dovrebbe essere abbastanza chiaro, con l'eccezione dei campi 2 e 7. Il campo NOME MODIFICATO è necessario in quanto abbiamo deciso di offrire una certa libertà nel formato del nome, sia in fase d'immissione che in quella di ricerca.
Durante i primi stadi dello sviluppo, abbiamo ritenuto opportuno convertire tutti i nomi immessi secondo un formato standard: tutte le lettere convertite in maiuscole, tutti i caratteri non alfabetici (spazi, punti, ecc.) rimossi e, per separare il nome dal cognome, un unico spazio.
La necessità di seguire un formato comune per tutti i nominativi ci consente di semplificare le routine di riordino e di ricerca. D'altra parte, però, quando viene chiesta la ricerca di un particolare nominativo, vorremmo che esso fosse visualizzato tale e quale lo abbiamo immesso. Questo problema ha due soluzioni: effettuare la conversione in formato standard soltanto durante le fasi di riordino e di ricerca, oppure convertire i nominativi al momento dell'immissione e conservarli in un apposito campo del record, in modo da essere immediatamente accessibili durante la ricerca o il riordino.
Ambedue le soluzioni hanno qualche inconveniente: infatti, una conversione "ad hoc" può far risparmiare una notevole quantità di memoria, ma richiede tempi notevolmente lunghi d'esecuzione. Il secondo metodo, invece, pur non essendo economico in termini di memoria, è decisamente più veloce.
Il secondo campo che potrebbe generare qualche perplessità è INDICE. Lo abbiamo incluso nel record soltanto come campo di riserva: se in futuro volessimo apportare modifiche o estensioni, la sua presenza ci eviterà di dover riscrivere considerevoli porzioni di programma. L'uso di un campo INDICE introduce il concetto di "legame" tra i dati di un database: le informazioni contenute in un record possono avere particolari connessioni logiche con quelle contenute in altri record. Poter disporre, quindi, di un campo da utilizzare per "collegare" i vari record, è particolarmente utile, specialmente nell'aggiunta di nuovi nominativi.
Prima di affrontare il problema dell'immissione di nuovi record, è bene soffermarsi sulla struttura dei file da impiegare. In primo luogo limiteremo a 50 il numero massimo di nominativi (vedremo poi come aumentare questo limite) ed, inoltre, supponiamo che, nella subroutine INIZIALIZZAZIONE, i dati siano già stati opportunamente trasferiti in apposite matrici.
Quando si immette un nuovo record, la soluzione più semplice consiste nell'accodarlo all'ultimo record del file (ossia il primo elemento libero nella matrice utilizzata dal programma). Molto probabilmente, però, il nuovo gruppo d'informazioni non risulterà in una posizione ordinata rispetto agli altri, ma di questo ci occuperemo in seguito. In più punti del programma occorre conoscere la dimensione delle matrici usate: questa informazione potrebbe essere ricavata nella subroutine INIZIALIZZAZIONE. È il classico caso nel quale viene impiegata una variabile cosiddetta "globale" (ossia, utilizzata in più sezioni del programma). Noi la chiameremo DIMENSIONE. Un'altra variabile globale che può tornarci utile è l'indice del record "corrente". Quando viene inizialmente lanciato il programma, non esiste ancora alcun record "corrente", quindi l'indice, che chiameremo CORR, verrà stabilito soltanto quando accederemo al primo dato ed assegnargli un valore iniziale non ha alcun senso. Comunque, durante INIZIALIZZAZIONE, possiamo assegnare il valore 0 a CORR.

Quando si lancia il programma, viene eseguita una serie di inizializzazioni ed i dati saranno letti (da nastro o da disco) per essere poi trasferiti nelle matrici alfanumeriche. Fatto ciò, l'esecuzione passa alla sezione SELEZIONE, che presenta un menu di opzioni. Se si sceglie l'opzione 6, questo valore viene assegnato alla variabile SCELTA, e ciò provocherà l'esecuzione del sottoprogramma AGGREC. Questo sottoprogramma, supponendo noto il valore di DIMENSIONE, può immediatamente richiedere l'immissione dei dati di un record, mediante delle INPUT. Ovviamente, le matrici saranno già state dimensionate opportunamente nella sezione INIZIALIZZAZIONE.
Quasi certamente, l'aggiunta di un nuovo record altera l'ordine logico degli elementi nella matrice. Poiché il riordino degli elementi richiede un certo tempo, non è conveniente eseguirlo immediatamente dopo ogni nuovo inserimento. Per il momento, basta prender nota dell'avvenuto inserimento modificando il valore di una variabile "di comodo" (o "flag").
Siamo adesso in grado di formulare un primo elenco delle matrici, delle variabili e delle flag da utilizzare nel programma.

MATRICI

CAMNOM$	(Campo del nominativo)
CAMMOD$	(Campo del nominativo modificato)
CAMVIA$	(Campo dell'indirizzo)
CAMLOC$	(Campo della città)
CAMNAZ$	(Campo della nazione)
CAMTEL$	(Campo del numero telefonico)
CAMIND$	(Campo dell'indice)

VARIABILI

DIMENSIONE	(dimensione "attuale" del file)
CORR	(indice del record "corrente")

FLAG

RAGG	(nuovo record aggiunto)
SORT	(sort eseguito dall'ultima modifica)
SALV	(memorizzazione avvenuta dall'ultima modifica)
RMOD	(apportata modifica dall'ultima memorizzazione)

Può darsi che, nel successivo sviluppo del programma, si senta la necessità di definire qualche altra matrice e certamente saranno necessarie altre variabili. Per quanto concerne le flag, ne potranno occorrere altre, ma le quattro indicate potrebbero non servirci tutte. Infatti, non c'è bisogno né di memorizzare né di riordinare le matrici, salvo che non vi sia stata apportata una qualche modifica. Pertanto, la sola flag necessaria, in realtà, è RMOD. Comunque, se decidessimo di impiegare tutte e quattro le flag, ad esse andranno assegnati degli opportuni valori in INIZIALIZZAZIONE.

Vediamo adesso come sviluppare la sezione AGGREC:

I. (ESECUZIONE) 6 (AGGREC)

```
INIZIO
   Trovare la dimensione del file
   Sollecitare l'immissione dei dati
   Collocare i dati "in coda" alle matrici
   Variare il valore di RMOD
FINE
```

II. (ESECUZIONE) 6 (AGGREC)

```
INIZIO
   (la dimensione del file è DIMENSIONE)
   (Sollecito immissione dati)
   Pulizia dello schermo
   Visualizzazione del messaggio relativo all'immissione del
   primo dato di matrice (DIMENSIONE)
   Input del dato matrice (DIMENSIONE)
   (ripetere l'operazione per gli altri dati)
   Assegna il valore 1 a RMOD
FINE
```

La codifica non comporta grossi problemi: non implica l'uso di cicli né di altre complicate strutture, perciò possiamo passare direttamente alla trasposizione in BASIC. L'unica osservazione da fare è che il valore della variabile DIMENSIONE è già stabilito esternamente a questo modulo.

III. (ESECUZIONE) 6 (AGGREC) CODICE BASIC

```
CLS : REM OPPURE PRINT CHR$(12) PER PULIRE LO SCHERMO
INPUT "NOMINATIVO ";CAMNOM$(DIMENSIONE)
INPUT "INDIRIZZO ";CAMVIA$(DIMENSIONE)
INPUT "CITTÀ ";CAMLOC$(DIMENSIONE)
INPUT "NAZIONE ";CAMNAZ$(DIMENSIONE)
INPUT "TELEFONO ";CAMTEL$(DIMENSIONE)
LET RMOD = 1
LET CAMIND$ = STR$(DIMENSIONE)
GOSUB ★★ MODNOME ★★
RETURN
```

La terzultima riga assegna al campo CAMIND$ il valore di DIMENSIONE (convertito in stringa alfanumerica dall'istruzione STR$), in modo da poterlo usare come indice in altre parti del programma. La subroutine MODNOME, chiamata nella penultima riga, non è altro che la sezione di programma descritta a pagina 254, leggermente modificata. Il suo compito è di convertire il nome immesso in un formato standard e di collocarlo nell'elemento di CAMMOD$(DIMENSIONE). Tutti i riordini o le ricerche ora possono esser fatti sul campo CAMMOD$. Risalire ai corrispondenti valori delle altre matrici è semplice, poiché queste hanno l'indice in comune con quello di CAMMOD$.
I due sottoprogrammi MODREC e CANREC (rispettivamente per modificare un record e per cancellarlo) sono molto simili ad AGGREC, salvo il fatto che il particolare record da modificare o cancellare deve essere prima individuato.

Di conseguenza, ambedue queste sezioni inizieranno con una chiamata alla subroutine TROVAREC, la cui formulazione si richiama a quella di ricerca, descritta a pagina 273. Nel nostro caso, sarà molto improbabile che esistano due dati identici nel file, quindi tutto risulta più facile.
Esistono due criteri per effettuare una ricerca. Il primo prevede la scansione del file, iniziando dal primo elemento, fino a giungere al dato ricercato. Questo metodo può essere molto lento (il dato cercato potrebbe essere l'ultimo del file!), anche se offre il vantaggio di evitare il riordino dei dati ogni volta che viene aggiunto, modificato o cancellato un record.
Qualora, invece, i dati siano ordinati (numericamente o alfabeticamente, per esempio) si può applicare il metodo della ricerca binaria. In questo caso, addirittura, più sono i dati contenuti nel file, maggiore è l'efficienza del metodo. Inoltre, è possibile velocizzare la ricerca individuando, preliminarmente, la posizione della prima e dell'ultima comparsa, nel file, dell'iniziale del nominativo cercato. Un altro sistema per rendere la ricerca più rapida consiste nel mantenere una tabella in memoria contenente la posizione di ciascuna lettera iniziale dei nominativi. La tabella, però, deve essere costantemente aggiornata. Sulla ricerca ed il riordino dei dati è stata scritta una quantità immensa di libri e, almeno per adesso, non sembra esistere un'unica soluzione ideale per tutte le circostanze. Quindi, per la nostra rubrica computerizzata, ci accontenteremo di adottare dei sistemi semplici e pratici, considerando anche che il programma deve poter funzionare sia con unità a dischi, che a nastri.
Vediamo, qui di seguito, la parte di programma, in pseudolinguaggio, relativa alla ricerca nella matrice CAMMOD$. La chiave di ricerca (ossia il valore cercato) è contenuta nella variabile alfanumerica KEY$.

```
Chiedi il nome da cercare
LET KEY$ = nome da cercare
LET FDO = 1
LET RICERCA = 0
LET ALTO = DIMENSIONE
CICLO finché (FDO < = ALTO) AND (RICERCA = 0)
  LET MET = INT ((FDO + ALTO)/2)
  SE KEY$ = CAMMOD$(MET)
    ALLORA
      PRINT CAMNOM$(MET)
      PRINT CAMVIA$(MET)
      PRINT CAMLOC$(MET)
      PRINT CAMNAZ$(MET)
      PRINT CAMTEL$(MET)
      LET RICERCA = 1
    ALTRIMENTI
      SE KEY$ > CAMMOD$(MET)
      ALLORA LET FDO = MET + 1
      ALTRIMENTI LET ALTO = MET — 1
      FINE SE
  FINE SE
FINE CICLO
SE RICERCA = 0 ALLORA PRINT "Record non trovato"
FINE
```

La struttura del modulo è sostanzialmente identica a quella illustrata a pagina 274, ma è stata aggiunta la visualizzazione di un messaggio qualora non venga trovato il record cercato.
Sfortunatamente, l'impostazione di questo modulo, benché valida in linea di principio, in pratica si rivelerebbe poco affidabile. Il punto debole sta proprio nel nucleo centrale della subroutine (il confronto SE KEY$ = CAMMOD$). Infatti, nonostante abbiamo previsto di standardizzare il formato dei nominativi, possono ancora presentarsi delle discrepanze tra il valore contenuto in KEY$ e quello di CAMMOD$. Una ricerca, dunque, può dare esito negativo, sebbene il nominativo cercato faccia parte del database. Scorrendo velocemente con lo sguardo una normale rubrica, possiamo facilmente sorvolare tutta una serie di piccoli particolari insignificanti e concentrarci su quelli essenziali alla ricerca. Un computer è costretto a comportarsi differentemente.
Il primo miglioramento che possiamo apportare al programma è di limitare una prima ricerca al solo cognome. Torna utile, in questo caso, prevedere un formato standard COGNOME (spazio) NOME, per gli elementi di CAMMOD$. Avendo precedentemente sviluppato una breve subroutine per invertire di posizione il nome ed il cognome (vedere "Basic & Basic"), potremmo usarla all'interno di AGGREC.
Localizzata la prima ricorrenza, nel database, del cognome cercato, la subroutine TROVAREC dovrebbe operare un confronto sul nome, per vedere se corrisponde a quello richiesto. Se corrisponde, il problema è risolto, altrimenti le cose si complicano ed è necessaria un'attenta analisi per stabilire il procedimento migliore.
Un sistema potrebbe essere quello di scorrere tutti i cognomi, e, qualora non venga trovata alcuna corrispondenza "esatta", provare con una ricerca "approssimativa". Il problema, adesso, è di definire cosa sia una ricerca "approssimativa"!
Facciamo qualche esempio. Se nella ricerca usassimo "Pippo", al posto di "Filippo", potremmo considerare questa un'approssimazione? Vediamo di definire più accuratamente il problema e sviluppare, poi, un adeguato modulo di programma.
Se la stringa in memoria è ROBERTO, quale delle due seguenti forme è la più "approssimativa", ROB o RBRT?
La seconda azzecca quattro lettere su sette, mentre la prima solo tre, ma nella giusta sequenza.
Per la definizione, dunque, esiste la più ampia libertà di scelta. Nel nostro programma daremo la priorità ad una corrispondenza esatta tra una porzione del nome usato come

chiave di ricerca e quello presente in memoria. Se la ricerca fallisce, proveremo ad individuare tutti quei nomi col massimo numero di lettere in comune con la chiave. Ecco il problema in termini di input/output:

INPUT
Una stringa di caratteri

OUTPUT
La stringa, tra quelle in memoria, più simile a quella impiegata per la ricerca

Qui di seguito, riportiamo lo sviluppo, in pseudolinguaggio, di un modulo per tentare l'individuazione in base alle prime "n" lettere del nome, dove "n" è la lunghezza della chiave di ricerca (KEY$). Viene visualizzato un messaggio qualora la ricerca abbia esito negativo.

```
DIM MATRICE$(4)
FOR L = 1 TO 4
READ MATRICE$(L)
NEXT L
DATA "ROBERTO", "RICCARDO", "ROSSANA", "ROBERTA"
LET KEY$ = "ROB"
LET LKEY = LEN(KEY$)
LET RICERCA = 0
CICLO FOR INDICE = 1 TO 4
  IF KEY$ = LEFT$(MATRICE$(INDICE),LKEY)
    THEN PRINT "HO TROVATO :";MATRICE$(INDICE)
      LET RICERCA = INDICE
  FINE IF
FINECICLO
IF RICERCA = 0
  THEN PRINT KEY$;" NON TROVA CORRISPONDENZA NEI
       PRIMI ";LKEY;" CARATTERI"
```

Si potrebbe, a questo punto, ripetere la ricerca, tentando un confronto tra la chiave ed altrettanti caratteri nei nominativi, ma partendo dalla seconda lettera, poi dalla terza, e così via. Infine, si potrebbe ancora tentare con i nominativi che hanno il maggior numero di caratteri in comune con la chiave. Ma lasciamo queste alternative come eventuali esercitazioni, in quanto appesantirebbero notevolmente il funzionamento della nostra rubrica.
Nei programmi applicativi commerciali vengono spesso offerte numerose forme di ricerca "approssimata", dalle più semplici (come quella che abbiamo presentato), a sofisticate approssimazioni basate su complicati algoritmi. Una forma molto utile di ricerca offre la possibilità di usare dei caratteri "jolly": specificando una chiave del tipo R?N, il programma trova RENATA, RENZO, RINALDO, ecc. (Il punto interrogativo rappresenta il carattere della chiave non specificato, e su di esso non viene eseguito alcun confronto). Altre forme prevedono ricerche fonetiche, particolarmente utili con nomi stranieri, quando non si è certi di come sono scritti: chiedendo SMITH, il programma trova SMYTHE, SCHMITT e così via.

Basic & Basic

SPECTRUM

Programma per lo scambio di posizione tra nome e cognome, già riportato a pagina 136, nella versione Spectrum:

```
100 CLS
200 PRINT "IMMETTERE IL NOMINATIVO
    NELLA FORMA"
300 PRINT "NOME COGNOME"
400 PRINT "AD ES: CARLO VERDI"
500 INPUT "NOMINATIVO";N$
600 GOSUB 9500
700 PRINT "IL NOME IN FORMATO STAN-
    DARD È"
800 PRINT N$
1000 STOP
9500 REM INVERSIONE NOME/COGNOME
9520 GOSUB 9600
9540 IF P = 0 THEN RETURN
9560 LET N$ = S$ + ", "+F$
9580 RETURN
9600 REM SEPARAZIONE DI N$ IN NOME E
     COGNOME
9620 LET N = LEN(N$)
9630 LET P = 0
9640 FOR K = 1 TO N
9650 IF FN M$(N$,K,1) = " " THEN LET P = K
     : LET K = N
9660 NEXT K
9670 IF P = 0 THEN RETURN
9680 LET F$ = FN L$(N$,P—1)
9700 LET S$ = FN R$(N$,N—P)
9720 RETURN
9900 REM FUNZIONI SPECIALI DEFINITE DA
     UTENTE
9990 DEF FN M$(X$,P,N) = X$(P TO P+N—1)
9991 DEF FN L$(X$,N) = X$( TO N)
9992 DEF FN R$(X$,N) = X$(LEN X$—N+1 TO)
```

LEFT$

Sul Commodore 64, sul Vic-20, sull'Oric-1 e sul Lynx, sostituire le linee dalla 9600 alla 9720 del listato precedente con:

```
9600 REM SEPARAZIONE DI N$ IN NOME E
     COGNOME
9620 LET N = LEN(N$)
9630 LET P = 0
9640 FOR K = 1 TO N
9650 IF MID$(N$,K,1) = " " THEN LET P = K:
     LET K = N
9660 NEXT K
9670 IF P = 0 THEN RETURN
9680 LET F$ = LEFT$(N$,P—1)
9700 LET S$ = RIGHT$(N$,N—P)
9720 RETURN
```

RIGHT$

MID$

si cancellino le linee da 9900 a 9992.

INSTR

Sul Dragon 32 e sul Micro BBC, sostituire le linee da 9600 a 9720 del listato per lo Spectrum, con:

```
9600 REM SEPARAZIONE DI N$ IN NOME E
     COGNOME
9620 LET N = LEN(N$)
9640 LET P = INSTR(N$," ")
9670 IF P = 0 THEN RETURN
9680 LET F$ = LEFT$(N$,P—1)
9700 LET S$ = RIGHT$(N$,N—P)
9720 RETURN
```

si cancellino le linee da 9900 a 9992.

La funzione INSTR, come già detto precedentemente, è utile in applicazioni simili a queste. Con essa si potrebbe facilmente implementare una ricerca "approssimata".

INPUT

Per il Micro BBC, sostituire la linea 500 con:

```
500 INPUT "NOMINATIVO",N$
```

Konrad Zuse

Le macchine concepite da Zuse, come gli altri primi computer, vennero usate per scopi militari



per concessione della SIEMENS-MUSEUM, MONACO

Nel 1940, mentre in America veniva costruito il primo computer a valvole (ENIAC), Konrad Zuse cominciò a lavorare su un calcolatore programmabile, probabilmente il primo computer del mondo.
Zuse nacque a Berlino il 22 giugno 1910. Dopo aver studiato al Politecnico della sua città, lavorò come ingegnere aeronautico per la compagnia aerea Henschel, progettando nuovi tipi di ali. I principi matematici di base impiegati negli studi sul consolidamento delle strutture alari di un aereo, al fine di renderle più resistenti alle tensioni del volo ad alta velocità, furono formulati nel 1920. I calcoli necessari per la progettazione di un paio d'ali richiedevano il lavoro di intere squadre di tecnici, muniti di macchine addizionatrici meccaniche e di regoli calcolatori. Zuse ben presto si rese conto che era necessaria una macchina che rendesse più rapidi i tempi di progettazione. Lavorando di sera con alcuni amici, nell'appartamento dei suoi genitori, iniziò la costruzione di un computer in grado di svolgere questo tipo di lavoro.
La prima macchina fu la Z1: un apparecchio meccanico capace di eseguire le 4 operazioni aritmetiche elementari, calcolare le radici quadrate e convertire i numeri decimali in notazione binaria e viceversa. Pur ignorando i risultati ottenuti da Charles Babbage (vedere a pag. 220), la cui Macchina Differenziale era stata progettata per sviluppare i complessi calcoli necessari per le carte nautiche, Zuse arrivò a conclusioni assai simili ed in alcuni casi più avanzate. La sua più grande intuizione fu il riconoscere che una leva era equiparabile ad un interruttore che poteva essere spostato in due differenti posizioni, "on" ed "off", e che, quindi, poteva essere usata come mezzo di memorizzazione dati o strumento di controllo per un apparecchio.
Zuse insistette nella sua idea di rappresentare tutti i dati e le istruzioni in forma binaria e nel 1941 decise di costruire un computer elettromagnetico che fu chiamato Z2. Inizialmente il governo tedesco, totalmente assorbito dallo sforzo bellico, mostrò scarso interesse per la sua invenzione. Comunque le potenzialità militari di questa invenzione furono alla fine riconosciute e vennero stanziati dei fondi per lo sviluppo del nuovo computer Z3.
Zuse costruì lo Z3 superando enormi difficoltà: i continui bombardamenti lo costrinsero a spostare più volte i suoi laboratori; venne inviato due volte in prima linea e per due volte fu richiamato dal fronte orientale per proseguire il suo lavoro di ricerca. La mancanza di materiali, causata dalla guerra, lo costrinse ad improvvisare e ad utilizzare per il suo lavoro componenti di apparecchi telefonici e vecchie pellicole cinematografiche, punzonate con codici ad 8 fori per fotogramma, al posto dei nastri di carta.
Lo Z3 poteva memorizzare 64 parole, ciascuna di 22 bit di lunghezza. Le informazioni venivano immesse mediante tastiera ed i risultati erano visualizzati da un sistema di lampade montato su di un piano. Purtroppo lo Z3 fu distrutto assieme ai primi computer di Zuse nel bombardamento a tappeto di Berlino nel 1945.
Uno dei computer fu utilizzato dalla compagnia aerea Henschel per la progettazione della "bomba volante" HS-293. Si trattava di un aereo senza pilota, lanciato da un bombardiere e guidato sul bersaglio da un segnale radio.
L'ultimo computer progettato da Zuse durante la guerra, lo Z4, usava parole di 32 bit. Fu trasportato a Gottinga mentre le Forze Alleate si avvicinavano a Berlino. Concluse il suo viaggio a Basilea, in Svizzera, dove continuò a funzionare sino al 1954: fu uno dei più importanti computer funzionanti in Europa in quegli anni.
Zuse non poté costruire altri computer nella Germania post-bellica, per cui concentrò il proprio lavoro sulla teoria. Sviluppò un linguaggio sofisticato, chiamato Plankalkül, che poteva operare logicamente sia con la matematica che con informazioni di carattere più generale. Quando egli poté di nuovo dedicarsi alla costruzione di computer, fondò l'industria Zuse, che fu la maggior casa produttrice di computer della Germania fino al 1969, anno in cui fu assorbita dalla Siemens Corporation. Il professor Zuse è tuttora attivo nell'industria dei computer.

Le micidiali V1 e V2

I computer di Zuse furono prodotti per sostituire interi staff di tecnici che dovevano usare il regolo calcolatore per i loro calcoli aeronautici. In particolare, essi furono applicati al progetto delle bombe volanti V1 e V2 che furono massicciamente usate nella II Guerra Mondiale



© THE IMPERIAL WAR MUSEUM

Viaggiare grazie ai computer

Per il trasporto di passeggeri e di merci, al giorno d'oggi, il computer è indispensabile



per concessione della YRM ARCHITECTS & PLANNERS

Il "People Mover"
Come già hanno fatto molti aeroporti negli Stati Uniti, l'aeroporto di Gatwick, a sud di Londra, ha installato un rivoluzionario sistema di trasporto tra terminali che combina in sé la stabilità di rotta e la possibilità di funzionare senza guidatore con la comodità e la convenienza di un autobus o di una corriera. Progettato e costruito dalla Westinghouse Corporation, famosa per i suoi sistemi di segnalazione ferroviaria, il People Mover può trasportare fino a 100 passeggeri

Un secolo e mezzo fa si impiegavano tre mesi per andare dall'Europa in Australia; nel 1984 questo stesso viaggio può essere compiuto in meno di mezza giornata. Questo miracolo tecnologico non sarebbe stato possibile, tuttavia, senza metodi estremamente sofisticati di controllo computerizzato del trasporto.

Il viaggio aereo è la forma di trasporto che pone i problemi più complessi. L'aeroporto di Heathrow a Londra, ad esempio, controlla più di mille aerei al giorno, con un movimento di 120 aerei all'ora nei periodi di punta. Senza l'assistenza di un computer per controllare questa attività, il sistema da solo non potrebbe funzionare.

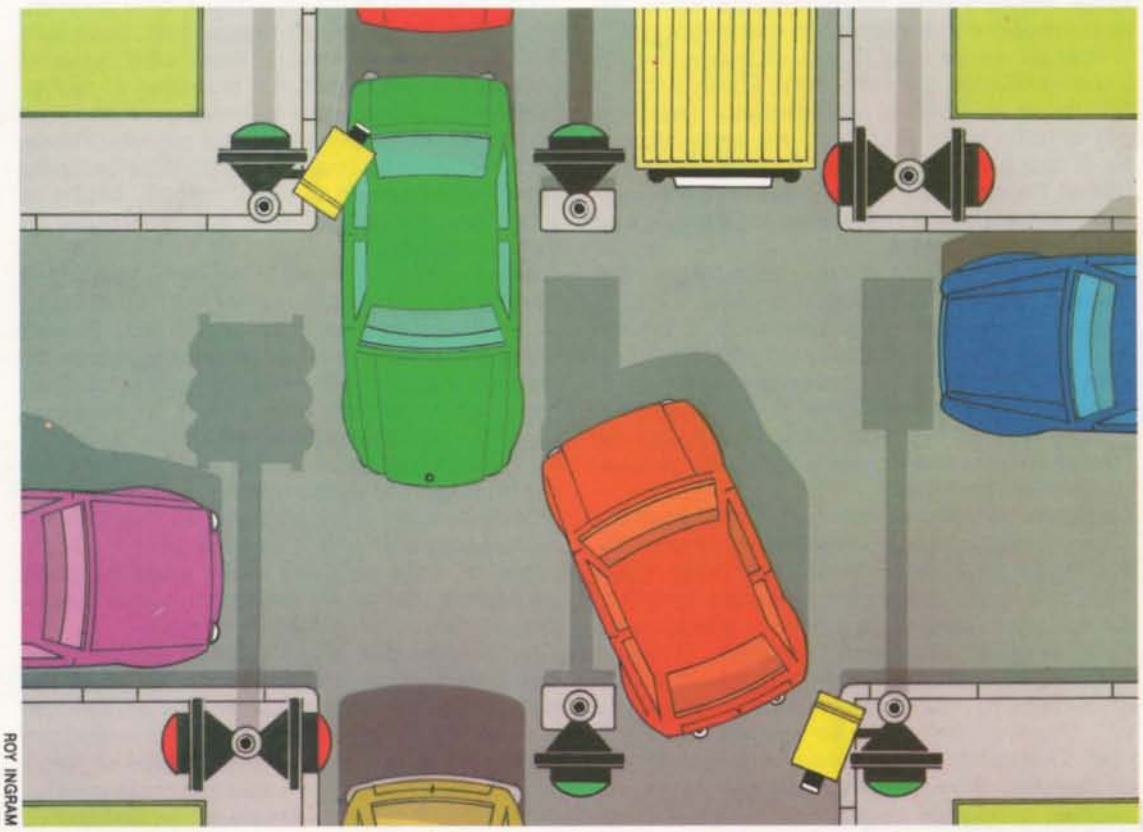
Prendiamo il caso di un viaggiatore che desideri andare da Londra a New York. A partire da quando viene acquistato il biglietto e riservato il posto sull'aereo (molte agenzie di viaggio usano il Prestel per accedere al computer delle prenotazioni della compagnia aerea), fino al momento dell'atterraggio all'aeroporto Kennedy, sono coinvolti almeno 15 diversi computer.

La prima considerazione riguarda il modello d'aereo impiegato: i moderni aerei per trasporto di passeggeri hanno un costo veramente notevole e, per massimizzare i profitti, gli operatori devono mantenere questi aerei nelle migliori condizioni possibili. Il modo migliore per ottenere questo risultato è l'applicazione di una "manutenzione pianificata". Dopo un determinato numero di ore di volo, l'aereo ritorna alla base di manutenzione tecnica, dove il personale può prendere visione, mediante registrazioni computerizzate, della storia completa dell'aereo dal primo giorno della sua costruzione e del numero di serie di ciascun suo componente, sia passata che presente. Le registrazioni riguardano, in particolare: tutti gli interventi tecnici operati sull'aereo fino a quel momento; i rapporti sulle sue prestazioni, compilati dal personale di volo e dagli altri membri dell'equipaggio; i dati sul consumo di carburante e qualsiasi altra informazione che possa rivestire interesse. Nel suo piccolo, l'utente di un piccolo computer potrebbe considerare l'applicazione di simili metodi, anche se non così dettagliati, alla manutenzione di un'automobile.

L'aereo torna in servizio soltanto quando il programma di manutenzione è stato completato ed aggiornato. A questo punto, l'aereo entra a far parte di un altro sistema computerizzato (il sistema di controllo operativo della compagnia aerea), che svolge le seguenti funzioni: assegna all'aereo la rotta da seguire, definisce gli scali di rifornimento da effettuare lungo la rotta, sceglie l'equipaggio, i cibi da servire, il programma degli spettacoli cinematografici che verranno proiettati durante il volo ed ogni altra combinazione necessaria per trasportare tre/quattrocento persone attraverso mezzo mondo.

Un altro sistema di controllo computerizzato opera all'interno dell'aeroporto, dove gli impiegati devono poter far fronte all'enorme massa di richieste formulate dalle compagnie aeree, per le quali un ritardo, anche di pochi minuti, può comportare una notevole perdita di denaro. È soltanto grazie ad un sistema di

pianificazione computerizzato, che tutte queste operazioni vengono realizzate con successo. Il sistema computerizzato dell'aeroporto si occupa anche di chiamare i passeggeri alle operazioni di "check in" per i voli in partenza ed, inoltre, tiene costantemente aggiornati i tabelloni degli arrivi e delle partenze.
Ancor prima che i passeggeri siano arrivati all'aeroporto, il pilota avrà steso un dettagliato piano di volo, concordandolo con il Controllo del Traffico Aereo. Sorprendentemente, il lavoro che il CTA deve svolgere è solo parzialmente computerizzato: grazie ad un sistema di radiofari ad impulsi, i controllori di volo non devono più affidarsi alle comunicazioni verbali del pilota per conoscerne la posizione. Essi possono vedere sui loro schermi radar ciascun "blip" (segnale puntiforme di ritorno), identificato da un numero di volo, assieme ad una lettura dell'altitudine, decifrata mediante un computer ed un codice di destinazione trasmesso automaticamente dall'aereo.
Il controllore di volo dispone anche di un ulteriore livello di assistenza computerizzata, che si presenta sotto forma di strisce di carta stampate, ciascuna delle quali si riferisce ad un settore della rotta programmata, estratto dal piano di volo del pilota. Queste strisce di carta, che forniscono informazioni riguardanti la traiettoria del volo, l'altitudine, il carico pagante ed il tipo di aeroplano, aiutano il controllore a guidare il velivolo, attraverso la sua area di competenza, nel modo più rapido ed economico.
Ma i computer vengono largamente usati anche nel trasporto ferroviario: il lavoro svolto dal controllore dei segnali di una ferrovia, per quanto sia meno complesso di quello del controllore di volo, presenta tuttavia alcuni elementi comuni. In questo caso si tratta di guidare con sicurezza, e col minimo costo, convogli merci e traffico passeggeri attraverso una particolare area di competenza. Fin dalla metà degli anni '70 le ferrovie britanniche utilizzano un sistema di controllo computerizzato, seguendo in questo modo il pionieristico lavoro svolto negli Stati Uniti dalla Southern Pacific Railroad.
Il suo Sistema Totale di Elaborazione Operazionale cura tutti gli aspetti relativi al controllo dei convogli ferroviari: tiene costantemente aggiornato un accurato inventario di ciascuna locomotiva e di ogni pezzo del materiale rotabile, controlla la formazione e la composizione dei convogli e ne definisce i percorsi.
Ciascun vagone ha un proprio numero di matricola che viene registrato dal computer di sistema assieme al nome della località in cui il vagone si trova. Quando occorre trasferire un vagone da una località all'altra, esso viene agganciato ad un particolare treno scelto dal computer e la sua destinazione viene memorizzata. Terminato il viaggio, il suo nuovo "status" viene registrato sul computer, che può quindi assegnarlo nuovamente ad un altro convoglio. Considerando il volume di traffico del sistema ferroviario britannico (alla fine del 1970 c'erano 185000 vagoni in servizio, vale a dire circa 2000 treni al giorno), un simile sistema di controllo computerizzato è assolutamente indispensabile.
Molti problemi relativi alle operazioni aeroportuali si presentano anche nelle stazioni



ROY INGRAM

Occhio al verde

La sincronizzazione dei semafori stradali è molto utile nel tentativo di regolare il flusso di autoveicoli, soprattutto sulle strade principali delle grandi città. Oggi i semafori, per mezzo di rivelatori radar ad effetto Doppler, possono rilevare la densità di traffico nelle loro immediate vicinanze e, quindi, comunicare i dati ad un sistema computerizzato. L'alternanza delle luci viene così regolata adattandosi alle mutevoli condizioni di traffico

ferroviarie e molto spesso sono state adottate soluzioni simili. Tuttavia, le ferrovie hanno un problema in più: il crescente impiego di stazioni prive di personale. Sono stati condotti numerosi esperimenti nei quali il compito di rispondere alle richieste dei viaggiatori ed, in qualche caso, di annunciare, mediante sintetizzatori vocali, gli arrivi e le partenze dei treni, è stato affidato a dei microcomputer. La computerizzazione ha inoltre reso possibile l'adozione, in via sperimentale, di treni senza personale. La Victoria Line della metropolitana di Londra, ad esempio, è dotata di queste possibilità che però, in pratica, non vengono utilizzate perché i treni computerizzati, senza conduttore, sono ancora visti con sospetto e diffidenza dal pubblico.

Una realizzazione automatizzata molto simile si ha nel modellismo: nei plastici più sofisticati ogni locomotiva ha un proprio codice d'identificazione, immagazzinato in un piccolo microcomputer dotato di un singolo chip. La stazione di controllo invia dei segnali a ciascun treno, modulando opportunamente la corrente immessa sulle rotaie, in modo tale che soltanto una particolare locomotiva è in grado di demodulare il segnale. In questo modo un gran numero di treni può viaggiare contemporaneamente sulla stessa linea sotto il diretto controllo del microcomputer centrale.

La realizzazione di convogli privi di macchinista è possibile in quanto essi viaggiano su rotaie, mentre lo stesso principio si è dimostrato difficilmente applicabile ai veicoli che viaggiano su strada. Tuttavia, i computer vengono già usati nel trasporto stradale, soprattutto nei trasporti pubblici e nelle operazioni di trasporto merci. In questi casi, essi aiutano a pianificare i percorsi dei veicoli ed a compilare tabelle orarie, lavoro notevolmente complesso in una città delle dimensioni, ad esempio, di Londra dove è necessario integrare i servizi di metropolitana, autobus e treni in un unico sistema di trasporto pubblico.

Il problema consiste nel riuscire a mantenere operativo un numero di veicoli sufficiente ad evitare ritardi inaccettabili per il pubblico e, nello stesso tempo, il più possibile limitato al fine di rendere remunerativo questo complesso sistema. Essendo questo un problema squisitamente statistico, i computer sono adattissimi a risolverlo. Un altro problema che può essere affrontato usando metodi statistici è la pianificazione dei percorsi per i furgoni che consegnano merce a domicilio. L'obiettivo è quello di ridurre al minimo tanto il numero di consegne che ciascun furgone deve effettuare, quanto la distanza da coprire tra una consegna e l'altra. Un'interessante variante a questo problema è la dislocazione dei taxi e delle automobili della polizia, che devono spostarsi secondo itinerari pianificati, invece di far periodicamente ritorno alle loro basi operative. La posizione di ciascun veicolo viene registrata sotto il nome di una strada ed il computer la visualizza su di un tabellone di riferimento. Una chiamata per un taxi o per un intervento d'emergenza viene notificata al computer nello stesso modo. È abbastanza semplice, così, rispondere in modo rapido ed adeguato alle richieste in arrivo: basta gestire richieste ed interventi secondo principi organizzativi pianificati in precedenza.

Uno degli esperimenti più interessanti di utilizzazione dei computer nel trasporto pubblico è il sistema "dial-a-bus" (traducibile con "componi il numero di un autobus"), oggi in funzione nei sobborghi di Hannover, in Germania. Questo sistema, basato su una "flotta" di minibus che non seguono itinerari programmati, permette al passeggero di chiamare la stazione di controllo centrale dalla fermata dell'autobus, comunicando la propria destinazione. Il miniterminale della fermata (che assomiglia al terminale di una "cassa continua" di una banca) stampa su di uno scontrino il tempo occorrente all'autobus per arrivare alla fermata (mai più di 5 minuti), la durata della corsa (prendendo in considerazione anche le necessità degli altri passeggeri) e l'importo da pagare.

Il trasporto di merci e passeggeri corrisponde a circa il 20% di tutto il commercio mondiale. L'uso dei computer in questo campo è più diffuso che in altri ed ha certamente contribuito in modo significativo alla sua espansione. La maggior parte degli esempi citati si basa sull'impiego di computer di piccole o medie dimensioni, ma alcuni di essi sono, in linea di principio, applicabili anche su home computer. È in vendita un certo numero di pacchetti software, in grado di gestire (o simulare) problemi di distribuzione, pianificazione, tabulazione d'orari ed altre simili funzioni.



per concessione della BRITISH AIRPORTS AUTHORITY

È in partenza...

Una delle caratteristiche di più immediato rilievo, nel sistema operativo computerizzato di un aeroporto, è il tabellone di arrivi e partenze. Questo apparecchio elettromeccanico viene tenuto costantemente aggiornato dal computer centrale via via che giungono nuove informazioni

Navi fabbrica

L'industria del trasporto navale usa i computer molto meno degli altri mezzi di trasporto, ma un importante campo d'applicazione è quello relativo al trasporto dei "container". I microcomputer giocano un importante ruolo nella gestione di questo servizio: dalla compartecipazione (più spedizionieri marittimi utilizzano in comune un container), all'organizzazione di un terminale per container, e perfino alle operazioni di stivaggio di una nave porta-container, onde assicurare un'equilibrata distribuzione del carico



per concessione della "THE MOTOR SHIP"

I linguaggi del computer

Una sintesi sulle più diffuse alternative al BASIC

La quasi totalità degli home e personal computer viene venduta corredata dal linguaggio BASIC, ma ciò non significa che esista solo questo linguaggio. Anche se il BASIC è considerato un linguaggio particolarmente facile da imparare, ne esistono molti altri, più adatti per particolari applicazioni.
A seconda del computer, la gamma di linguaggi disponibili è più o meno ampia, e viene venduta su supporti di vario genere: una cartuccia da inserire nel computer, oppure una ROM da sostituire a quella del BASIC, o un nastro (o dischetto), il cui contenuto deve essere trasferito nella RAM. In quest'ultimo caso, il computer deve possedere una memoria piuttosto ampia, affinché, dopo aver caricato il nuovo linguaggio, resti spazio RAM sufficiente a contenere i programmi.

PASCAL-INGLESE

INGLESE-PASCAL

Il PASCAL fu sviluppato all'inizio del 1970 come erede del BASIC, ma la sua gamma di strutture dati e di controllo, che derivano dal FORTRAN e dall'ALGOL, spinge il programmatore ad affrontare la programmazione in modo sistematico ed a scrivere in modo ben strutturato anche le procedure più semplici. Questo modo di procedere sviluppa una buona tecnica di programmazione, ma rende difficili le fasi iniziali di studio, soprattutto a causa del rigore logico che il linguaggio impone, ma anche perché questo è un linguaggio che impiega compilatori, piuttosto che interpreti. I programmi in PASCAL sono considerati "eleganti", di facile realizzazione e comprensione. Ecco la versione PASCAL del programma presentato in BASIC:

```
VAR
NOME       : PACKED
             ARRAY (1..30)
             OF CHAR;

ETA,CONTO  : INTEGER;
RISP       : PACKED
             ARRAY (1..3)
             OF CHAR;

RUNNING    : BOOLEAN;

BEGIN
RUNNING:=TRUE
WHILE RUNNING DO
  BEGIN
  WRITE ('Come ti chiami?');
  READLN(NOME);
  WRITE ('e quanti anni hai?');
  READLN(ETA);
  FOR CONTO:=1 TO ETA DO
    WRITE(CONTO:3,'Ciao, ':10,NOME);
  WRITE ('Vuoi ripetere?');
  READLN(RISP);
  IF RISP(1) = 'N'
  THEN RUNNING:=FALSE;
  END;
WRITELN('Arrivederci, ',NOME)
END
```

BASIC-INGLESE

INGLESE-BASIC

Il BASIC, linguaggio molto calunniato, è stato sviluppato dal FORTRAN (che fu uno dei primi linguaggi di programmazione di alto livello, ed è tuttora il più popolare per applicazioni scientifiche ed ingegneristiche) come linguaggio d'introduzione didattica alla programmazione per studenti universitari. Destinato ad un uso autodidattico, il BASIC è più diffuso come interprete che come compilatore (vedere a pag. 184) ed è questa la principale ragione per cui esso è diventato il linguaggio base di quasi tutti gli home computer. Implementare su di un computer un programma interprete risulta economico sia in termini di denaro che di memoria, pur offrendo notevoli facilitazioni per lo sviluppo di programmi.
Il BASIC è oggi un linguaggio molto potente, ma viene osteggiato a causa della gran varietà di dialetti non standard (quasi ogni computer ha una propria versione di BASIC) e per la mancanza di sofisticate strutture di dati e di controllo. Teoricamente, questi limiti, ostacolando l'applicazione di una buona tecnica di programmazione, possono creare negli autodidatti cattive abitudini di programmazione. Il breve programma presentato evidenzia il fascino e le caratteristiche generali, ma anche le limitazioni del BASIC.

```
100 INPUT "Come ti chiami";N$
200 INPUT "e quanti anni hai";A
300 FOR K = 1 TO A
400 PRINT K,"Ciao, ";N$
500 NEXT K
600 INPUT "Vuoi ripetere";Y$
700 IF LEFT$(Y$,1) = "S" THEN GOTO 100
800 PRINT "Arrivederci, ";N$
900 END
```

COMAL-INGLESE

INGLESE-COMAL

Il COMAL è stato ideato per abbinare l'accessibilità del BASIC alle potenti strutture e al disciplinato modo di affrontare i problemi del PASCAL. Il COMAL assomiglia ad entrambi e potrebbe essere il modello dal quale è stato sviluppato il BASIC per il Micro BBC. Il COMAL ha riscosso molto successo nelle scuole di calcolo ed in Svezia (dove è nato), ma sembra improbabile che possa sostituire uno dei suoi progenitori come linguaggio d'introduzione alla programmazione.
Questo è il programma "Ciao!" in COMAL.

```
100  RUNNING:=TRUE
200  WHILE RUNNING DO
300       INPUT "Come ti chiami?";N$
400       INPUT "e quanti anni hai?";A
500       REPEAT
600            FOR K:=1 TO A DO
700                 PRINT K,"Ciao, ";N$
800            NEXT K
900       INPUT "Vuoi ripetere?";Y$
1000      IF Y$="N" THEN RUNNING:=FALSE
1100 ENDWHILE
1200 PRINT "Arrivederci, ";N$
```

LISP-INGLESE

INGLESE-LISP

Il LISP venne sviluppato agli inizi degli anni '60 come linguaggio adatto all'elaborazione di liste e da allora è stato molto usato nel campo dell'Intelligenza Artificiale, che frequentemente implica la ricerca ed il confronto di liste di dati, correlazioni e risposte. Diversamente dal BASIC, nel quale il flusso del programma avviene mediante una sequenza d'istruzioni e procedure, il LISP è un linguaggio "funzionale", nel quale un gruppo di comandi elementari può essere costruito in modo da formare sofisticate funzioni, alle quali il programmatore può assegnare nomi di propria scelta. Per esempio:

(SETQ ELENCO1 '(4 7 2 5 1))

crea una lista chiamata ELENCO1, i cui elementi sono i numeri 4, 7, 2, 5, 1.

(CAR ELENCO1)

individua il primo elemento di ELENCO1 (ossia 4, nell'esempio).

(CDR ELENCO1)

produce una lista contenente gli elementi di ELENCO1, ma rimuovendone il primo: (7, 2, 5, 1).

(SETQ ELENCO1 (CDR ELENCO1)).

crea una nuova lista, ELENCO1, con gli stessi elementi del vecchio ELENCO1, ma eliminando il primo elemento.
Il LISP è adattissimo nella soluzione di problemi "ricorsivi", risolvibili definendo una funzione relativamente semplice e riapplicandola a se stessa un certo numero di volte

LOGO-INGLESE

INGLESE-LOGO

Il LOGO fu ideato da uno psicologo che lavorava sull'Intelligenza Artificiale. Assomiglia al FORTH nella sua interattività e nell'uso di un insieme di funzioni "primitive", che si possono incorporare in quelle definite dal programmatore. Ma il principio fondamentale che lo caratterizza è che, per imparare ad usare il LOGO, occorre insegnare a qualcun altro, vale a dire al computer, come usarlo. Il LOGO è considerato un linguaggio rivoluzionario, che presto insegnerà ai bambini come pensare in un modo completamente nuovo. Il LOGO è spesso usato per programmare un piccolo robot su ruote, chiamato tartaruga (vedere a pag. 30).
Qui c'è una parte di un programma in LOGO che disegna una casa, composta da un quadrato sormontato da un triangolo.

```
TO TRIANGLE LENGTH
   REPEAT 3(FORWARD:LENGTH RIGHT 120)
END
TO SQUARE LENGTH
   REPEAT 4(FORWARD:LENGTH RIGHT 90)
END
TO HOUSE LENGTH
   RIGHT 30
   TRIANGLE :LENGTH
   LEFT 90
   SQUARE :LENGTH
END
```

Adesso, digitando HOUSE 15, si ottiene il disegno di una casa con una misura laterale di 15 unità

FORTH-INGLESE

INGLESE-FORTH

Il FORTH assomiglia al LOGO, essendo un linguaggio "funzionale" interattivo, e si compone di un numero di funzioni predefinite, chiamate "primitive", con le quali è possibile costruire funzioni più complesse. Le operazioni matematiche in FORTH sono eseguite sfruttando il concetto di "stack", consentendo un'occupazione di memoria dinamica e quindi ottimizzata. Una conseguenza di questa particolare impostazione è che non viene usata la comune notazione algebrica: invece di scrivere (12 + 4)/2, per trovare la media tra 4 e 12, in FORTH si deve scrivere 12 4 + 2/. Questa forma è nota come Notazione Polacca Inversa, e viene frequentemente usata nelle piccole calcolatrici scientifiche da tasca.
Il FORTH, dunque, è un tipo di linguaggio molto insolito, che obbliga ad un'ottica totalmente nuova per la soluzione di problemi mediante i meccanismi del computer.
Il FORTH, nella gerarchia dei linguaggi ad alto livello, si trova almeno un gradino al di sotto rispetto agli altri, essendo relativamente vicino al linguaggio Assembly.

In queste due pagine viene offerta una sintesi sui principali linguaggi di programmazione disponibili per i personal ed home computer. Essendo tutti di provenienza anglosassone, un confronto, per così dire, linguistico è possibile soltanto con la lingua inglese.

Il computer astronomo

L'osservazione della volta stellare è facilitata da sistemi automatici di posizionamento



MARCUS WILSON-SMITH

Caccia... stellare!
La localizzazione di un corpo celeste diventa più facile quando si può prevedere accuratamente la sua posizione in un particolare giorno. Anche gli home computer sono in grado, utilizzando le informazioni di appositi database, di eseguire i calcoli. Se collegato ad una serie di servo-motori "passo passo", un computer può guidare il puntamento di un telescopio

Se guardiamo il cielo di notte ed individuiamo la stella più vicina, stiamo in realtà osservando la stella com'era 4 anni fa, perché questo è il tempo che la luce proveniente da Proxima Centauri impiega ad arrivare sulla Terra. In questi stessi 4 anni i computer si sono trasformati, da rara e costosa novità quali erano, in apparecchi di uso oramai comune in molte case. L'astronomia è un campo di studio che sfrutta appieno le potenzialità, anche di un semplice home computer, nel trattamento delle informazioni, nel calcolo e nella robotica.

Esistono tre principali tipi di problemi che un astronomo deve affrontare quando studia il cielo: l'osservazione iniziale di un corpo celeste, il trattamento dei dati ottenuti mediante l'osservazione e l'analisi significativa di questi dati. Il computer è di valido aiuto in ciascuno di questi campi.

Iniziamo dall'esaminare come il computer possa assistere nei calcoli necessari per costruire lo strumento d'indagine più importante per un astronomo, ossia il telescopio. La progettazione e la sistemazione delle lenti e degli specchi in un telescopio sono fasi molto critiche, che incidono sulla qualità dell'immagine finale e si possono calcolare matematicamente per ottenere i migliori risultati. Gli astronomi dilettanti, in genere, preferiscono fabbricarsi i propri sistemi ottici ma, prima che i computer divenissero economicamente accessibili, spesso si passava direttamente a montaggi empirici, piuttosto che eseguire manualmente i faticosi calcoli necessari. Con un computer, i calcoli che, manualmente, potevano impegnare per intere settimane, si eseguono nel termine di qualche minuto.

Localizzare una stella nel cielo è un altro problema che il computer può contribuire a risolvere. Le stelle non sono corpi fissi, ma seguono precise traiettorie nel cielo (ovviamente, in conseguenza della rotazione terrestre) e la loro posizione celeste, inoltre, è soggetta a variazioni stagionali.

Il metodo per localizzare una stella è simile al sistema di latitudine e longitudine usato per la geografia terrestre. Se immaginiamo un sistema di coordinate proiettato sulla "faccia interna" della volta celeste notturna, ciascun corpo celeste può essere individuato mediante 2 coordinate chiamate "declinazione" ed "ascensione retta".

Pertanto, ogni oggetto celeste viene raffigurato su di una mappa stellare, in base alle sue coordinate astronomiche e le varie mappe, relative a ciascuna zona del cielo, vengono riunite in atlanti celesti. Questi atlanti, importantissimi per l'osservazione di pianeti o di altri oggetti che si muovono rispetto alle stelle fisse o per scoprire corpi celesti completamente nuovi come le comete, sono stati immagazzinati in "database" (archivi organizzati), anche su home computer. Le informazioni contenute nei database di questo genere, includono dati sulla brillantezza o luminosità di ciascun corpo celeste, sulla qualità spettrale della luce emessa (ottenuta facendo passare un fascio di luce attraverso

un prisma o analizzandola con uno spettrometro) e sul tipo e l'età della stella. Tutto ciò può essere visualizzato sullo schermo del computer, sotto forma di mappe che mostrano il cielo come lo si può vedere da qualsiasi latitudine ed in qualsiasi momento dell'anno. A causa della rotazione terrestre, bastano pochi minuti perché le stelle escano dal campo visivo di un telescopio fisso, anche se l'ottica è di dimensioni ragguardevoli.
Usando uno strumento che può mettere a fuoco soltanto una ristretta zona del cielo, è essenziale far ruotare continuamente il telescopio per compensare il movimento di rotazione della terra. Per molti anni questo lavoro è stato eseguito da motori meccanici ma, recentemente, gli astronomi dilettanti hanno potuto dotare il loro telescopio di sistemi di puntamento controllati da computer. Il telescopio viene montato su un asse "equatoriale", che punta verso il nord "effettivo", ed il motore ruota l'asse esattamente alla stessa velocità con cui si muove la Terra: questa rotazione mantiene l'oggetto osservato costantemente all'interno del campo visivo del telescopio. Sull'asse e sul telescopio sono inseriti piccoli dispositivi di codifica e digitalizzatori, che inviano al computer le informazioni necessarie al controllo del motore. In questo modo, un telescopio può essere lasciato tutta una notte, sotto l'esclusivo controllo del computer, ad inseguire una stella poco luminosa mentre la sua debole luce impressiona lentamente una lastra fotografica. Negli USA è in vendita un adattatore, chiamato "Celestial Navigator MkII", che serve a collegare il telescopio con il computer, mediante un'apposita interfaccia.
Sempre negli Stati Uniti, alcuni astronomi dilettanti hanno conseguito risultati veramente sorprendenti adattando alle loro necessità le più recenti tecnologie di rinoscimento e di sintesi vocale: quando un astronomo entra nell'osservatorio e pronuncia la parola "aprire", la cupola viene aperta; quando pronuncia le parole "ruotare la cupola", i motori entrano in azione e fanno ruotare la cupola sulle sue ruote. La sintesi vocale è molto utile anche nell'oscurità dell'osservatorio, poiché permette di ottenere informazioni dal computer in forma parlata. Quest'ultimo può anche scandire il tempo ad alta voce per aiutare l'astronomo ad eseguire accurate esposizioni fotografiche.
Gli astronomi professionisti usano telescopi che captano anche parte dello spettro elettromagnetico, come onde radio o raggi x. Con l'aumentare del diametro d'apertura dei telescopi e, di conseguenza, del loro peso, i problemi ingegneristici relativi alla costruzione dei telescopi stanno diventando critici. Nel 1964, il Mark II, un disco ellittico di 38 × 25 metri, situato vicino a Manchester (GB), fu il primo telescopio ad utilizzare un computer digitale per convertire le coordinate celesti in istruzioni necessarie ai suoi diversi motori: questo processo si ripeteva 4 volte al secondo, ininterrottamente.
I computer possono essere d'aiuto ai telescopi calcolando per loro i cambiamenti atmosferici, come ad esempio le variazioni di temperatura e d'umidità, che rifrangono e curvano la luce quando essa attraversa l'atmosfera. Inoltre, possono compensare e correggere le distorsioni delle immagini ricevute dai telescopi, usando varie tecniche di trattamento elettronico dell'immagine.
L'astronomia, grazie ai computer, ha fatto notevoli passi avanti, ma è vero anche il contrario: il linguaggio FORTH, per esempio, è stato ideato nel 1971 da un astronomo, Charles H. Moore, presso l'osservatorio di Kitt Peak in Arizona, con lo scopo di migliorare il controllo dei radiotelescopi e l'elaborazione dei dati rilevati con essi.
Ultimamente, si vanno diffondendo libri dedicati specificatamente alla programmazione di home computer per gli appassionati di astronomia ed esiste, in Gran Bretagna, anche una rivista, chiamata "Apex", attraverso la quale i possessori di piccoli computer si scambiano informazioni e programmi. Esistono programmi per calcolare il giorno in cui cadono le feste cosiddette "mobili", per la conversione delle date storiche convenzionali secondo il calendario giuliano, per calcolare il sorgere ed il tramontare della luna e le tabelle giornaliere occorrenti per prevedere l'esatto punto del cielo in cui gli astronomi si aspettano di veder apparire corpi celesti quali le comete.



JOHN DRYSDALE/COLORIFIC

UNA SCUOLA PIENA DI STELLE

Gli allievi e gli insegnanti della Kettering Grammar School, nel Northamptonshire, sono da lungo tempo tenuti in gran considerazione per i loro lavori in astronomia, soprattutto nel campo dell'individuazione dei satelliti. In alcune occasioni la KGS è stata la prima stazione d'osservazione ad individuare la presenza in orbita di un nuovo satellite



La voce delle stelle

In seguito alla scoperta, in lontane galassie, di corpi celesti radio emittenti, fu costruito questo imponente radiotelescopio a Jodrell Bank, nei pressi di Manchester. I radiotelescopi, che funzionano come potentissime antenne, possono individuare oggetti invisibili anche ai più potenti telescopi ottici

Un passo alla volta

Il codice Gray facilita il controllo digitale di spostamenti meccanici

Il codice Gray

La tabella mostra la corrispondenza tra il codice Gray, quello binario, e gli equivalenti decimali per numeri da 0 a 15

Decimale	Binario	Gray
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

In numerose applicazioni, è necessario determinare con assoluta precisione la posizione fisica di un oggetto mobile e, quindi, codificarla opportunamente, affinché un computer possa elaborare l'informazione. Con i robot, ad esempio, il computer deve sapere con esattezza la posizione e l'orientamento di ciascuna parte meccanica; in catene di lavorazione controllate da computer, è necessario conoscere le distanze tra i vari punti che richiedono l'impiego di particolari attrezzi. Ovviamente, tutte queste misurazioni devono essere convertite in valori comprensibili al computer, ossia in forma binaria.

Un metodo potrebbe consistere nell'adoperare un sistema analogico. La parte mobile verrebbe collegata ad una resistenza variabile e la variazione di tensione, ad essa applicata, dovrebbe essere passata ad un convertitore analogico/digitale (o direttamente al canale analogico del computer, se disponibile). Questo metodo non è molto affidabile e soprattutto è soggetto a rapida usura.

L'alternativa consiste nell'applicare alla parte in movimento una striscetta stampata, sulla quale è riportato graficamente un codice binario, che, letto da appositi rivelatori fotosensibili, viene utilizzato direttamente dal computer. In genere, il codice viene raffigurato mediante opportune alternanze di zone bianche e nere. Inoltre, variando il numero di strisce (e di conseguenza il numero di celle fotosensibili) si possono ottenere codici composti dalla quantità di bit necessari alla particolare applicazione. Se il movimento da misurare è di tipo angolare (la rotazione di un "gomito", ad esempio), s'impiegano dischetti sui quali il codice è riportato in anelli concentrici.

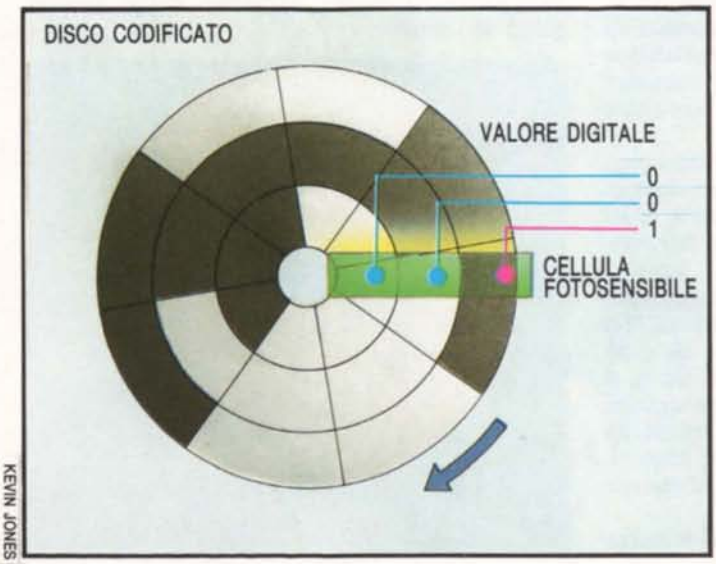
I problemi sorgono durante il momento di passaggio tra un valore del codice e quello successivo, oppure se il meccanismo si ferma proprio a metà tra due valori, producendo una lettura che potrebbe risultare errata, dovuta alla normale tolleranza dei dispositivi di lettura. Se, ad esempio, il meccanismo si ferma tra i valori 11 (binario 1011) e 12 (binario 1100), l'unica lettura sicuramente esatta è la cifra binaria più significativa (quella più a sinistra, comune ad ambedue i valori). In certe situazioni, addirittura, le certezze sono ancora minori: quando, per esempio, si passa dal valore 7 (binario 0111) ad 8 (binario 1000) e le quattro cifre binarie sono tutte incerte.

La normale numerazione binaria, quindi, risulta poco efficiente e si è ricorsi a una diversa sequenza di codici, nella quale, passando da un valore a quello successivo, cambia soltanto uno dei bit che lo compongono.

Un codice, largamente impiegato, con questo requisito, è il codice Gray: passando da un valore a quello successivo, cambia soltanto il bit più a destra.

Facciamo un esempio: partendo da 0000, il successivo valore è 0001. Ma, a differenza del normale codice binario, il 2 è rappresentato con 0011 (si ricordi che si può modificare un solo bit alla volta). Pertanto, il valore 3 corrisponde a 0010, e via dicendo (una tabella del codice Gray è riportata qui a fianco, per i valori da 0 a 15, assieme agli equivalenti decimali e binari).

Le letture effettuate mediante questo codice vengono successivamente convertite, da appositi programmi, nei valori binari corrispondenti, in modo tale da permettere al computer di eseguire le normali operazioni aritmetiche e logiche sui dati così acquisiti.



KEVIN JONES

LA CORRETTA ANGOLAZIONE

L'esatta posizione di un meccanismo soggetto a rotazione può essere determinata applicando un disco sul quale è riportato uno speciale codice. Un fascio di luce, diretto sul disco, viene riflesso o meno dalla superficie del disco e rilevato da una serie di apposite celle fotosensibili. Il codice generato è di tipo parallelo ed il numero di bit che lo compongono dipende dal numero di anelli concentrici sul disco (che deve corrispondere al numero di celle fotosensibili). Il codice più diffuso, per questo tipo di applicazioni è il Gray

L'Apple IIe

Anatomia di un microcomputer a diffusione europea: tecnologie a confronto

Per molti versi, l'Apple II è l'apparecchio che ha iniziato l'era degli home computer. Pur non essendo il primo in assoluto, è stato grazie all'Apple che sofisticate prestazioni (quali una grafica ad alta risoluzione, a colori, e la generazione di effetti audio) si sono diffuse in apparecchi economici. Tuttavia, sono forse state alcune caratteristiche meno appariscenti, oltre a quelle menzionate, che hanno decretato il successo di questo microcomputer.

Il livello della documentazione, per esempio, è tale che anche gli utenti inesperti riescono immediatamente ad usare l'apparecchio. A suo tempo, ciò rappresentò una grossa innovazione, poiché la tendenza delle case costruttrici era quella di nascondere, piuttosto che palesare ampiamente, i segreti dei propri apparecchi. Questa impostazione, che purtroppo alcune marche continuano ad ignorare, è stata accolta molto favorevolmente dal largo pubblico.

La seconda peculiarità dell'Apple II è la possibilità di collegare ad esso una vasta gamma di accessori e periferiche. L'Apple ha trovato innumerevoli applicazioni, proprio grazie alla considerevole capacità d'espansione, tanto che anche molte altre ditte, oltre alla stessa Apple, hanno messo in commercio moltitudini di interessantissimi apparecchi aggiuntivi.

Nella generalità dei piccoli computer, i connettori per espansioni sono al massimo due, e, per di più, l'estensione di memoria associata è molto limitata. Nella gamma Apple sono sette (nel recente modello IIe) oppure otto (nei meno recenti II e II+), ciascuno dei quali è "visto" dalla CPU come due sezioni di memoria (una piuttosto piccola, l'altra di circa 2 Kbyte).

Di conseguenza, una scheda aggiuntiva, da collegare all'Apple, può esser dotata di ben 2048 Byte di programma "residente". Ciò significa che non è necessario né apportare modifiche al sistema operativo né scrivere appositi programmi che permettano l'impiego di dette schede.

Le schede oggi disponibili per l'Apple sono numerosissime e vanno da semplici interfacce per I/O (Input/Output) a sofisticati dispositivi (penne ottiche, espansioni di memoria superiori al Megabyte, ecc.). Tuttavia, poiché la CPU 6502 è capace d'indirizzare soltanto 64 Kbyte, la memoria viene suddivisa in "banchi", selezionabili uno alla volta. Inoltre, all'Apple si possono collegare computer dotati di chip a 16 o 32 bit, ottenendo sistemi di prestazioni a dir poco eccezionali.

Ovviamente, contemporaneamente a questo colossale sviluppo della parte hardware, si è potuto assistere ad una altrettanto enorme crescita di software: si calcola che la biblioteca di programmi disponibile per l'Apple sia addirittura superiore a quella, di per sé ricchissima, utilizzabile sotto il sistema operativo CP/M. Senza considerare il fatto che, grazie ad una scheda d'espansione dotata di un microcomputer Z80, è possibile usare il CP/M sull'Apple.

Nonostante le piccole dimensioni ed un aspetto quasi dimesso, questo microcomputer ha qualità superlative. È l'home computer sul quale "gira" il maggior numero di sistemi operativi (11), linguaggi (almeno 27), programmi "editor" (più di una dozzina).

Il cammino dell'Apple II non si è certo concluso: una nuova versione del sistema operativo, chiamata ProDOS, offre prestazioni paragonabili a sistemi molto più costosi.



CHRIS STEVENS

LA TASTIERA DELL'APPLE

La tastiera è di ottima fattura, evidentemente progettata per dattilografi abituati alle comuni macchine da scrivere. I tasti sono leggermente incavati e l'impressione d'uso che se ne ricava è molto buona. Due tasti, contrassegnati col marchio della Apple e posti lateralmente alla barra spaziatrice, servono come tasti di controllo nei programmi applicativi. Il tasto di RESET, la cui posizione può sembrare pericolosamente vicina al tasto DELETE (cancellazione), non può essere usato accidentalmente, in quanto funziona soltanto se premuto contemporaneamente al tasto CTRL



Il monitor
L'Apple IIe funziona con qualsiasi tipo di monitor per microcomputer, sebbene quello prodotto dalla stessa Apple si adatti meglio di altri sotto il profilo estetico e sia dotato di un filtro anti-riflesso. Sono visualizzabili righe di 80 caratteri

CHRIS STEVENS

CPU 6502

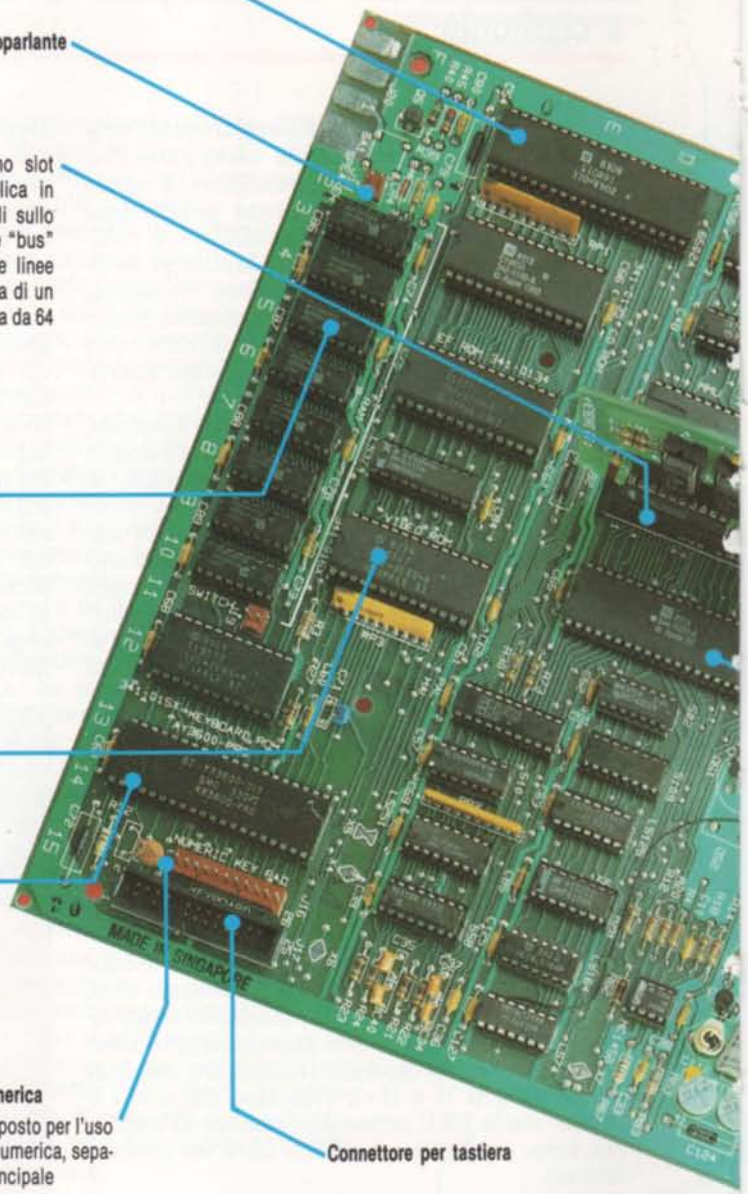
Presa per altoparlante

Connettore ausiliario per espansioni
Nell'Apple IIe, esiste uno slot supplementare, che duplica in parte i segnali disponibili sullo slot n. 3: oltre al normale "bus" di segnali, possiede altre linee che consentono l'aggiunta di un secondo banco di memoria da 64 Kbyte

RAM
L'Apple IIe è dotato di 64 Kbyte di RAM, contenuti in otto chip 4164, ma la gestione della memoria è assai particolare, rispetto ad altri computer, onde permettere la gestione di più banchi di memoria

ROM del video
I caratteri visualizzabili sullo schermo sono generati da questo chip di ROM. Diverse versioni di questa ROM consentono di visualizzare caratteri e simboli speciali

ROM della tastiera
Questa è la ROM necessaria alla gestione della tastiera. È possibile, sostituendola, adattare l'interpretazione dei tasti secondo criteri diversi dallo standard QWERTY



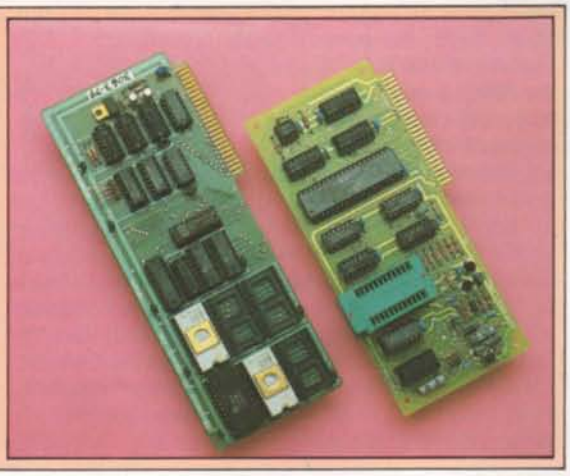
Connettore per tastierina numerica
L'Apple II è predisposto per l'uso di una tastierina numerica, separata da quella principale

Connettore per tastiera



Unità a dischi
I disk drive della Apple sono del tipo "non intelligente", pertanto occorre impiegare un'apposita scheda per il funzionamento. La capacità dei dischetti, di 143 Kbyte, è bassa rispetto alla media, benché sufficiente per molte applicazioni

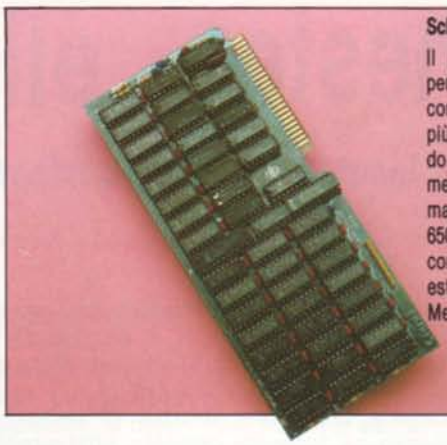
Il programmatore di EPROM e la scheda ROM
Ci sono innumerevoli strumenti di sviluppo applicabili all'Apple: tra questi un programmatore di memorie EPROM, grazie al quale è possibile trasferire su memorie permanenti programmi sviluppati nella normale RAM. Il chip di memoria, contenente il nuovo programma, viene poi montato su di una scheda d'espansione che consente l'uso del programma impegnando solo parzialmente la RAM



Unità per la gestione della memoria

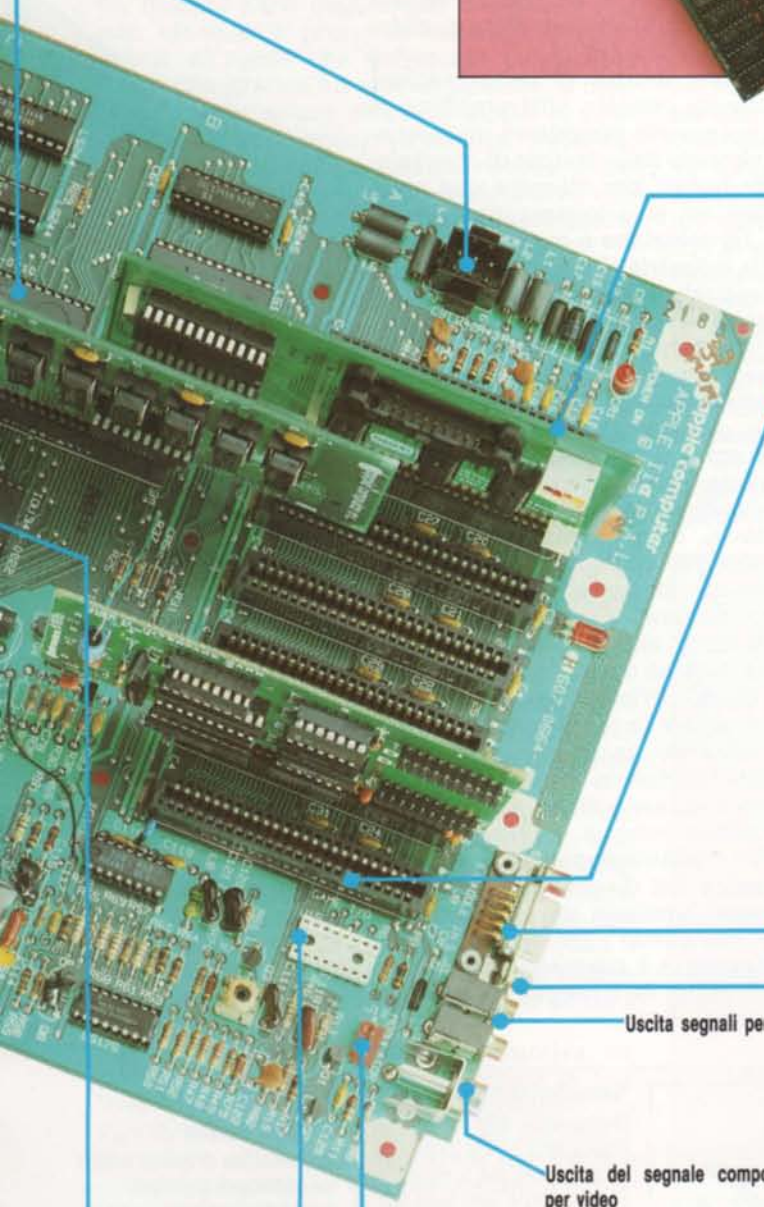
Questo chip è una delle due ULA (Uncommitted Logic Array) che differenziano l'Apple II dalla versione IIe. Esso si occupa in parte della gestione del video ad 80 colonne ed in parte del secondo banco di memoria RAM da 64 Kbyte

Presa d'alimentazione



Scheda RAM da 256 Kbyte

Il particolare sistema adottato per la gestione della memoria, consente all'Apple di indirizzare più "banchi" di memoria, offrendo una capacità totale notevolmente superiore ai 64 Kbyte normalmente indirizzabili dalla CPU 6502. La scheda qui riprodotta contiene 256 Kbyte di RAM, estendibili ad addirittura oltre 1 Megabyte



Slot 1

A questo connettore viene collegata, in genere, una stampante parallela

Slot 7

I vari segnali speciali di uscita video sono disponibili soltanto su questo connettore, al quale vengono normalmente collegate penne ottiche, modulatori di colore, ecc.

Connettore per la porta dei giochi

Una delle maggiori innovazioni dell'Apple è stata quella di fornire una porta di input/output per segnali, tra i quali un ingresso analogico, utilizzabile per i giochi

Ingresso segnali da registratore

Uscita segnali per registratore

Uscita del segnale composito per video

Connettore ausiliario per video

Connettore per la porta dei giochi

L'Apple IIe dispone di due tipi di connettori, posti in parallelo, relativi ai segnali impiegati nei giochi

Unità di Input/Output

Questo chip provvede alla gestione delle unità periferiche di ingresso/uscita

APPLE IIe

DIMENSIONI

mm. 460 x 385 x 115

CPU

6502

CLOCK

1 MHz

MEMORIA

16 Kbyte di ROM, 64 Kbyte di RAM, estendibile a 128 Kbyte ed oltre, mediante gestione "a banchi"

VIDEO

24 linee di 40 caratteri, un solo colore. Grafica in bassa risoluzione, 48 x 40, in 16 colori. Grafica in alta risoluzione, 192 x 280, in 6 colori

INTERFACCE

Connettore per registratore, per video composito, 7 connettori (slot) per espansioni, porta per giochi

LINGUAGGI FORNITI

BASIC Applesoft

ALTRI LINGUAGGI DISPONIBILI

La maggior parte dei comuni linguaggi, oltre a qualche linguaggio esclusivamente disponibile sull'Apple

ACCESSORI COMPRESI

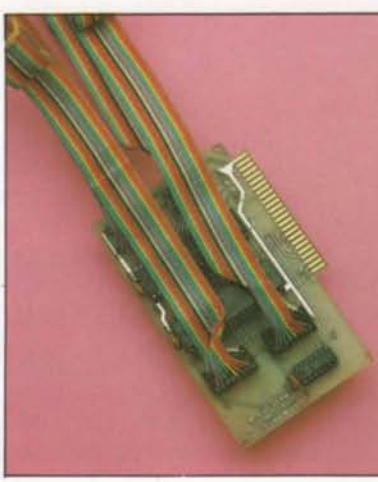
Cavo di connessione video, manuale d'installazione e manuale BASIC

TASTIERA

62 tasti di alta qualità

DOCUMENTAZIONE

La documentazione di corredo è di altissima qualità, benché parte del materiale utile a maggiori approfondimenti tecnici venga venduto separatamente. Esistono molte pubblicazioni che trattano il funzionamento e le applicazioni dell'Apple



Scheda I/O per usi generici

Talvolta, una scheda può offrire tali e tante funzioni diverse, che l'impiego di un sistema di controllo basato su ROM diverrebbe altamente limitativo. È il caso della scheda illustrata, che contiene due versatili chip 6522 (chiamati "adattatori d'interfaccia"), consentendo l'uso di ben 40 diverse linee di controllo. Due "registri a scorrimento" (shift register) convertono i segnali dalla forma parallela a quella seriale. Sono inoltre presenti quattro circuiti di temporizzazione a 16 bit

CHRIS STEVENS

I dischi Winchester

Megabyte a profusione, anche per i computer più piccoli, con i dischi "rigidi"

In questi ultimi anni, i microcomputer hanno assimilato molte delle caratteristiche presenti sugli elaboratori per uso commerciale, ma c'è un settore in cui essi sono rimasti relativamente arretrati: le dimensioni delle memorie a dischi magnetici.

Mentre il possessore di un home computer può spesso accontentarsi di un singolo dischetto capace di contenere 100 Kbyte, gli elaboratori per uso commerciale necessitano di capacità molto più ampie. Per questa ragione sono stati sviluppati i cosiddetti "dischi fissi" (o anche "rigidi"), con capacità di svariati Megabyte. L'IBM, intorno al 1960, ha svolto un lavoro d'avanguardia in questo campo ed i primi dischi, che possedevano una capacità di 30 Megabyte, vennero fatti funzionare "in doppietta", aumentando del 100% la loro capacità. Per analogia con il famoso fucile, furono chiamati "Winchester".

I dischi Winchester sono costituiti da dischi rigidi, al contrario dei dischetti (floppy disk), realizzati in materiale plastico flessibile. Questa caratteristica ha permesso di aumentare il numero di tracce presenti sul disco: nei dischetti abitualmente il numero massimo di tracce per pollice è di 96, mentre nei dischi Winchester si possono avere parecchie centinaia di tracce per pollice (in inglese: tracks per inch, abbreviato in tpi). Grazie allo sviluppo di sofisticate tecnologie di costruzione è possibile ottenere capacità d'immagazzinamento di 5, 10 o anche 20 Megabyte, in contenitori simili a quelli delle unità a dischetti da 5 e 1/4 pollici.

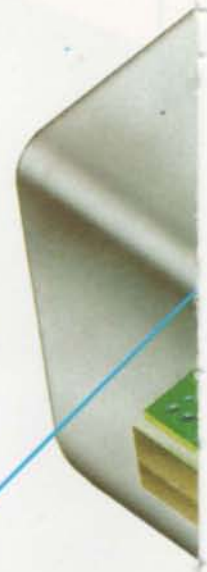
I possessori di home e personal computer cominciano già a beneficiare degli sviluppi derivati dalla tecnologia dei dischi rigidi: infatti, sono oggi disponibili microdischetti (i Sony da 3,5 pollici e gli Hitachi da 3 pollici, ad esempio), che rappresentano una via di mezzo tra i dischi "floppy" e gli "hard", capaci d'immagazzinare molti più dati di un qualsiasi dischetto da 5 e 1/4 pollici.

Questa crescente densità di immagazzinamento ha tuttavia causato altri problemi. È stato infatti necessario progettare un meccanismo di movimento della testina di concezione totalmente nuova, per ottenere una maggior precisione nel posizionamento della testina stessa. La soluzione a questo problema proviene dalle industrie che operano nel campo della riproduzione del suono: le testine, nei Winchester, vengono sempre più spesso azionate da una bobina elettromagnetica del tipo impiegato negli altoparlanti.

La testina, letteralmente, "vola" sulla superficie del disco, poggiando su di un cuscino d'aria, senza entrare in diretto contatto con la sua superficie. Questo accorgimento riduce in modo significativo l'usura del disco, ma implica, tuttavia, la necessità di sigillare i dischi in involucri a tenuta d'aria, onde evitare i gravissimi danni causati dalla polvere o da altri corpi estranei. Ciò significa, ovviamente, che il disco non può essere rimosso dall'involucro, benché siano da poco disponibili anche dischi Winchester a cartucce estraibili. Per evitare la penetrazione di qualsiasi corpo estraneo, la pressione dell'aria all'interno della cartuccia viene mantenuta, grazie ad una speciale pompa, ad un livello più alto di quella esterna e, inoltre, l'aria pompata viene costantemente filtrata. Un'altra caratteristica dei dischi rigidi è che, in realtà, essi sono costituiti da più dischi sovrapposti "a castelletto": un disco Winchester da 10 Megabyte è normalmente composta da 2 dischi rigidi da 5 Megabyte collocati nello stesso contenitore.

Contenitore

La maggior parte delle unità Winchester sono contenute in un involucro sigillato per pressofusione, in parte responsabile del loro notevole peso, necessario del resto per mantenere perfettamente allineate le varie componenti dell'apparecchio

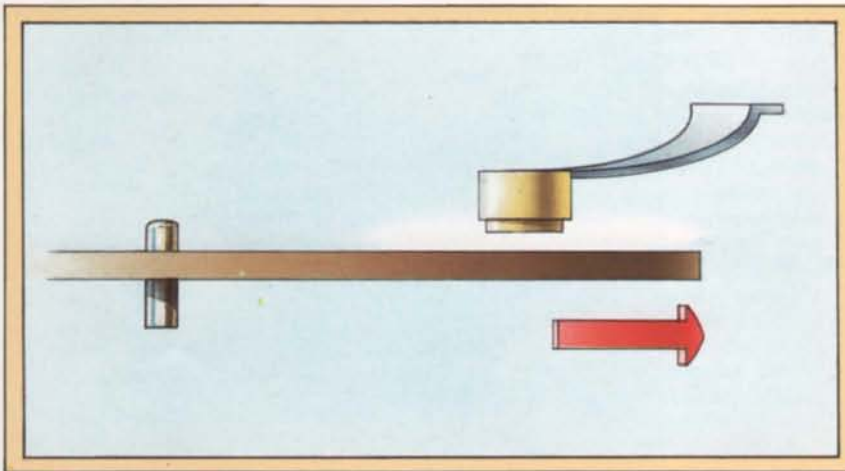


Dischi rigidi

La superiore capacità d'immagazzinamento delle unità Winchester dipende esclusivamente dalla presenza di più dischi impilati su uno stesso asse. L'unità in figura ne ha quattro, ma la maggior parte ne ha due o tre

Saldatura ermetica

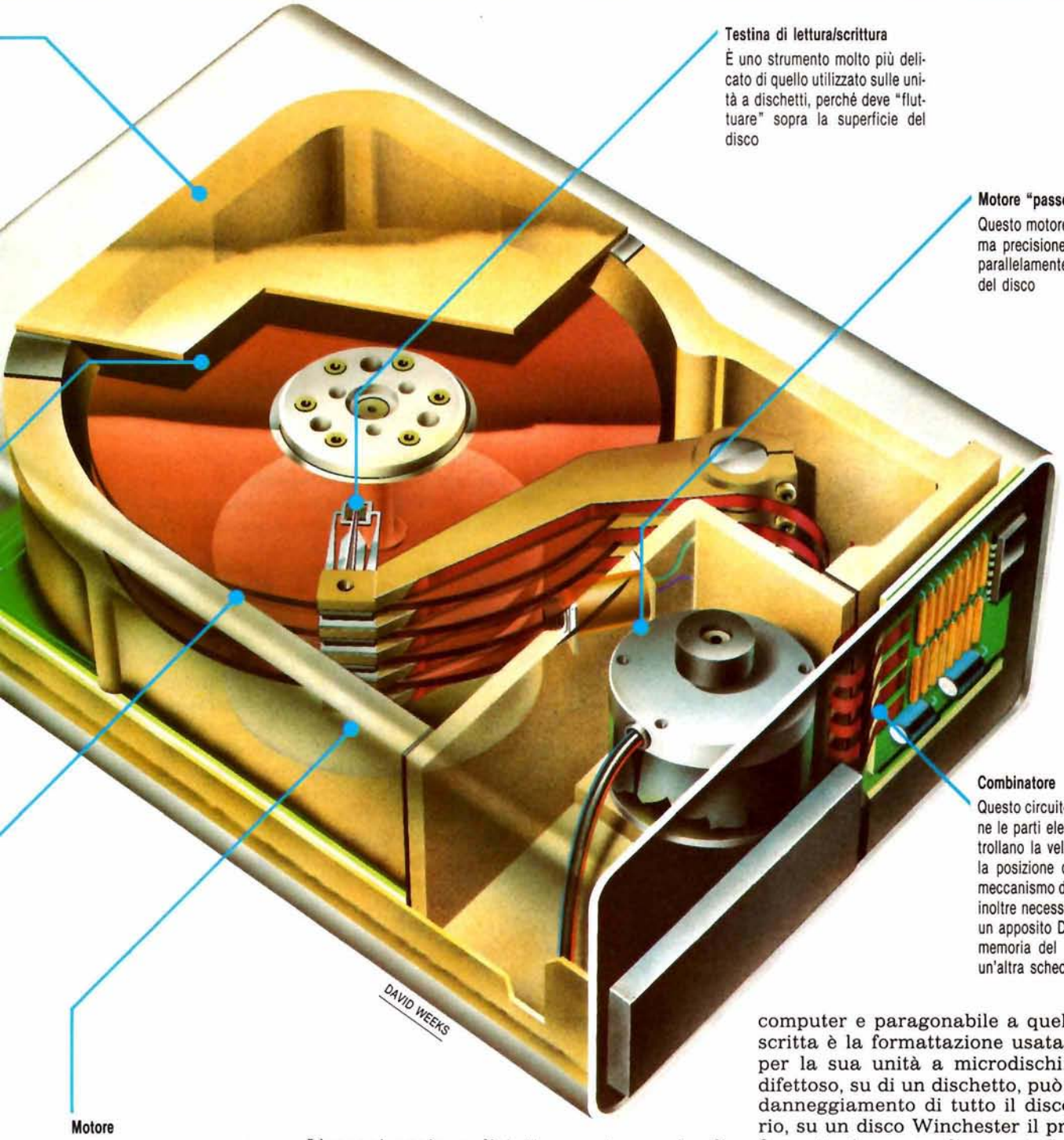
La meccanica di questa unità è completamente sigillata, per evitare che polvere o altre particelle atmosferiche danneggino la superficie dei dischi



Testine fluttuanti

Diversamente dalle unità a dischetti, nelle quali la testina entra in contatto con la superficie di registrazione, sulle unità Winchester la testina "fluttua", o "vola" ad una distanza infinitesimale dalla superficie del disco, ma senza venirne mai a contatto. Il disco ruota così velocemente che si crea un cuscino d'aria sufficiente per sorreggere la testina. Se essa dovesse accidentalmente urtare il disco, la superficie magnetica di registrazione verrebbe irreparabilmente danneggiata

KEVIN JONES



Testina di lettura/scrittura
È uno strumento molto più delicato di quello utilizzato sulle unità a dischetti, perché deve "fluttuare" sopra la superficie del disco

Motore "passo passo"
Questo motore elettrico di estrema precisione muove la testina parallelamente alla superficie del disco

Combinatore
Questo circuito stampato contiene le parti elettroniche che controllano la velocità di rotazione, la posizione della testina ed il meccanismo di lettura dei dati. È inoltre necessaria la presenza di un apposito DOS, caricato nella memoria del computer o su di un'altra scheda

Motore
Un motore a corrente continua ed un piccolo generatore sono montati assieme sullo stesso perno che di volta in volta fa ruotare il disco. L'output dal generatore fornisce una misura della velocità del motore ed interagisce con uno speciale circuito di controllo. In questo modo viene determinata con precisione la velocità di rotazione

L'organizzazione di tutto questo spazio, disponibile per l'archiviazione di dati, viene conseguita suddividendo lo spazio stesso in numerose sezioni. Per conservare la compatibilità con il software caricato sul computer, che spesso è programmato per gestire solo dischetti, ciascuna di queste sezioni possiede approssimativamente la capacità di un normale dischetto.
Quando un disco Winchester viene formattato (quando, cioè, con un apposito programma vengono delimitate per la prima volta tracce e settori), il DOS (vedere a pag. 324) deve essere in grado di saltare automaticamente quei settori che dovessero presentare difetti sulla superficie magnetica di registrazione.
L'unica operazione presente su di un home computer e paragonabile a quella sopra descritta è la formattazione usata sul Sinclair per la sua unità a microdischi. Un settore difettoso, su di un dischetto, può provocare il danneggiamento di tutto il disco. Al contrario, su un disco Winchester il programma di formattazione semplicemente "prende nota" dell'indirizzo del settore inutilizzabile e ne rende impossibile un successivo impiego. Dopo tutto, con cinque milioni di byte a disposizione, chi mai sentirà la mancanza di poche centinaia di byte?
In armonia con i progressi verificatisi negli altri settori dell'industria dei computer, le dimensioni del disco Winchester stanno divenendo sempre più piccole: esiste anche un Micro-disco Winchester da 5 Megabyte. Con simili rapidi sviluppi nel campo delle unità di memoria a dischi magnetici, non è poi così lontano il giorno in cui si potrà acquistare un home computer, capace di memorizzare 10 Megabyte di dati, ad un prezzo di poco superiore a quello di un videogioco.

Dai moduli al programma intero

Mettiamo assieme le parti di programma sviluppate finora per la rubrica computerizzata

Benché molti dettagli della nostra rubrica computerizzata debbano ancora essere stabiliti, la struttura generale del programma dovrebbe essere sufficientemente chiara. A questo punto, nello sviluppo di un qualsiasi programma, torna molto utile formulare un diagramma a blocchi (flowchart), che permetta di sintetizzare le funzioni delle varie parti del programma.
Questo è anche il momento di considerare l'impostazione dell'"interfaccia umana" e dell'"immagine" nei confronti del futuro utente del programma. Troppo spesso, anche nel progetto di software commerciale, questa impostazione non viene purtroppo rispettata. L'"interfaccia umana" consiste nella facilità d'impiego del software. L'"immagine" è, invece, l'impressione generale che l'utente ricava dall'uso dei programmi. Vediamo come e quanto è possibile tener conto di questi due elementi nella realizzazione del nostro programma.
La seguente tabella illustra i blocchi principali della "rubrica computerizzata" finora esaminati. Per semplicità, i nomi delle subroutine e delle procedure sono di soli sei caratteri, quelli delle stringhe di sette caratteri (includendo il $), quelli delle variabili numeriche semplici di quattro caratteri, mentre, per le variabili alfanumeriche "globali", abbiamo usato cinque caratteri (includendo il $). Per le variabili "locali" è stata impiegata una sola lettera.

BLOCCHI PRINCIPALI DEL PROGRAMMA

	CREMAT	(creazione delle matrici ed inizializzazione delle variabili)
INITIL	RDINFL	(lettura del file da nastro o disco)
	SETFLG	(modifica le "flag" ed alcune variabili)
GREETS		(messaggio iniziale)
CHOOSE	CHMENU	(presentazione menu)
	INCHOI	(input della scelta)
	FNDREC	(trova e visualizza un record)
	FNDNMS	(trova un record usando un nominativo parziale)
	FNDTWN	(trova i record in base alla città)
	FNDINT	(trova un record usando le iniziali)
EXECUT	LSTREC	(elenca tutti i record)
	AGGREC	(aggiunta di un record)
	MODREC	(modifica di un record)
	DELREC	(cancella un record)
	EXPROG	(salva il file ed esce dal programma)

Ciascun blocco principale della seconda colonna dovrà essere suddiviso in più parti elementari, che, a loro volta, verranno definite in dettaglio prima di passare alla codifica in BASIC. Il procedimento è stato già illustrato, nelle passate lezioni, per alcuni blocchi. Supponendo che i vari blocchi siano stati sviluppati, raffinati ed, infine, codificati e verificati come moduli individuali, come si procede per riunificare tutte le varie componenti in un unico blocco di programma? Il miglior metodo consiste nell'archiviare (su nastro o su disco) ciascun modulo con lo stesso nome assegnatogli durante la fase di sviluppo. Generalmente, nel "caricare" (richiamare) un programma in memoria, si usa il comando LOAD, seguito dal nome del programma (ad esempio: LOAD "AGGREC"). Ciò presenta, però, un inconveniente: il contenuto della memoria viene azzerato prima del caricamento del programma, pertanto, se carichiamo AGGREC e, di seguito, carichiamo il modulo MODREC, quest'ultimo si sovrappone e cancella AGGREC.
Esiste, nella maggior parte dei BASIC, un comando molto utile in questi casi: il MERGE (lett.: fusione). Il suo funzionamento è simile al LOAD, ma il programma caricato per secondo viene "accodato" a quello già presente in memoria. L'unica precauzione da prendere, nell'impiego del comando MERGE, è che il numero di linea iniziale della seconda porzione di programma deve essere superiore a quello finale della prima porzione, altrimenti potrebbero verificarsi accavallamenti tali da compromettere il buon esito dell'operazione. L'inconveniente è facilmente aggirabile, se la versione di BASIC possiede il comando RENUM (letteralmente: "rinumera"), che permette di riorganizzare automaticamente la numerazione di un programma prima di archiviarlo. Ma, non essendo sempre disponibile un tale comando, a maggior ragione è necessario prestare attenzione anche alla numerazione delle linee dei vari moduli già fin dall'inizio. Una volta preparato un piano generale del programma (come quello riportato nella pre-

cedente tabella), è già possibile attribuire il numero di linea iniziale di ciascun modulo. Se dovessimo prevedere sostanziali modifiche o aggiunte a certe parti di programma, potremmo assegnar loro una numerazione con incrementi di 50 in 50, o, addirittura, di 100 in 100, per lasciar sufficiente spazio all'inserimento di nuove linee di programma. Se la particolare versione di BASIC non prevede il comando MERGE, allora sarà necessario scrivere il programma aggiungendo manualmente modulo per modulo.
Il listato riportato più avanti è un semplice esempio sull'impiego del comando MERGE: i vari moduli corrispondenti a quelli della tabella, "fusi" assieme in un blocco unico, sono sviluppati soltanto a livello di verifica, e non costituiscono certo la versione definitiva.
Il primo blocco del programma è INITIL, che dovrebbe occuparsi d'inizializzare le variabili, di dimensionare le matrici, assegnare valori alle flag, e via dicendo.
Questa subroutine è suddivisa in CREMAT (per la creazione delle matrici), in RDINFL (per la lettura dei file ed il trasferimento dei dati nelle varie matrici) ed, infine, in SETFLG (per l'assegnazione delle flag).
Dopo queste operazioni preliminari, si passa a GREETS, una subroutine per la visualizzazione di un messaggio d'inizio. La sua ultima parte prevede che l'operatore digiti uno spazio per proseguire nell'esecuzione.
Si giunge così a CHOOSE, una subroutine composta da due parti: la prima presenta un elenco delle opzioni disponibili, la seconda prevede l'immissione della scelta e la corrispondente assegnazione di un valore (numerico) alla variabile CHOI.
Detto valore viene utilizzato nel successivo blocco del programma, EXECUT, per selezionare uno fra nove blocchi di programma diversi, che, una volta eseguito (con l'eccezione di EXPROG, che consente l'uscita dal programma), deve fare in modo che l'esecuzione ritorni al modulo CHOOSE, permettendo una nuova scelta.
Il principale inconveniente del programma, nella forma presentata, è che il controllo sul flusso d'esecuzione non è corretto. Per esempio, INITIL prevede la lettura di un file dalla memoria di massa senza che venga controllata l'esistenza o meno del file. La prima volta che il programma viene lanciato, non esisterà ancora alcun archivio di dati e, quindi, il BASIC si bloccherà, avendo individuato una condizione d'errore.
Un corretto procedimento sarebbe quello di invocare l'uso del modulo RDINFL da uno dei moduli di EXCUT, per una sola volta durante il lancio di un programma. Ciò suggerisce l'impiego di una flag, chiamata per esempio INFL, il cui valore, inizialmente 0, è posto ad 1 qualora il file venga effettivamente letto. Successivamente, controllando questa flag, si evita il tentativo di una nuova lettura. Il modulo AGGREC provvede a cercare, nella matrice, il primo elemento libero in cui si possa depositare un nuovo record. Poiché detto record, quasi certamente, non si troverà nella corretta sequenza di riordino, dovremo utilizzare un'altra flag, RMOD, che assumerà valore 1 ogni qualvolta viene aggiunto, modificato o cancellato un record. Si provi ad aggiungere, per esercizio, queste modifiche al programma.
L'inserimento, la modifica o l'annullamento di un record sono tutte operazioni che possono influenzare l'ordine logico degli elementi, cosicché il valore di RMOD andrebbe controllato in più parti del programma, per verificare se sia stata effettuata qualche modifica all'ordine degli elementi. Se questo è il caso, possiamo decidere se effettuare un riordino (prima di qualsiasi altra operazione) o se accettare il fatto che l'ordine degli elementi può non essere corretto, e proseguire. Il modulo EXPROG dovrebbe, comunque, controllare il valore di RMOD, in modo da ordinare i dati, se necessario, prima di archiviarli e di uscire dal programma.
Gli aspetti relativi all'"interfaccia umana" si possono suddividere in:

Interfaccia con l'operatore
Immagine offerta all'operatore
Recupero a seguito di errori
Livello di sicurezza
Adattabilità

L'interfaccia con l'operatore riguarda le modalità relative allo scambio d'informazioni tra l'operatore ed il programma. Noi abbiamo scelto il sistema basato sull'impiego di "menu", preferendolo a quello fondato sull'impostazione di una serie di comandi speciali. Il criterio adottato, qualunque esso sia, deve essere però coerente: non c'è niente di più scomodo da usare di un programma nel quale vengono mescolati i due sistemi, o, peggio ancora, gli stessi comandi vengono impiegati con differenti scopi nelle varie parti di uno stesso programma.
Nel nostro programma, allo stato attuale, questo aspetto è molto trascurato: il messaggio iniziale rimane in attesa che l'operatore digiti un carattere "spazio", il menu delle opzioni è terminato con l'immissione di un numero (da 1 a 9), e l'immissione dei dati di un record viene, invece, terminato con la pressione del tasto RETURN. Un'incoerenza simile sarebbe inaccettabile in programmi applicativi da mettere in commercio.
L'immagine offerta all'operatore dipende dall'andamento generale del programma, e l'interfaccia con l'operatore ha su di essa una notevole influenza. In tutti i programmi, una sostanziosa parte del lavoro viene eseguita senza che l'operatore se ne avveda direttamente: le uniche indicazioni sul progresso delle varie fasi di elaborazione provengono da eventuali messaggi visualizzati sullo schermo. L'"immagine" del nostro program-

ma dovrebbe richiamarsi all'impiego di una normale rubrica telefonica manuale, così come l'immagine desiderabile, per un word processor, sarebbe quella di presentare all'utente un "foglio di carta" elettronico, sul quale scrivere. Raramente si riesce ad ottenere un'immagine perfetta (è piuttosto improbabile che per trovare un nome in un'agendina sia necessario "PREMERE IL TASTO 4"!).

Tuttavia, l'impressione generale può essere migliorata sensibilmente, ricorrendo a "videate" (ossia, gruppi d'informazioni e messaggi riportati sullo schermo) ben bilanciate ed omogenee con il resto del programma. Una o due righe dello schermo, dovrebbero essere riservate ai messaggi d'errore ed alle varie comunicazioni dirette all'operatore, evitando di cambiar loro posizione ad ogni nuova visualizzazione. Inoltre, un buon programma tiene l'operatore costantemente informato sulle operazioni in corso, mediante sintetici messaggi. Nel software commerciale non è affatto rara la possibilità di visualizzare, in qualsiasi momento e senza pregiudicare l'operazione in corso, una o più pagine di informazioni relative alle varie operazioni (i cosiddetti "help menu", o menu di assistenza).

Un altro aspetto da evitare è l'incompleta visualizzazione dei dati sui quali si sta lavorando: è pericoloso, infatti, non comunicare all'operatore che parte delle informazioni sono, magari, contenute in un "buffer" o in un file "d'appoggio", rischiando così che vengano ignorate o, peggio ancora, cancellate per errore. Questo tipo di difetto, purtroppo, è diffuso anche in certi database commerciali. In ultima analisi, l'operatore non si sente a proprio agio lavorando in simili condizioni.

Un altro fattore molto importante è il recupero a seguito di errori ("error recovery"). La maggioranza dei BASIC avverte il programmatore con opportuni messaggi, qualora venga rilevato un errore formale nel programma o un'istruzione venga mal impiegata, ma ciò non costituisce alcun vantaggio per chi usa il programma (salvo forse il messaggio REDO, a seguito di un'errata immissione di dati).

Il trattamento degli errori ha due aspetti: la visualizzazione di un messaggio d'errore e la procedura di recupero da detto errore. Un notissimo word processor, per esempio, cura molto bene il primo aspetto, ma trascura il secondo: nella fase di archiviazione, magari dopo aver steso una lunga relazione, il programma avverte che lo spazio sul disco è esaurito, ma non indica alcun modo per salvare il lavoro fatto!

Qualsiasi operazione che comporti la cancellazione o la modifica di dati (MODREC nella nostra rubrica) dovrebbe sempre esser seguita da una richiesta di conferma del tipo: "QUESTA OPERAZIONE CANCELLA IL RECORD, CONFERMA (SI/NO)?". Nel caso di un word processor, un messaggio potrebbe essere: "QUESTO COMANDO ARCHIVIA IL TESTO, MA CANCELLA LA VERSIONE PRECEDENTE, OK (SI/NO)?".

Talvolta, il trattamento degli errori è complesso quanto il programma principale, poiché occorre prevedere ogni possibile fonte d'errore e fornire, di conseguenza, opportune funzioni di recupero, in particolare quando si rischia di cancellare informazioni preesistenti.

Vediamo, adesso, cosa intendiamo per livello di sicurezza. A seguito di eventi accidentali (quali ad esempio un'improvvisa mancanza di corrente) quali danni è possibile evitare? Qual è la quantità di dati che rischiamo di perdere? Potremmo limitarla al minimo? Esistono programmi talmente sofisticati che, anche a seguito di simili incidenti, permettono di recuperare totalmente tutto il lavoro, ma i sistemi adottati richiedono, in genere, grandi capacità nella memoria di massa e tempi di accesso ad essa molto rapidi.

L'adattabilità, infine, è la caratteristica che permette, appunto, di adattare il programma alle più varie esigenze, apportando poche e semplici modifiche. Non esistono regole precise in merito, ma si possono dare alcuni suggerimenti. Se nella numerazione delle linee di programma si procede per incrementi ragionevoli (di 50 in 50, ad esempio), rimane spazio sufficiente per l'inserimento di nuove linee, qualora occorra. Similmente, un abbondante uso di commenti (preceduti dall'istruzione REM), oltre a documentare opportunamente i vari stadi del programma possono in seguito essere sostituiti da linee contenenti istruzioni (attenzione però a non mangiare troppa memoria con lunghi commenti). Infine, nel definire o dimensionare le matrici, può essere saggio abbondare, in previsione di future espansioni. L'unico fatto certo è che viene sempre in mente qualche nuova aggiunta da apportare ad un programma, per migliorarne le prestazioni ed aver previsto ciò fin dal principio permette di evitare una completa ristrutturazione del programma.

Basic & Basic

SPECTRUM

Si veda il programma a pagina 318 per la versione Spectrum delle linee da 1300 a 1370. Il ciclo previsto nel programma principale (linee da 3750 a 2780) funziona correttamente sullo Spectrum, ma potrebbe causare qualche inconveniente, dovuto alla ripetizione automatica dei tasti. Il manuale dello Spectrum consiglia, nell'uso della INKEY$, l'inclusione della linea:

```
3755 IF INKEY$ <> "" THEN GOTO 3755
```

La funzione VAL(A$) esiste sullo Spectrum, ma i programmi si bloccano qualora il primo carattere di A$ non sia numerico. Ciò si può evitare con:

```
3790 LET SCEL = CODE A$-48
```

Attenzione, però, questa soluzione funziona se A$ è costituita da un solo carattere (come nel nostro programma). Lo Spectrum non dispone dell'istruzione ON...GOSUB, ma permette l'uso di un'espressione numerica, quale argomento per la GOSUB. Quindi, la linea 4010 diventa:

4010 GOSUB (290 + SCEL*20)

Il comando RENUM non è disponibile.

MERGE

Il comando non è disponibile sull'Oric-1, sul Commodore 64, sul Vic-20, sul Lynx, sul Dragon 32 né sul Micro della BBC. Il BASIC del Lynx prevede un comando APPEND, per accodare, in memoria, un programma ad un altro. L'unica condizione è che il primo numero di linea del secondo programma sia maggiore dell'ultimo nel programma già presente in memoria.

RENUMBER

Il comando non è disponibile sullo Spectrum, sui Commodore, né sull'Oric-1. Il numero di linea può esser modificato ricorrendo all'impiego del comando EDIT su ciascuna linea.

INKEY$

Vedere "Basic & Basic" a pagina 175

OPEN CLOSE

Vedere "Basic & Basic" a pagina 319

PRINT CHR$(12)

Si sostituisca con CLS, eccezion fatta per il Commodore 64 ed il Vic-20, nei quali si impiega PRINT "(tasto SHIFT + tasto CLR)": questa sequenza fa apparire sullo schermo un simbolo a cuore o una S maiuscola.

ON... GOSUB

Nella linea 4010, alla ON...GOSUB sono assegnate delle linee di programma inesistenti per il momento, ma che verranno in seguito utilizzate (310, 330, 350, ecc.). Se, in fase di verifica di massima, si vuol evitare un blocco del programma si inseriscano le opportune linee, contenenti una semplice RETURN:

```
310 RETURN
330 RETURN
```

... ecc.

STR$

Il Lynx non possiede questa funzione, quindi occorre sostituire la linea 9080 con:

```
9075 N = DIMENSIONE
9077 GOSUB 9500
9080 CAMIND$(DIMENSIONE) = N$
```

ed inserire la subroutine:

```
9500 REM Conversione di N in N$ (N numero
     intero)
9510 LET N$ = " "
9520 IF N < 0 THEN LET N$ = ""
9530 N = ABS(N)
9540 X = 10
9550 I = 1
9560 FOR K = 1 TO 8
9570 I = I*X
9580 R = K
9590 IF I > N THEN K = 8
9600 NEXT K
9610 FOR K = 1 TO R
9620 I = I/X
9630 Z = INT(N/I)
9640 LET N = N-Z*I
9650 N$ = N$ + CHR$(48 + INT(Z))
9660 NEXT K
9670 RETURN
```

```
10 REM 'MAINPG'
20 REM * INITIL *
30 GOSUB 1000
40 REM * GREETS *
50 GOSUB 3000
60 REM * CHOOSE *
70 GOSUB 3500
80 REM * EXECUT *
90 GOSUB 4000
100 END
1000 REM SUBROUTINE * INIT *
1010 GOSUB 1100: REM SUBROUTINE * CREMAT * (CREAZ. MATRICI)
1020 GOSUB 1300: REM SUBROUTINE * RDINFL * (LETTURA FILE)
1030 GOSUB 1320: REM SUBROUTINE * SETFLG * (ASSEGNA FLAG)
1040 REM
1050 REM
1060 REM
1070 REM
1080 REM
1090 RETURN
1100 REM SUBROUTINE * CREMAT * (CREAZIONE MATRICI)
1110 DIM CAMNOM$(50)
1120 DIM CAMMOD$(50)
1130 DIM CAMVIA$(50)
1140 DIM CAMLOC$(50)
1150 DIM CAMNAZ$(50)
1160 DIM CAMTEL$(50)
1170 DIM CAMIND$(50)
1180 REM
1190 REM
1200 REM
1210 LET DIMENSIONE = 0
1220 LET RMOD = 0
1230 LET SVED = 0
1240 LET CORR = 0
1250 REM
1260 REM
1270 REM
1280 REM
1290 RETURN
1300 REM SUBROUTINE * RDINFL * (VEDERE 'BASIC & BASIC')
1310 ON ERROR GOTO 1370
1320 OPEN "I", #1,"RUBR.DAT"
1330 FOR L = 1 TO 50
1340 INPUT #1 CAMNOM$(L),CAMVIA$(L),CAMLOC$(L),CAMNAZ$(L),CAMTEL$(L)
1350 NEXT L
1360 CLOSE #1
1370 RETURN
1380 REM SUBROUTINE * SETFLG * (PER IL MOMENTO VUOTA)
1390 RETURN
3000 REM SUBROUTINE * GREETS *
3010 PRINT
3020 PRINT
3030 PRINT
3040 PRINT
3050 PRINT TAB (12); "RUBRICA COMPUTERIZZATA"
3060 PRINT TAB (9); "PRESENTATA NEL CORSO"
3070 PRINT TAB (12); "IL MIO COMPUTER"
3080 PRINT
3090 PRINT TAB(5);"PREMERE IL TASTO 'SPAZIO' PER CONTINUARE"
3100 FOR L = 1 TO 1
3110 IF INKEY$ () " " THEN L = 0
3120 NEXT L
3130 PRINT CHR$(12) : REM PULIZIA DELLO SCHERMO
3140 RETURN
3500 REM SUBROUTINE * CHOOSE *
3510 REM
3520 REM 'CHMENU'
3530 PRINT CHR$(12)
3540 PRINT TAB(12); "MENU DELLE OPZIONI"
3550 PRINT
3560 PRINT
3570 PRINT
3580 PRINT "1. TROVARE UN RECORD (USANDO IL NOME)"
3590 PRINT "2. TROVARE UN RECORD (USANDO PARTE DEL NOME)"
3600 PRINT "3. TROVARE RECORD (USANDO LA LOCALITA')"
3610 PRINT "4. TROVARE UN RECORD (USANDO LE INIZIALI)"
3620 PRINT "5. ELENCO DI TUTTI I RECORD"
3630 PRINT "6. IMMISSIONE DI UN RECORD"
3640 PRINT "7. VARIAZIONE DI UN RECORD"
3650 PRINT "8. ANNULLAMENTO DI UN RECORD"
3660 PRINT "9. FINE DEL LAVORO"
3670 PRINT
3680 PRINT
3690 REM * INCHOI *
3700 REM
3710 LET L = 0
3720 LET I = 0
3730 FOR L = 1 TO 1
3740 PRINT "QUALE OPZIONE (1 - 9)"
3750 FOR I = 1 TO 1
3760 LET A$ = INKEY$
3770 IF A$ = "" THEN I = 0
3780 NEXT I
3790 LET SCEL = VAL(A$)
3800 IF SCEL < 1 THEN L = 0
3810 IF SCEL > 9 THEN L = 0
3820 NEXT L
3830 RETURN
4000 REM SUBROUTINE * EXECUT * (VEDERE "BASIC & BASIC")
4010 ON SCEL GOSUB 310,330,350,370,390,410,430,450,470
4020 RETURN
9000 REM SUBROUTINE * AGGREC *
9010 PRINT CHR$(12)
9020 INPUT "NOME ";CAMNOM$(DIMENSIONE)
9030 INPUT "INDIRIZZO ";CAMVIA$(DIMENSIONE)
9040 INPUT "CITTA' ";CAMLOC$(DIMENSIONE)
9050 INPUT "NAZIONE ";CAMNAZ$(DIMENSIONE)
9060 INPUT "N. TELEFONICO ";CAMTEL$(DIMENSIONE)
9070 LET RMOD = 1
9080 LET CAMIND$(DIMENSIONE) = STR$(DIMENSIONE)
9090 GOSUB * MODNOM *
9100 RETURN
```

Qualità musicali...

Effetti musicali sul Micro BBC modello B

Il Micro BBC è dotato di tre oscillatori indipendenti ad onda quadra, 8 tipi di rumore, 4 ADSR indipendenti e di inviluppi di tonalità. Ciò significa che le sequenze musicali possono essere costruite sulla base di tre voci (polifonia). Particolari comandi permettono di suonare le note di un accordo in perfetta sincronia.

Creazione del suono

È possibile produrre suoni, nella loro forma più semplice, usando il comando SOUND:

```
SOUND C,V,P,D
```

In questo caso:

C = Numero di canale o oscillatore (0/3)
V = Volume (0/—15)
P = Tonalità del suono (0/—255)
D = Durata o lunghezza del suono (1/—255)

Si può selezionare uno qualunque dei tre oscillatori (C = 1 , 2 o 3) per suonare delle note, mentre il numero 0 seleziona il generatore di rumore. Il volume può assumere valori compresi tra 0 (spento) e —15 (volume massimo). La tonalità viene definita in intervalli di mezzo semitono, compresi tra 0 (la# a 116,5Hz) e 255 (re a 6498,64Hz) offrendo un'estensione di cinque ottave e mezzo. Il do centrale corrisponde al valore 28. Selezionando il canale 0, la scelta spazia tra otto tipi di rumore, identificati dai seguenti numeri:

Numero	Tipo di rumore
0	"Tremolo" ad alta tonalità
1	"Tremolo" a media tonalità
2	"Tremolo" a bassa tonalità
3	"Tremolo". La tonalità cambia al variare della tonalità del canale 1
4	Tonalità alta
5	Tonalità media
6	Tonalità bassa
7	La tonalità cambia al variare della tonalità del canale 1

Infine, la durata di un suono viene impostata, in intervalli di un ventesimo di secondo, con valori compresi tra 0 e 255; ne consegue che la durata massima di un suono può essere di 12.75 secondi. Il comando:

```
SOUND 1, —7,46,10
```

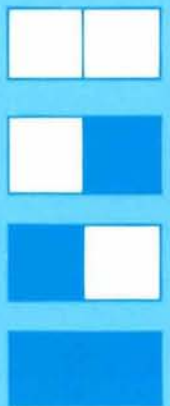
genera un la (immediatamente superiore al do centrale) sul canale 1, ad un medio volume (—7) e per una durata di mezzo secondo. Se il computer riceve un secondo comando SOUND prima di aver portato a termine il primo, la nota da eseguire verrà posta in linea d'attesa sul canale. Oltre alle semplici funzioni ora descritte, le possibilità offerte dal comando SOUND possono essere ampliate cambiando il suo formato nel seguente:

```
SOUND &HSFC,V,P,D
```

In questo caso, il simbolo "&" comunica al

Lampi di luce

La grafica sul Commodore 64



Il Commodore 64 offre possibilità grafiche molto interessanti: il limite principale, comune anche al VIC-20, è che il gruppo di comandi BASIC forniti come standard è assai limitato. Intervenendo direttamente sulla memoria, con i comandi POKE e PEEK, è possibile sfruttare tutte le caratteristiche della macchina, ma alcune delle procedure necessarie possono facilmente causare una certa confusione. Come per il VIC-20, esistono numerosi "pacchetti" di software, prodotti da varie case, che permettono di programmare con facilità "sprite" e caratteri grafici definiti dallo stesso programmatore. La Commodore inoltre produce una propria cartuccia di programmi chiamata "Simon's BASIC".

Il Commodore 64 possiede caratteri maiuscoli e minuscoli standard (utilizzabili anche in video "inverso"), oltre a speciali caratteri grafici, usati per la prima volta sul PET ed ora anche sul VIC-20. Lo schermo possiede 25 righe e 40 colonne ed i caratteri vengono visualizzati nel colore desiderato mediante il comando PRINT, oppure inserendo, con il comando POKE, i relativi codici nelle memorie video e colore descritte minuziosamente sul manuale d'uso. I caratteri grafici speciali del Commodore 64, più di 60, permettono di creare discrete immagini anche in bassa risoluzione. Inoltre, ciascun carattere speciale può essere visualizzato anche in forma inversa, dando così all'utente ben più di 120 possibilità di scelta nella progettazione di un'immagine.
Se i caratteri grafici standard non soddisfano le esigenze del programmatore, è possibile definirne di nuovi. Sfortunatamente, questa operazione può rivelarsi abbastanza complicata. Si dovrà innanzi tutto copiare dalla memoria ROM nella RAM tutti i caratteri normali di cui si avrà bisogno e quindi, per mezzo del comando POKE, inserire i codici necessari nelle locazioni a disposizione per i nuovi caratteri.
Ogni carattere è composto da una matrice di 8×8 pixel, rappresentati in forma binaria da gruppi di 1 e 0. Di solito il numero 1 simboleg-

computer che le quattro lettere HSFC devono essere considerate un numero esadecimale a quattro cifre. Comunque, per capire come usare questo nuovo formato, è necessario analizzare la funzione svolta da ciascuna delle ultime tre cifre (la prima, ossia la C, infatti, indica semplicemente il numero di canale già descritto).
H può assumere valore 0 (funzione disabilitata) o 1 (funzione abilitata). In tal caso la nota, precedentemente posta sullo stesso canale, viene suonata per intero, altrimenti, se questa nota è stata programmata per una lunga fase di emissione (va, cioè, ad estinguersi lentamente) il computer potrebbe erroneamente ritenerla terminata e dare il via alla nota successiva.
La H selezionata ad 1 costringe il computer ad attendere la fine della nota precedente. Se il parametro H è presente (diverso da zero), la restante parte del comando SOUND & viene ignorata.
La cifra indicata con S permette al programmatore di specificare il numero di note che devono essere suonate contemporaneamente sui diversi canali, creando un accordo. S = 0 riporta il comando al suo stato normale. S, selezionato ad 1, permette la simultanea emissione di note da un altro canale. Se S è selezionato a 2, le note su entrambi gli altri canali verranno suonate contemporaneamente. In realtà, quando il computer interpreta un valore di S, esso conserva temporaneamente la nota cui S si riferisce, finché, avendo individuato le altre una o due note indicate dallo stesso valore di S, ma relative agli altri canali, non suona l'intero accordo all'unisono.
F può assumere i valori 0 ed 1. 0 non ha alcun effetto, mentre 1 induce il computer a scartare tutte le note che si trovano sul canale, in attesa di essere suonate, ad interrompere la nota in corso di emissione ed a suonare subito la nota contenuta nel comando in esecuzione.
Il comando SOUND & può essere meglio chiarito con un esempio. Il seguente programma suona la prima riga musicale della canzone *Happy birthday to you* in tonalità di fa# (sopra il do centrale):

fa# fa# sol# fa# si la#

È compreso anche un accordo maggiore (triade) per il la# finale, composto da un re, un fa ed un la#:

```
10 SOUND 1,-7,40,20:REM ★PRIMO FA#★
20 SOUND 1,-7,40,10:REM ★SECONDO FA# BREVE★
30 FOR I=1 TO 3: READ N
40 SOUND 1,-7, N,20:NEXT I:REM ★SOL# FA# SI★
50 SOUND &201,-7,32,30:REM ★RE★
60 SOUND &202,-7,38,30:REM ★FA★
70 SOUND &203,-7,48,30:REM ★LA#★
80 DATA 44,40,50
90 END
```

Le possibilità offerte dal comando ENVELOPE verranno trattate in seguito.

gia un pixel "acceso" (nel colore prescelto per i contorni dell'immagine), mentre lo 0 identifica un pixel "spento", ossia dello stesso colore dello sfondo. Nel Commodore 64 esiste la possibilità, in "modo multicolor", di definire ben 4 colori all'interno di ciascuna matrice che forma un carattere. In questo caso, vengono usati due bit per definire il colore di ciascun pixel.
Una coppia di bit può presentare quattro diversi valori binari, corrispondenti ad altrettante combinazioni di colori. Ciascuna cella di memoria video può contenere caratteri normali o caratteri multicolor ma, in questo secondo caso, la gamma dei colori disponibili si ridurrà da 16 a 8. Una limitazione, imposta dal modo di funzionamento appena descritto, è che i due bit multicolor devono essere gli stessi per tutti i caratteri sul video; inoltre, il grado di risoluzione orizzontale risulta essere diminuito di un fattore 2.
Il BASIC della Commodore non possiede comandi per la grafica ad alta risoluzione. È comunque possibile programmare immagini ad alta risoluzione usando la tecnica nota come "bit-mapping". L'immagine video del Commodore 64 è composta da 64000 pixel. Il "bit-mapping" permette di intervenire individualmente su ciascun pixel "accendendolo" (on) o "spegnendolo" (off). Questo procedimento risulta abbastanza complicato per i meno esperti e l'uso del BASIC, in questo caso, rende estremamente lenta la costruzione di un'immagine ad alta risoluzione. In definitiva, chi desidera utilizzare la grafica ad alta risoluzione ha due alternative: acquistare la cartuccia "Simon's BASIC" della Commodore oppure imparare a programmare in linguaggio macchina. Il modo standard di alta risoluzione del Commodore 64 suddivide lo schermo in 200 linee di 320 pixel ciascuna. Il "modo multicolor" è utilizzabile anche in alta risoluzione e permette di assegnare quattro differenti colori a ciascun blocco di 8×8 pixel.
Il seguente breve programma usa la gamma di caratteri fornita dalla Commodore per riprodurre un ambiente tipico di un supermercato. Più avanti, sempre nella sezione "Suoni e Luci", esamineremo più approfonditamente gli sprite ed il "Simon's BASIC".

Primo Passo
Quest'immagine statica di un supermercato è stata creata usando grafica a bassa risoluzione. In seguito saremo in grado di aggiungere la figura di un compratore che spinge un carrello, usando grafica a sprite.

IAN McKINNELL

```
3000 REM ** SUPERMERCATO **
3010 PRINT"[graphic characters]"
3020 POKE53280,6:POKE53281,12
3030 PRINT"[graphic characters]"
3040 PRINT"[graphic characters]"
3050 PRINT"[graphic characters]"
3060 PRINT"[graphic characters]"
3070 PRINT"[graphic characters]"
3080 PRINT"[graphic characters]"
3090 PRINT"[graphic characters]"
3100 PRINT"[graphic characters]"
3110 PRINT"[graphic characters]"
3120 PRINT"[graphic characters]"
3130 PRINT"[graphic characters]"
3140 PRINT"[graphic characters]"
3150 PRINT"
3160 PRINT"[graphic characters]"
3170 PRINT"
3180 PRINT"[graphic characters]"
3190 PRINT"
3200 PRINT"[graphic characters]
3210 PRINT"
3220 PRINT"[graphic characters]"
3230 PRINT"
3240 PRINT"[graphic characters]"
3250 FORI=1063TO2023STEP40
3260 POKEI,160:POKE54272+I,6:NEXT
3270 GOTO3270
```

Leonardo Torres

Il poliedrico ingegnere spagnolo e la notazione a virgola fluttuante

Leonardo Torres y Quevedo, lo scienziato che per primo applicò ai computer la notazione aritmetica a virgola fluttuante, nacque nella città spagnola di Santa Cruz il 28 dicembre 1852. Studiò all'Istituto di Bilbao e all'Università di Ingegneria di Madrid. Successivamente, intraprese la carriera di ingegnere ed inventore.

Le sue capacità inventive raggiunsero il culmine durante la seconda parte della sua vita. Progettò il ponte di transito sul fiume Niagara, la funivia ancor oggi in funzione presso le Cascate del Niagara ed un dirigibile semirigido, fabbricato durante la I^a Guerra Mondiale. Ma, fondamentalmente, Torres fu un figlio dei suoi tempi ed i suoi maggiori interessi si rivolsero agli apparecchi elettromeccanici. Nel 1906, alla presenza del Re di Spagna, fece una dimostrazione con un modellino di nave radiocomandato nel porto di Bilbao; nel 1911 progettò la prima scacchiera automatica: il movimento dei pezzi era ottenuto grazie a delle elettrocalamite, poste sotto la scacchiera, ed era programmato per giocare semplici partite contro un avversario umano. L'interesse di Torres per l'automazione derivava dalle esperienze da lui fatte nella progettazione di catene di montaggio per varie industrie europee del primo '900. Per tutta la sua vita cercò di stabilire quali problemi necessitassero dell'intervento umano e quali invece potevano essere risolti mediante operazioni automatiche.

Nel 1914 pubblicò un articolo per dimostrare come fosse possibile fabbricare la Macchina Analitica di Babbage (vedere a pag. 220) impiegando tecniche elettromeccaniche e, per la prima volta, ipotizzò l'uso della notazione aritmetica a virgola fluttuante nei futuri computer. Nel 1920 progettò un calcolatore elettromeccanico che utilizzava una macchina da scrivere modificata per l'immissione di dati numerici e per la stampa automatica dei risultati. La macchina da scrivere fu collegata al calcolatore mediante cavi telefonici e Torres prefigurò la possibilità di collegare più terminali ad un calcolatore centrale (o unità centrale di elaborazione).

Egli ricevette varie onorificenze dall'Accademia Francese delle Scienze ed, in seguito, divenne presidente dell'Accademia delle Scienze del proprio paese. Torres morì a Madrid il 18 dicembre 1936.

Scienziato poliedrico

Come molti dei suoi contemporanei, Torres fu un vero scienziato multidisciplinare. I suoi interessi spaziarono dalla progettazione e costruzione di apparecchi meccanici, come il dirigibile mostrato in figura, alle macchine calcolatrici elettromagnetiche fino alla matematica pura



per concessione della ROYAL AERONAUTICAL SOCIETY

NOTAZIONE ARITMETICA A VIRGOLA FLUTTUANTE

In Gran Bretagna i registratori di cassa visualizzano i totali in sterline e "pence" (ad esempio, 12.45 sterline) ed in apparecchi di questo tipo c'è bisogno solo di 2 cifre dopo il punto decimale. Ma in un computer è spesso necessaria una precisione molto maggiore ed il numero delle cifre decimali deve poter variare o "fluttuare" a seconda delle necessità del problema in esame. Questo procedimento prende il nome di "notazione aritmetica a virgola fluttuante".

Un numero può essere scritto in diversi modi. Ad esempio, 0,8752 metri può essere scritto come 875,2 millimetri o 0,8752 x 1000 millimetri oppure, più semplicemente, $0{,}8752 \times 10^3$. Quest'ultimo si è dimostrato il metodo di codificazione più adatto per un computer. Se, infatti, sono disponibili soltanto sei locazioni di memoria per rappresentare un numero (e per maggior chiarezza viene usato il sistema decimale invece del binario), allora il massimo numero di cifre rappresentabili, tornando all'esempio precedente, sarà 875203. Mentre le prime quattro cifre a sinistra vengono chiamate "mantissa", le ultime 2 a destra prendono il nome di "indice" e rappresentano la potenza del 10 (in questo caso 3). Facciamo un altro esempio: il numero 418302 equivale a $0{,}4183 \times 10^2$ o a 41,83. Mantissa ed indice vengono di solito "normalizzati" per eliminare gli zeri che eventualmente precedono la mantissa. Ad esempio il numero 41,83 può presentarsi come 004104, ma deve essere normalizzato in 418302; in questa forma conterrà nella mantissa un maggior numero di cifre significative.

Il formato indice/mantissa della notazione aritmetica a virgola fluttuante permette di rappresentare un vastissimo arco di numeri. Con il computer visto precedentemente, che dispone di due sole locazioni per l'indice, sarà possibile elaborare numeri grandi quanto $0{,}9999 \times 10^{99}$, oppure ci si potrà imbattere in un numero tanto piccolo da presentare 98 zeri dopo la virgola, seguiti da un qualsiasi numero diverso da zero.

Comunque, la precisione di questo sistema rimane circoscritta alle cifre che si trovano nella mantissa. Per questa ragione alcuni numeri potranno venir rappresentati solo approssimativamente e sarà necessario prestare grande attenzione e cura nelle tecniche di programmazione aritmetica, onde bloccare gli errori al loro primo apparire. Questa è la ragione per cui l'espressione (1/3) * 3 su alcuni computer dà come risultato 0,9999999 e non un più esatto 1.

Inventiamo un gioco

Scrivere il programma di un gioco non è poi molto difficile



per concessione della MILTON BRADLEY LTD

IAN McKINNELL

La mano invisibile

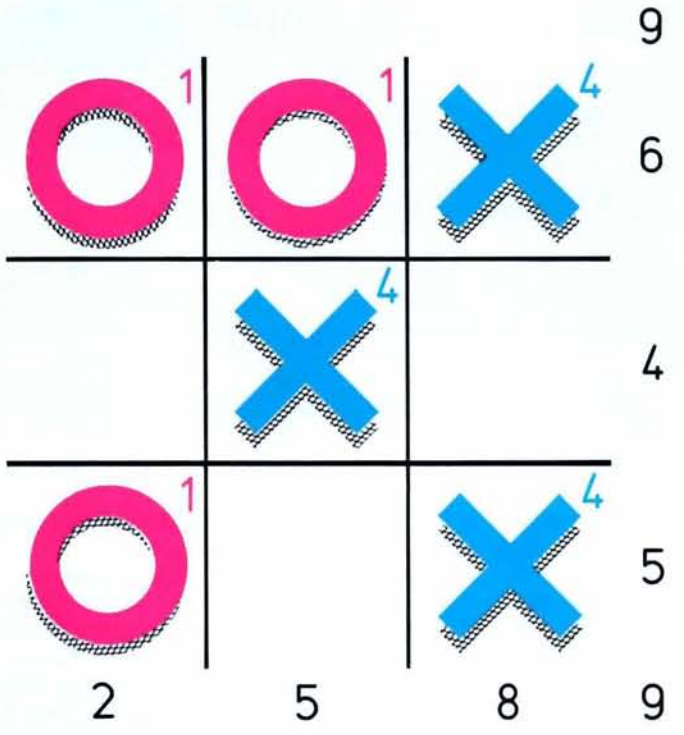
Le macchine elettroniche per il gioco degli scacchi impiegano più o meno gli stessi componenti di un computer: una CPU, una RAM ed una ROM contenente i programmi. La differenza principale sta nell'input e nell'output dei dati. Il Phantom, l'apparecchio illustrato qui di fianco, si avvale di elettromagneti e di servomeccanismi per spostare automaticamente i pezzi sulla scacchiera

La stesura di un programma per il gioco degli scacchi è un sogno comune a moltissimi programmatori. Questa ambizione non proviene dal fatto che simili programmi non esistano: al contrario, è in vendita tutta una gamma di programmi per computer (o di apparecchi specifici) che permettono di giocare agli scacchi a qualsiasi livello. La ragione principale, forse, risiede nella sfida intellettuale rappresentata dalla realizzazione di un simile programma. Inoltre, molti considerano una simile applicazione del computer un passo avanti verso la creazione di una macchina "intelligente". Se, da un lato, la stesura completa di un programma per gli scacchi è senza dubbio molto complessa ed impegnativa, i principi di base possono essere illustrati mediante la spiegazione della logica impiegata per realizzare alcuni giochi più semplici. I giochi, cui ci riferiamo, non sono quelli diffusi nelle sale giochi, che sfruttano tecniche di programmazione e capacità creative di natura ben diversa: sono giochi più semplici, basati principalmente sulla logica. Come primo esempio, consideriamo il comunissimo gioco della *Morra cinese*.

Le regole sono veramente semplici: i due giocatori devono scegliere, mentalmente, uno fra tre oggetti (un sasso, un paio di forbici o un foglio di carta), poi, contemporaneamente, palesano la propria decisione con una particolare posizione della mano. La vittoria viene stabilita in base alle varie combinazioni possibili: il foglio vince sul sasso (in quanto lo avvolge), ma a sua volta è vinto dalle forbici (che lo tagliano). Le forbici, infine, vengono vinte dal sasso (che le spunta).

Per chi abbia seguito la sezione "Programmazione Basic", dovrebbe essere abbastanza semplice scrivere un breve programma per questo gioco: con la funzione RND il computer compie la propria scelta fra le tre possibili; il giocatore immette la propria scelta e, quindi, viene visualizzato il risultato. Il computer terrà conto del punteggio. Se la funzione RND è realmente in grado di generare numeri casuali, i risultati di una lunga serie di lanci, indipendentemente dalla strategia adottata

dal giocatore, si avvicinerà ad un valore medio. Possiamo tentare di migliorare il criterio di scelta del computer, affinché riesca a vincere un maggior numero di volte.
Quando abbiamo trattato la generazione di numeri casuali (vedere a pag. 209), è stato detto come sia impossibile, sia per il computer che per l'uomo, generare una serie di numeri veramente casuali, benché la macchina possa dare buone approssimazioni. A lungo andare, il giocatore umano tende a manifestare una propria preferenza verso uno dei tre oggetti. Potremmo usare una matrice numerica, nel programma, per tener conto delle scelte fatte dal giocatore umano: il computer sarebbe quindi in grado di individuare l'oggetto giocato più frequentemente dal suo avversario, e regolarsi di conseguenza.
Con questo metodo vi sono però tre inconvenienti. In primo luogo, se il computer inizia a giocare ripetutamente un oggetto, l'avversario ne trarrà subito profitto: occorre perciò mantenere una certa casualità nei tiri.



POSIZIONE VINCENTE

La "valutazione della posizione" è un processo fondamentale in qualsiasi gioco da tavolo, anche nel semplice gioco qui illustrato. In questo caso, il tavolo da gioco è rappresentabile mediante una matrice di 3 x 3 elementi, nella quale ciascun elemento avrà valore 1, se la corrispondente posizione viene occupata con un cerchietto (il giocatore) oppure avrà valore 4 se la posizione è occupata da una crocetta (il computer). Il totale ottenuto dalla somma di ciascuna riga, colonna o diagonale serve per determinare i risultati del gioco: 12 significa che il computer ha piazzato 3 crocette in fila ed ha vinto; 3 indica la vittoria del giocatore; un totale di 8, invece, significa che due crocette sono già piazzate, ed esiste una possibilità di vittoria per il computer. Spesso un'opportuna scelta di valori (in questo caso 1 e 4) consente di semplificare enormemente la stesura del programma per il computer

LIZ DIXON

In secondo luogo, il giocatore tende, a lungo andare, ad alternare la propria preferenza tra un oggetto e l'altro. Conviene memorizzare, quindi, solo una parte delle scelte: le ultime 20, per esempio. Ciò richiede l'uso di una matrice numerica di 20 × 3 elementi, e di una porzione di programma, relativamente complessa, per operare la scelta più proficua per il computer.
Tuttavia, la principale limitazione di questo sistema proviene dal terzo inconveniente: il giocatore può facilmente comprendere il meccanismo di scelta impiegato dal computer e riuscire a prevenirne le mosse. Ad esempio, il giocatore potrebbe giocare più volte lo stesso oggetto, per poi cambiare improvvisamente. Occorre, perciò, un altro tipo di algoritmo, che permetta di superare questi ostacoli. Può essere istruttivo, comunque, scrivere dei piccoli programmi basati sia sulla sola generazione casuale, sia su di una generazione "pilotata" delle scelte.
Poiché gli esseri umani non sono capaci di prendere decisioni totalmente casuali o irrazionali, ne consegue che ogni scelta deriva, in qualche modo, da quelle precedenti. Se il computer riuscisse ad approssimare, con una funzione, il comportamento del giocatore, allora si assicurerebbe una larga fetta di vincite. Naturalmente, questa funzione deve essere adattabile sia durante il gioco con un solo giocatore, sia ai criteri impiegati da giocatori diversi. I programmi in grado di operare questa specie di adattamento automatico si chiamano "euristici".
Un programma euristico riesce a determinare, con una buona approssimazione, i cambiamenti nella strategia del proprio avversario ed, in base a questi, a modificare l'algoritmo di scelta. Tenendo nota, ad esempio, delle ultime 50 mosse, esse vengono esaminate di continuo, alla ricerca di ciò che, in termini statistici, si chiama "correlazione". Ciò implica una complessa serie di confronti: tra tutte le mosse e quella precedente, o tra le ultime cinque, e via dicendo. Inoltre, il computer agisce in modo simile nei confronti delle proprie mosse passate. Vediamo in cosa consista la correlazione tra una mossa e quella precedente. Per semplicità, chiameremo Ele-

mento 1 le Forbici, Elemento 2 la Carta ed Elemento 3 il Sasso. Creiamo una matrice di 3 per 3 elementi, che chiameremo CORR1, in quanto rappresenta la nostra prima verifica di correlazione. Adesso dobbiamo esaminare le ultime 50 mosse: ogni volta che alle Forbici (1) segue il Sasso (3), incrementiamo di 1 il valore contenuto in CORR1(1,3); quando il Sasso (3) è seguito dalla Carta (2), sommiamo 1 a CORR1(3,2), e così via.

Se il giocatore riuscisse ad essere veramente casuale nei suoi lanci, i valori della matrice CORR1 sarebbero abbastanza omogenei, ma ciò non accade quasi mai in realtà. Così, se l'ultima scelta del giocatore è stata Carta, allora il massimo valore nella riga 2 (Carta) di CORR1 può rappresentare, con buone probabilità, il prossimo elemento scelto. Quanto maggiore è la differenza tra i vari elementi della matrice, tanto maggiore è il fattore di correlazione, assieme alla probabilità di indovinare la mossa. Ma spesso questa sola verifica non basta, ed occorre prendere in considerazione la penultima mossa, oppure il rapporto tra le mosse del computer e le "risposte" dell'avversario.

Ovviamente, le cose si complicano, soprattutto se i risultati delle varie verifiche differiscono tra loro: il programma dovrà decidere a quale risultato affidarsi. Nel semplice gioco preso ad esempio, la scelta può esser fatta tenendo conto del fattore di correlazione più alto. Se, per esempio, in base ai valori di CORR1 le previsioni sono: Forbici 51%, Carta 29%, Sasso 20% ed, in base ai valori di CORR2 (risultanti dall'esame della penultima mossa), abbiamo le previsioni: Forbici 24%, Carta 60% e Sasso 16%, allora CORR2 contiene un valore di correlazione più alto (il 60% per Carta). In effetti, spesso i programmi "intelligenti" di gioco sono impostati sull'impiego di numerose subroutine ognuna delle quali produce una serie di risultati, che vengono poi scelti da un'apposita parte del programma principale, che può addirittura giungere ad assegnare un "valore di affidabilità" a ciascuna subroutine.

Se un fattore di correlazione esiste, nelle mosse del giocatore, il computer può persino arrischiare qualche bluff, per sviare l'attenzione dell'avversario. Ciò è particolarmente vero nei giochi d'azzardo.

L'Università di Stato a Buffalo (USA) ha compiuto un interessante esperimento (del quale si è parlato anche nella nota rivista *Scientific American*, Luglio 1978): un certo numero di programmi (tutti dotati di capacità d'apprendimento) per il gioco del poker sono stati fatti giocare tra di loro, per migliaia di smazzate. Il programma vincente è stato l'*Adaptive Evaluator of Opponents* (valutatore dinamico degli avversari), il quale, compiuta un'analisi iniziale delle mani avversarie, modificava questa stima col progredire del gioco. Il programma chiamato *Sells and Buys Images* (venditore/acquirente d'immagini) ha ottenuto scarso successo: la sua tecnica era impostata in gran parte sul bluff, ossia nel "vendere" una propria immagine agli avversari, o nel "comprare" lo stile di gioco avversario. Il programma chiamato *Adaptive Aspiration Level* (traducibile con: livello d'ambizione dinamico), infine, tentava di simulare un atteggiamento ritenuto "umano": adattare le scelte in base al livello di rischio che si è disposti a correre, tenendo conto delle passate esperienze e della situazione attuale.

Non esistono due programmi per il gioco degli scacchi (o altri giochi cosiddetti intelligenti) perfettamente identici tra loro: ognuno differisce nell'impostazione e nella complessità dei criteri adottati per lo stile di gioco. Comunque, anche partendo dalle semplici considerazioni qui esposte, è possibile sviluppare interessanti teorie ed algoritmi da applicare ad un vasto numero di situazioni.

```
5 CLS : REM OPPURE PRINT CHR$(12) -- PULIZIA DELLO SCHERMO
10 DIM C1(3,3),C2(3,3),C3(3,3)
20 CR = 0 : PL = 1 : PP = 1 : P3 = 1
30 FOR L = 1 TO 3
40 IF C1(PL,L) > CR THEN BG=L:CR=C1(PL,L)
50 IF C2(PP,L) > CR THEN BG=L:CR=C2(PP,L)
60 IF C3(P3,L) > CR THEN BG=L:CR=C3(P3,L)
70 NEXT L
80 CT = BG-1
90 IF BG = 1 THEN CT = 3
100 GET PT: IF PT = 0 THEN 100
110 REM NON DISPONENDO DELL'ISTRUZIONE GET
120 REM RICORRERE ALL'IMPIEGO DI INKEY$
130 IF CT=PT-1 THEN CS=CS+1
140 IF CT=PT-2 THEN PS=PS+1
150 IF CT=PT+1 THEN PS=PS+1
160 IF CT=PT+2 THEN CS=CS+1
170 CLS
180 PRINT "LA TUA SCELTA E`: ";PT
190 PRINT "LA MIA SCELTA E`: ";CT
200 PRINT "IL TUO PUNTEGGIO E`: ";PS
210 PRINT "IL MIO PUNTEGGIO E`: ",CS
220 C1(PL,PT)=C1(PL,PT)+1
230 C2(PP,PT)=C2(PP,PT)+1
240 C3(P3,PT)=C3(P3,PT)+1
250 P3 = PP
260 PP = PL
270 PL = PT
280 GOTO 20
```

Il computer impara

Questo programma, basato sul gioco della "Morra Cinese", illustra come un computer possa costruirsi un'"esperienza" durante lo sviluppo del gioco. Il programma seleziona un valore (1, 2 o 3), lo confronta con quello scelto dal giocatore, ed assegna un punteggio. Viene impiegata l'istruzione GET (invece di INPUT), per velocizzare l'esecuzione. A lungo andare (qualche centinaio di "lanci"), il computer sarà in netto vantaggio. Ovviamente, data la semplicità del programma, si riesce facilmente ad ingannare il computer, ma con l'aggiunta di algoritmi più sofisticati la macchina può diventare quasi imbattibile, operando le proprie scelte sulla scorta dell'"esperienza" accumulata

Le mappe di memoria

Com'è distribuito l'utilizzo della memoria in un computer

La CPU di un computer è in grado di gestire una quantità di memoria che dipende dalla sua capacità d'indirizzamento: nei computer dotati di una CPU ad 8 bit (salvo siano previsti più "banchi" di memoria), la quantità massima di memoria è di 64 Kbyte. In ogni caso, parte di questa è di tipo RAM e parte di tipo ROM, comprese le eventuali espansioni ed interfacce. Uno degli aspetti più importanti, nel progetto di un computer, è, appunto, la definizione della "mappa di memoria", ossia di un diagramma dove vengano specificate sia le dimensioni di ciascun segmento di memoria, sia le funzioni svolte in ciascun segmento. Se si usa soltanto il linguaggio BASIC, avremo raramente bisogno di conoscere la mappa di memoria, ma desiderando addentrarsi nella programmazione in linguaggio macchina, allora la sua importanza diviene primaria.

In queste due pagine viene illustrata la configurazione tipica di una mappa di memoria. L'esempio si presta meglio ad un apparecchio dotato di una CPU 6502, ma può adattarsi abbastanza bene anche per apparecchi con una CPU Z80. Alcuni fabbricanti forniscono, nel manuale operativo, una mappa completa della memoria, mentre altri mantengono su di essa il più rigoroso riserbo.

Memoria di sistema

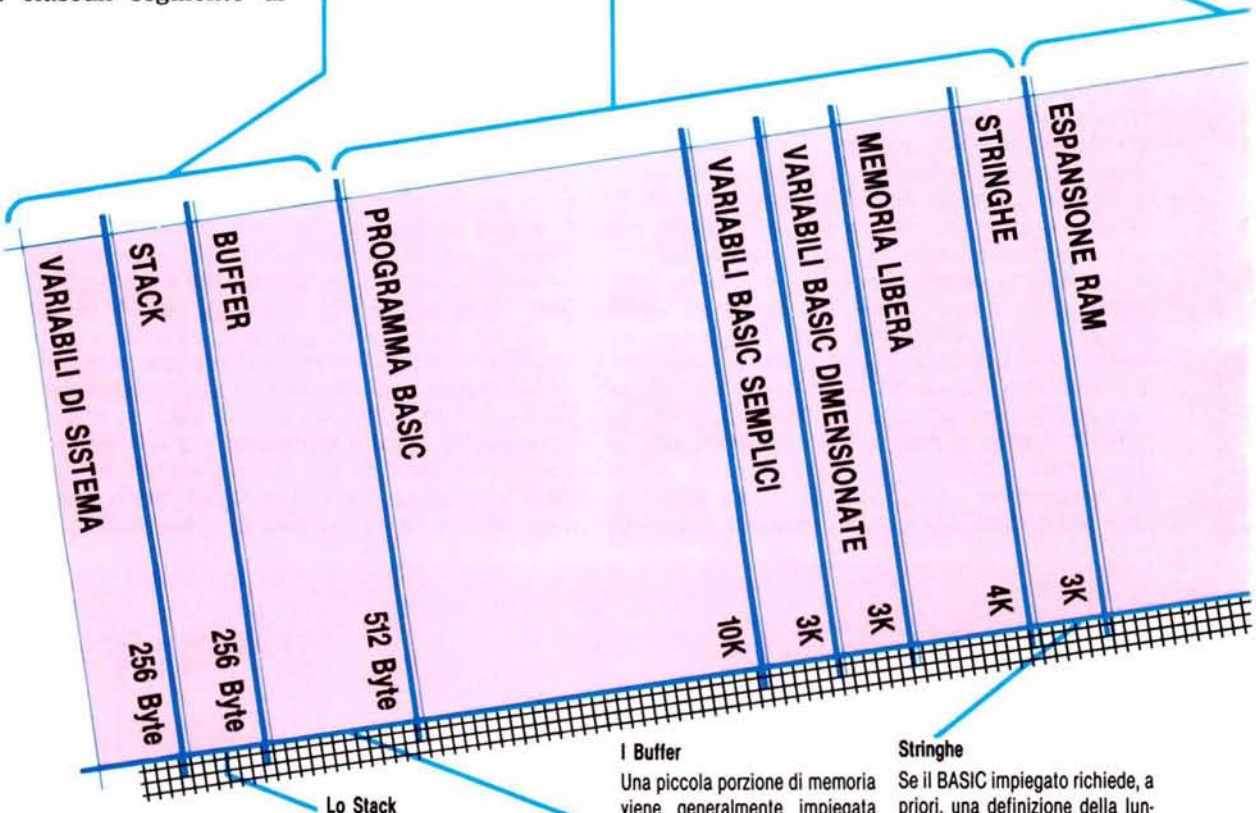
In un computer con 4 Kbyte di RAM, la quantità di memoria disponibile per i programmi dell'utente può anche essere di soli 3 Kbyte: la restante serve ai programmi di gestione interna. In essa vengono memorizzate, per esempio, le variabili usate dal sistema operativo, i puntatori alle informazioni contenute altrove, ecc.

RAM "utente"

L'estensione di questa sezione di memoria determina il grado di complessità dei programmi utilizzabili: maggiore è la RAM utente, migliori saranno i programmi. È quindi un elemento importante nell'acquisto di un computer

Parte vuota

La mappa di memoria deve anche tener conto di eventuali espansioni che si possano aggiungere in un secondo momento. Alcuni sistemi possono indirizzare più di 64 Kbyte, ma ciò avviene, in genere, mediante la gestione di più "banchi"



Lo Stack

Questa parte di memoria è riservata ad uso, come deposito temporaneo, per la CPU, che la organizza come un magazzino LIFO (Last In/First Out, ossia Ultimo entrato/primo ad uscire). Un byte può essere depositato sullo stack (operazione chiamata "push") oppure richiamato da esso (operazione chiamata "pop"). Ad esempio, quando viene eseguita un'istruzione GOSUB, in BASIC, la locazione alla quale il programma deve tornare, una volta eseguita la subroutine, viene appoggiata sullo stack. Inoltre, lo stack viene adoperato per l'esecuzione delle operazioni aritmetiche e nei cicli FOR...NEXT

I Buffer

Una piccola porzione di memoria viene generalmente impiegata come deposito temporaneo per i caratteri immessi da tastiera, cosicché, anche digitandoli velocemente, non ne viene perso neanche uno. Inoltre, viene impiegato un buffer anche per le operazioni di lettura/scrittura su unità a nastri o dischi

Stringhe

Se il BASIC impiegato richiede, a priori, una definizione della lunghezza delle variabili alfanumeriche, lo spazio necessario per memorizzarle viene riservato in un'apposita porzione di RAM. Qualora, invece, le stringhe vengano definite "dinamicamente" (ossia durante l'esecuzione del programma), esse vengono memorizzate in un'altra parte della memoria, le cui dimensioni variano continuamente. Periodicamente, un sottoprogramma del sistema operativo (il "Garbage Collector") provvede a riordinare il contenuto di questa zona di memoria, eliminando i dati non più necessari

RAM di sistema

In alcuni computer, la RAM di sistema non viene considerata come parte di quella utilizzabile dall'utente e viene impiegata per la gestione del video (dove ad ogni byte corrisponde un carattere sullo schermo) e dei colori (dove il valore di ciascun byte indica il colore dello sfondo e del singolo carattere visualizzato). Ai computer dotati di grafica ad alta risoluzione è necessaria un'ampia porzione di RAM e talvolta (ad esempio con sofisticati programmi di giochi) viene occupata quasi tutta la RAM per le funzioni grafiche

Parte vuota

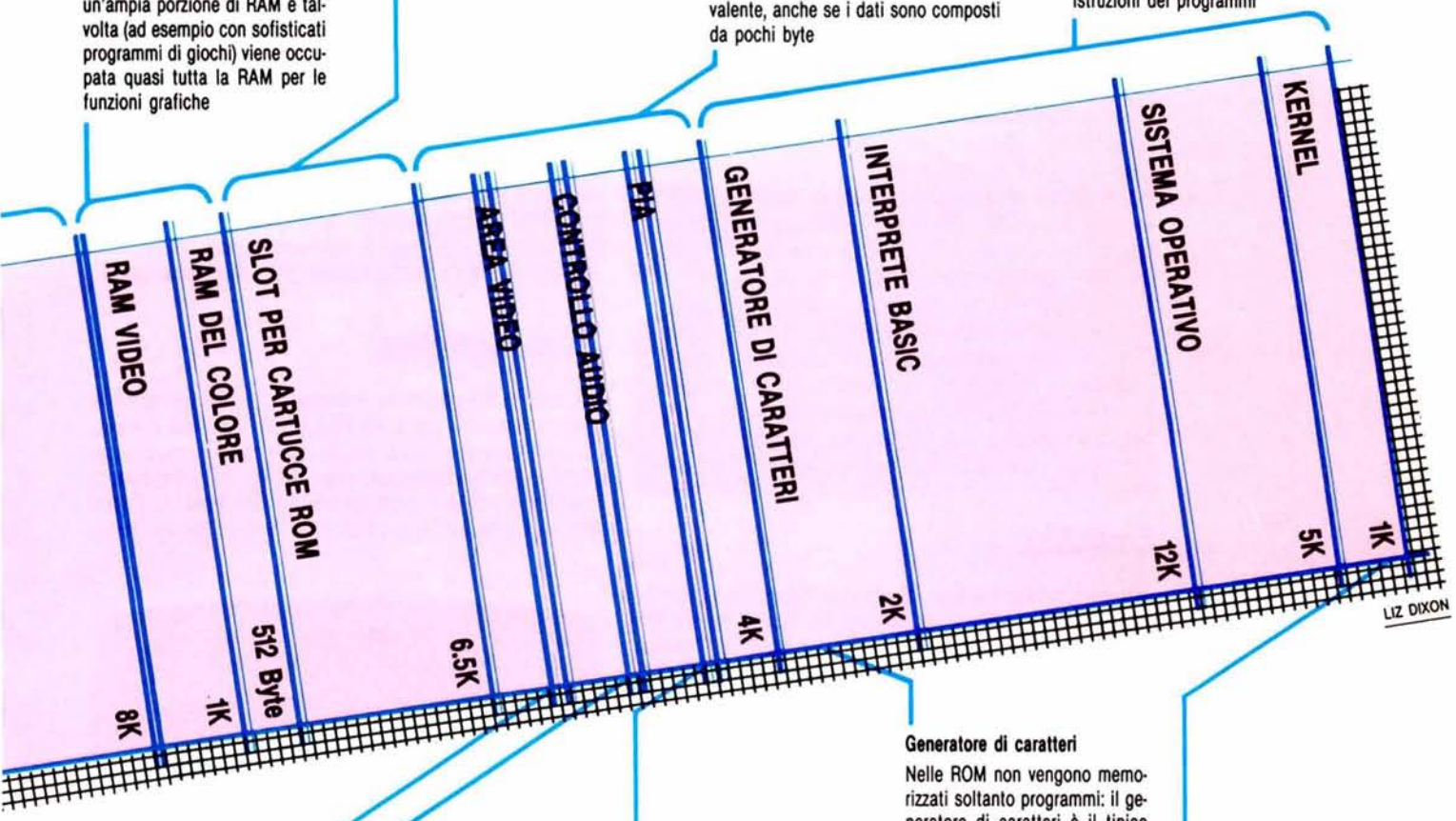
Usando programmi conservati su cartucce, la ROM contenuta in esse viene vista come un'espansione della memoria e quindi la mappa di memoria deve tenerne conto. Alcune macchine consentono l'impiego di linguaggi alternativi al BASIC, contenuti anch'essi in cartucce

Chip di input/output

La CPU è in grado di comunicare soltanto con dispositivi che appaiono come locazioni di memoria nella mappa, cosicché anche le porte di comunicazione (interfacce) o altre forme di ingresso/uscita devono essere incluse nella mappa di memoria. Sono incluse le interfacce per la tastiera, per il registratore o l'unità a dischi, per il video e tutte le interfacce per altre periferiche (stampante, ecc.). Generalmente, la CPU trasferisce i dati in blocchi (anche fino a 4 Kbyte per volta), quindi occorre riservare una porzione di memoria equivalente, anche se i dati sono composti da pochi byte

ROM di sistema

Nelle ROM vengono conservati, in forma permanente, tutti i programmi e le informazioni necessarie al funzionamento. Uno degli elementi fondamentali, generalmente contenuto nelle ROM, è il sistema operativo, che sovrintende a tutte le operazioni principali del computer (scansione della tastiera, lettura e scrittura da e sulla memoria di massa, ecc.). Talvolta, nelle ROM è conservato anche l'interprete BASIC, che traduce, in forma comprensibile alla CPU, le istruzioni dei programmi



Area video

Le capacità grafiche dei computer sono in continua evoluzione e la memoria necessaria per esse aumenta di conseguenza. In questa sezione vengono conservate informazioni relative ai colori da impiegare nelle immagini, alle posizioni, alle forme degli "sprite", ecc.

Controllo audio

Effetti sonori elementari si possono ottenere via software, ma per forme di suono più complesse, polifoniche o sintetizzate, è necessario un circuito di controllo audio che viene collegato ad un amplificatore esterno

I PIA

La sigla PIA sta per Peripheral Interface Adaptors (Adattatori d'interfaccia per periferiche). Questi circuiti vengono impiegati per il trasferimento di dati tra le varie periferiche (tastiera, registratore, joystick, ecc.). Alcuni sofisticati chip, quale il 6522, provvedono anche alla conversione dei dati dalla forma parallela a quella seriale o viceversa e sono, inoltre, dotati di temporizzatori per la sincronizzazione delle trasmissioni

Generatore di caratteri

Nelle ROM non vengono memorizzati soltanto programmi: il generatore di caratteri è il tipico esempio di conservazione di dati in queste memorie. Ogni carattere viene definito in base ad un'opportuna sequenza di bit, utilizzata per la visualizzazione sullo schermo. In certi computer è consentito trasferire il contenuto del generatore di caratteri nella RAM, per la creazione di nuovi simboli o caratteri

Il Kernel

Il "Kernel" (letteralmente: nòcciolo), rappresenta il cuore del sistema operativo. All'accensione dell'apparecchio, la CPU inizia automaticamente l'esecuzione del programma contenuto nel Kernel. Alcune funzioni tipiche di questo programma sono l'inizializzazione delle varie periferiche, il controllo della quantità e della qualità di memoria disponibile (per rilevare eventuali difetti hardware). Se il sistema operativo non è conservato su ROM, ma su dischetto, il Kernel provvede al suo caricamento in memoria

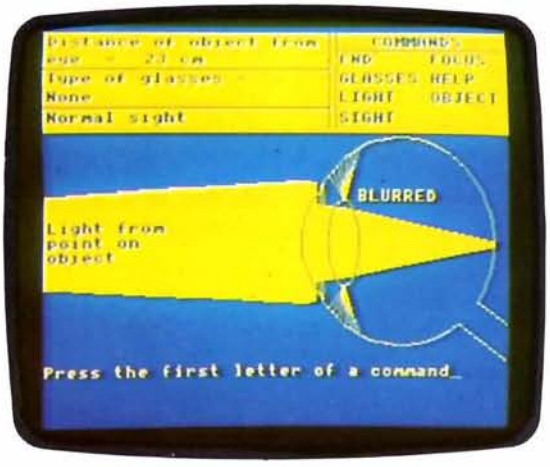
Un po' d'immaginazione

Il computer come strumento didattico: le simulazioni

I programmi di simulazione, come quelli che si trovano nelle sale giochi, nei quali il giocatore pilota un aereo o un'auto da corsa, sono progettati per offrire un'esperienza il più aderente possibile alla realtà. Esiste, tuttavia, una vasta gamma di programmi di simulazione concepiti per la didattica, e non soltanto per il divertimento. Questo software si adatta particolarmente bene alle materie scientifiche, poiché consente di compiere esperimenti altrimenti irrealizzabili a causa della loro pericolosità o del loro costo.
I programmi di simulazione possono essere impiegati anche nell'ambito domestico: esiste, per esempio, un programma chiamato *Car Journey* (viaggio in auto), nel quale l'operatore (generalmente un bambino) può esercitare le proprie nozioni aritmetiche e logiche per "guidare" un'auto. Poiché la maggior parte di questi programmi sfrutta le capacità grafiche del computer e queste variano da apparecchio ad apparecchio, la gamma di computer sui quali farli funzionare è abbastanza limitata. La situazione cambierà, probabilmente, quando i computer saranno maggiormente diffusi nelle scuole.

L'occhio

Questo programma illustra il funzionamento dell'organo visivo e come le varie parti di esso devono aggiustarsi per una corretta messa a fuoco. Viene simulato il percorso che la luce compie dall'oggetto alla retina dell'occhio. L'operatore può intervenire modificando alcuni fattori in gioco: la distanza dall'oggetto, le dimensioni dell'iride ed il fuoco del cristallino, ecc.



Sullo schermo viene visualizzata una sezione dell'occhio, corredata di opportune scritte esplicative. Se le variabili vengono selezionate in modo corretto, appare la scritta A FUOCO, altrimenti il messaggio SFUOCATO.
Appreso il funzionamento di un occhio normale, si possono simulare anche eventuali difetti della vista, quali la miopia, l'astigmatismo ed altri: è allora possibile aggiungere delle lenti correttive determinandone con discreta esattezza anche le diottrie. Benché studiato come sussidio nell'insegnamento della biologia e della fisica, questo programma viene spesso usato anche a scopo dimostrativo per far comprendere agli studenti le possibili applicazioni dei computer. Il programma, il cui uso è estremamente facile, è prodotto dalla Longmans per il Micro BBC.

La mongolfiera

Questo programma, adatto a bambini di età compresa tra gli 8 ed i 12 anni, simula il volo di una mongolfiera: sullo schermo appare un paesaggio schematizzato, sul quale fluttua il "pallone". Sono previste tutte le fasi di volo, dal decollo all'atterraggio. Nella parte inferiore dello schermo è riportata una serie di strumenti: altimetro, pressurimetro, termometro e indicatore del livello di carburante. Esistono soltanto due comandi di controllo: la valvola che permette la fuoriuscita dell'aria (per far scendere il pallone) ed il bruciatore, che, immettendo aria calda nel pallone, ne consente la salita.

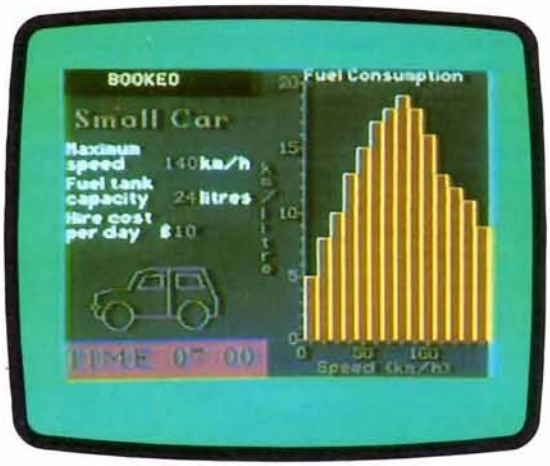


Una semplice sessione di prova insegna come usare i comandi e come reagire nelle particolari situazioni di volo. Quando si è pronti, si può dar corso a voli "indipendenti",

nei quali lo scopo è quello di atterrare in luoghi prestabiliti, segnalati con una X sullo schermo. All'atterraggio viene visualizzato un messaggio del tipo: "aiutare il contadino a salvare le pecore", o cose simili. Se accade di finire il carburante, occorre atterrare per far rifornimento.
Dopo i primi tentativi di volo, in genere costellati da tutta una serie d'incidenti, si riesce ad avere una sufficiente padronanza dei comandi, manovrando opportunamente i getti d'aria calda ed imparando a leggere gli strumenti. Uno degli aspetti interessanti è che le azioni si svolgono in "tempo reale", offrendo effetti molto suggestivi.
Il programma è stato scritto per lo Spectrum dalla Heinemann Educational Software.

Viaggio in auto

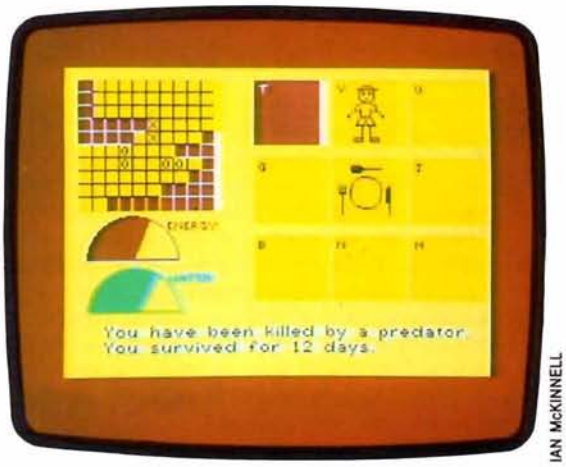
Prodotto anch'esso dalla Heinemann Educational Software per lo Spectrum, il programma simula accuratamente la guida di un piccolo furgone per consegne. Occorre prendere tutta una serie di decisioni (ad esempio, quali incarichi di consegna accettare, a quale velocità viaggiare e quale tipo di veicolo



usare). Per far ciò, il giocatore deve tenere in considerazione diversi fattori: i costi, le distanze, il tempo ed anche il consumo di carburante. Essendo stato sviluppato per la Gran Bretagna, viene visualizzata una carta stradale di quella nazione, nella quale sono riportate 15 città e le autostrade che le collegano. Gli strumenti "di bordo" sono un tachimetro, un contachilometri, un indicatore del livello di benzina ed un orologio.
Il primo compito è quello di decidere da quale città iniziare il giro, successivamente si deciderà quale incarico accettare tra quelli proposti (circa una dozzina). Per esempio, un incarico potrebbe esser quello di ritirare un carico di diamanti nella città di Bristol alle 12 in punto e consegnarlo al porto di Dover prima delle 18 dello stesso giorno. Per far ciò occorre noleggiare un veicolo, guidare fino a Bristol, prendere in consegna il carico e recarsi fino a Dover. In caso di successo, viene corrisposto un compenso (ahimè, fittizio!) di 400 sterline e, se la consegna avviene prima del previsto, addirittura un premio di 10 sterline. Il denaro guadagnato, però, viene speso per eventuali pernottamenti, riparazioni alla macchina e, per i guidatori più spericolati, per pagare le multe. Inoltre, se l'incarico non viene portato a termine, c'è una multa di ben 100 sterline.
Oltre ad apprendere elementari nozioni di trasporto, il programma stimola le capacità decisionali e logiche del giocatore ed, in senso lato, insegna qualche elemento di economia.

Sopravvivenza

Questo è un programma veramente particolare: se ci si è mai chiesti cosa si provi ad



IAN McKINNELL

essere un topo o un leone, questa è l'occasione per provare. Il giocatore può impersonare uno tra sei animali (un falco, un pettirosso, un leone, un topo, una mosca o una farfalla) e sperimentare il tipo di problemi che simili animali devono affrontare quotidianamente per assicurarsi la sopravvivenza.
Il mondo è rappresentato, sullo schermo, da una griglia quadrettata in cui si muove l'animale (raffigurato con la lettera A). L'obiettivo principale degli spostamenti, ottenuti mediante la tastiera, è quello di procurarsi cibo (nel quadrato contrassegnato con una O), evitando i predatori (simboleggiati da una X). Su una porzione di schermo appaiono due indicatori che informano sulla quantità d'energia e d'acqua attualmente disponibili. Se l'energia cala oltre certi limiti, occorre trovar cibo molto in fretta, mentre se è il livello dell'acqua a diminuire, è necessario trovare un rifornimento (un quadrato azzurro).
Per alcuni animali la vita è più dura: la sola fonte di cibo per la farfalla sono i fiori, non sempre facili da trovare. I falchi, invece, possono cibarsi di chiocciole, mosche e topi, ma possono cadere preda del cacciatore umano. Usando questo programma si impara, pur divertendosi, qualcosa sull'ambiente ecologico che ci circonda.

Che accadrebbe se...

Il computer può aiutare a prendere decisioni per il futuro

In qualsiasi decisione che noi prendiamo esiste sempre una certa forma di compromesso: tra il costo e l'affidabilità, o tra il tempo ed il costo, per esempio. È improbabile che, con il minimo "input", si ottenga il massimo "output". Il risultato ideale è sempre una via intermedia tra due estremi.

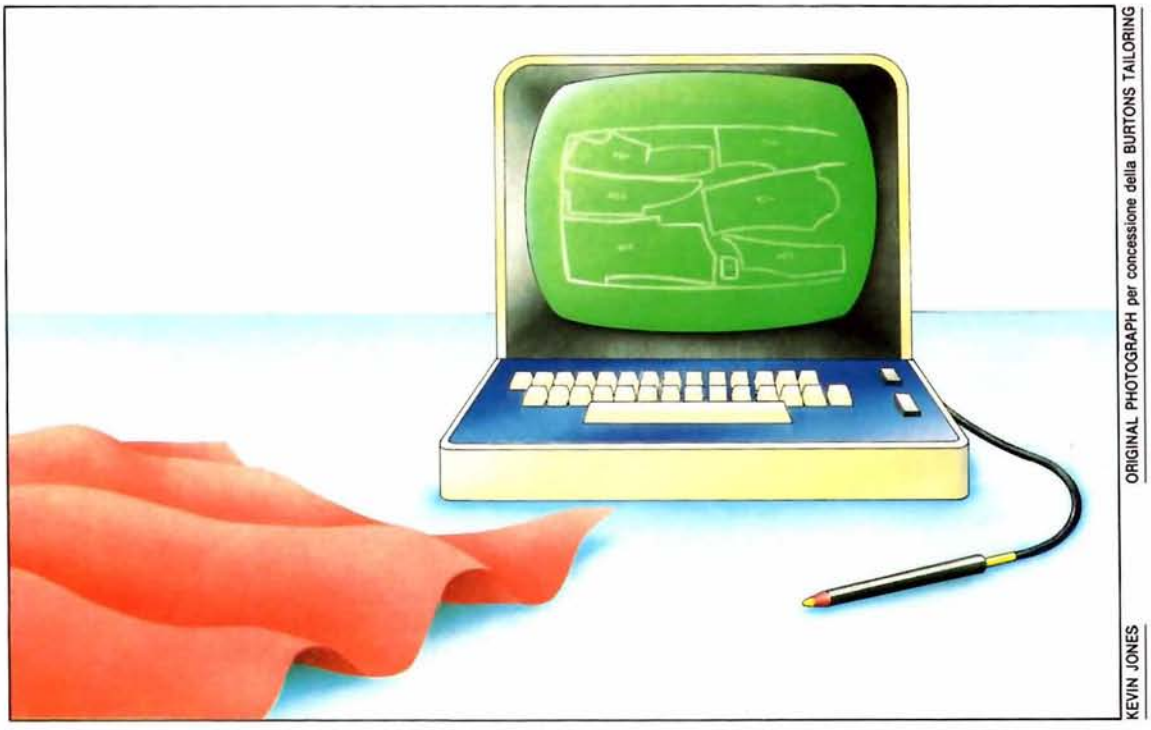
Nella scelta di un detersivo, il ragionamento può essere del tipo: «Il detersivo X costa 100 e pesa 150 grammi, il detersivo Y costa 270 ma pesa 300 grammi. Ma qual è quello veramente più economico, se devo usare una quantità del 20% superiore con quello meno costoso?». Quando le considerazioni si possono ridurre in termini matematici, la risposta può essere relativamente semplice, prima ancora di qualsiasi effettivo calcolo.

Il concetto di associare un "peso" ad un calcolo, riferendolo ad un valore costante, è abbastanza semplice e funzionale, quando le differenze tra componenti di natura simile (ad esempio, il prezzo o il peso) sono di per sé costanti. Ma, quando queste differenze variano in modo indipendente, le cose si complicano alquanto ed è necessario ricorrere a forme di calcolo che implicano la simultanea soluzione di un certo numero di equazioni. Se il numero di elementi è abbastanza piccolo, si può impiegare una matrice. Un altro metodo è quello di tirare a indovinare un ragionevole risultato e quindi lavorare per successive approssimazioni, finché non si giunge alla soluzione. Ovviamente, quanto più il valore iniziale si avvicina alla soluzione, tanto più velocemente si ottiene il risultato.

Tecniche di ottimizzazione simili sono indispensabili nel commercio e nell'industria e sono adottate universalmente, ma in particolare nei processi di fabbricazione. La Programmazione Lineare, l'Analisi del Percorso Critico (Critical Path Analysis) ed il PERT (Programme Evaluation Research Technique, ossia Tecnica di ricerca su stima programmata) non sono che alcuni dei nomi dati ai metodi di ottimizzazione. Antecedenti ai computer di circa una trentina d'anni, detti metodi richiedevano, inizialmente, un notevole impegno di tempo e di lavoro. Oggi sono facilmente risolvibili mediante l'impiego di computer, purché le matrici necessarie non abbiano dimensioni tali da esaurire tutta la memoria. Esistono anche programmi per personal ed home computer studiati appositamente per risolvere problemi di questo genere.

Un settore che ha tratto enormi benefici dai metodi di ottimizzazione è quello tessile: i tessuti vengono prodotti in pezze di dimensioni standard ed il problema maggiore, per una fabbrica di confezioni, è quello di eliminare gli sprechi dovuti a tagli irregolari.

In particolare, è possibile ottimizzare la disposizione delle sagome da taglio in modo da evitare al massimo gli scarti, pur producendo

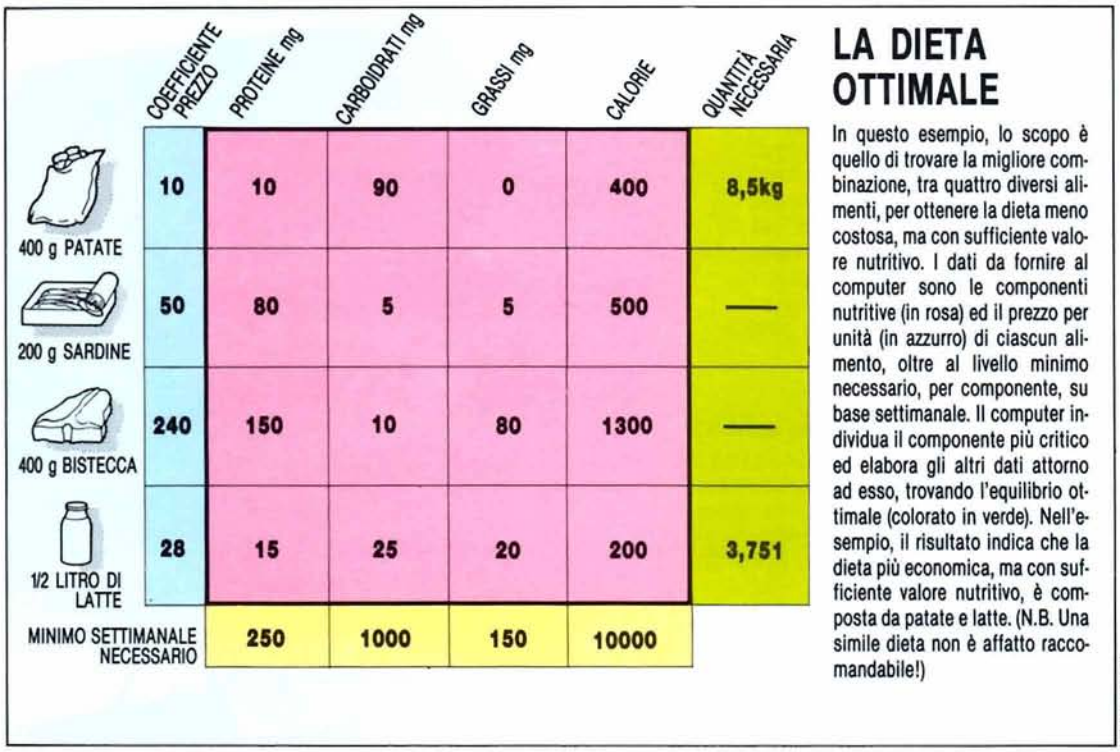


ORIGINAL PHOTOGRAPH per concessione della BURTONS TAILORING

KEVIN JONES

Adattamento perfetto

Un buon esempio di ottimizzazione computerizzata si ha nella ricerca della disposizione migliore (quella che consente i minori scarti di lavorazione) per il taglio di tessuti o di altri materiali. La disposizione ottimale viene presentata sullo schermo del computer e l'operatore può intervenire, se necessario, per apportare eventuali modifiche mediante l'impiego di una penna ottica



	COEFFICIENTE PREZZO	PROTEINE mg	CARBOIDRATI mg	GRASSI mg	CALORIE	QUANTITÀ NECESSARIA
400 g PATATE	10	10	90	0	400	8,5kg
200 g SARDINE	50	80	5	5	500	—
400 g BISTECCA	240	150	10	80	1300	—
1/2 LITRO DI LATTE	28	15	25	20	200	3,751
MINIMO SETTIMANALE NECESSARIO		250	1000	150	10000	

LA DIETA OTTIMALE

In questo esempio, lo scopo è quello di trovare la migliore combinazione, tra quattro diversi alimenti, per ottenere la dieta meno costosa, ma con sufficiente valore nutritivo. I dati da fornire al computer sono le componenti nutritive (in rosa) ed il prezzo per unità (in azzurro) di ciascun alimento, oltre al livello minimo necessario, per componente, su base settimanale. Il computer individua il componente più critico ed elabora gli altri dati attorno ad esso, trovando l'equilibrio ottimale (colorato in verde). Nell'esempio, il risultato indica che la dieta più economica, ma con sufficiente valore nutritivo, è composta da patate e latte. (N.B. Una simile dieta non è affatto raccomandabile!)

KEVIN JONES

abbigliamento su misura e rispettando i disegni dei tessuti. I risultati relativi vengono visualizzati su di uno schermo e l'operatore può intervenire, in base alle proprie esperienze ed al proprio giudizio, per modificare il calcolo del computer, anche se ciò avviene, in media, una sola volta su cinque.

Poiché nella confezione di abiti su misura le caratteristiche variano di volta in volta, questo processo è un valido esempio di combinazione tra procedure di ottimizzazione computerizzate ed esperienza dell'operatore. In altri casi, quando la linea di produzione non subisce altrettante variazioni, l'ottimizzazione produce risultati ancora migliori ed i relativi costi vengono ampiamente ammortizzati dal risparmio dovuto all'esiguità degli scarti.

L'Analisi del Percorso Critico, come suggerito dallo stesso nome, consiste nel determinare con esattezza la migliore sequenza temporale dei vari processi di lavorazione o di costruzione. In sostanza si tratta di formulare, in base alla durata ed al tipo delle varie fasi di lavorazione, un "ruolino di marcia" che consenta il minimo spreco di tempo e, conseguentemente, di denaro. Questo metodo viene adoperato con successo in molte imprese edili, dove, per esempio, la manovalanza ed i materiali vengono impegnati e distribuiti con rigore logico. Anche in questo caso, esiste in commercio una discreta quantità di software per microcomputer.

Sebbene i processi matematici coinvolti nel processo di ottimizzazione siano alquanto complessi per i meno esperti, non v'è dubbio sul successo e la forza di simili tecniche, basate, tutto sommato, sulla riorganizzazione logica e coerente dei problemi da affrontare.



per concessione della MINISTRY OF TRANSPORT

La via migliore

Il progetto delle autostrade, sia in aperta campagna che in prossimità delle città, si basa su criteri di ottimizzazione. Il progettista si occuperà maggiormente dei dislivelli da superare e della curvatura da dare al manto stradale, mentre i proprietari dei terreni sui quali sorgerà la strada ragioneranno secondo altri criteri. In ogni caso, una vasta mole di dati e di rilevazioni viene raccolta a formare un unico modello della situazione, che può servire sia per una rappresentazione grafica che per ottenere un'ottimizzazione del percorso

L'Electron della Acorn

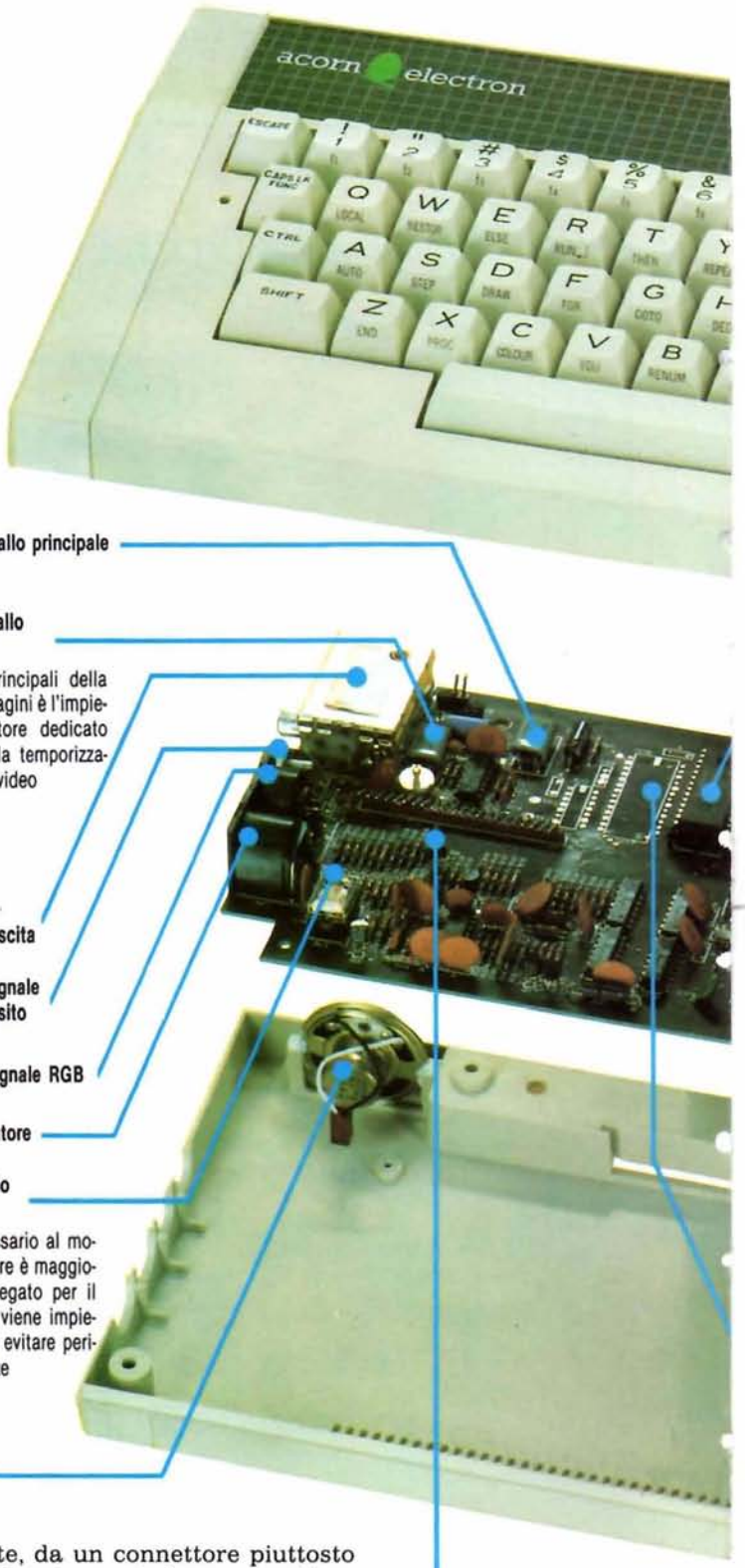
Anatomia di un microcomputer a diffusione europea: tecnologie a confronto

L'Electron, prodotto dalla Acorn (la stessa che ha progettato il Micro BBC), ha un aspetto robusto ed elegante. Rappresentando, per certi versi, una versione ridotta del Micro BBC, le sue prestazioni non sono altrettanto sofisticate, ma l'impiego è senza alcun dubbio facile e comodo. Molti dei comandi disponibili sono identici al BBC: per esempio, SOUND ed ENVELOPE vengono adoperati per generare effetti musicali sintetizzati su ambedue gli apparecchi.
Anche la grafica è la medesima, eccezione fatta per il modo 7 (Teletext) non presente sull'Electron, in quanto richiede un chip a parte. Comunque, il set di caratteri Teletext si può ottenere, con discreta approssimazione, nel modo 6 (nel quale sono disponibili, tuttavia, soltanto due colori). Ciò rappresenta una limitazione, in quanto la preparazione di pagine di testo nel modo 7 consente di risparmiare una notevole quantità di memoria.
Anche le funzioni di input/output sono più limitate rispetto al BBC. L'uscita video avviene sul canale 36 della TV, su apposito canale RGB per monitor oppure in video composito su monitor a colori o monocromo. Ma, a parte l'uscita per il registratore, non vengono offerti altri canali di comunicazione d'uso immediato.



Un duetto dinamico
I progettisti dell'Electron sono Chris Curry (a sinistra) e Herman Hauser (a destra). Quest'ultimo si è occupato anche del progetto del Micro BBC. Curry, invece, ha lavorato in passato per la Sinclair. Assieme hanno fondato la Acorn

JUDY GOLDHILL



Appare evidente, da un connettore piuttosto voluminoso collocato posteriormente all'apparecchio, che sono previste future espansioni al sistema, ma sul manuale non esiste alcuna indicazione sul tipo di segnali disponibili su di esso, né alcun suggerimento sul tipo di apparecchi collegabili.
Il BASIC, incorporato, è la stessa versione adottata con successo nel BBC, ma con tutta una serie di aggiunte interessanti, che rendono la programmazione molto piacevole.
In particolare, esiste una comoda funzione

Connettore per tastiera
Il numero di piedini (22) indica che l'uscita della tastiera non viene decodificata secondo il codice ASCII, altrimenti ne basterebbero 10 (otto per i dati, più due per l'alimentazione). Probabilmente ciò è dovuto all'impiego di una ULA

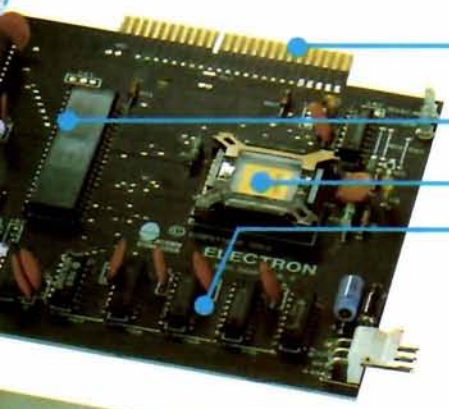


La tastiera

Le caratteristiche di questa tastiera sono tra le migliori nella categoria degli home computer. Non esistono tasti funzione, ma i caratteri di controllo si possono generare mediante la simultanea pressione del CAPS LOCK e di altri tasti. Inoltre, molte delle istruzioni BASIC possono esser impostate con il medesimo sistema, abbreviando i tempi di battitura

Connettore per espansioni

Sebbene il suo impiego non sia documentato, è evidente che questo connettore serve per future espansioni, tra le quali è presumibile un'unità a dischi



ROM

La CPU

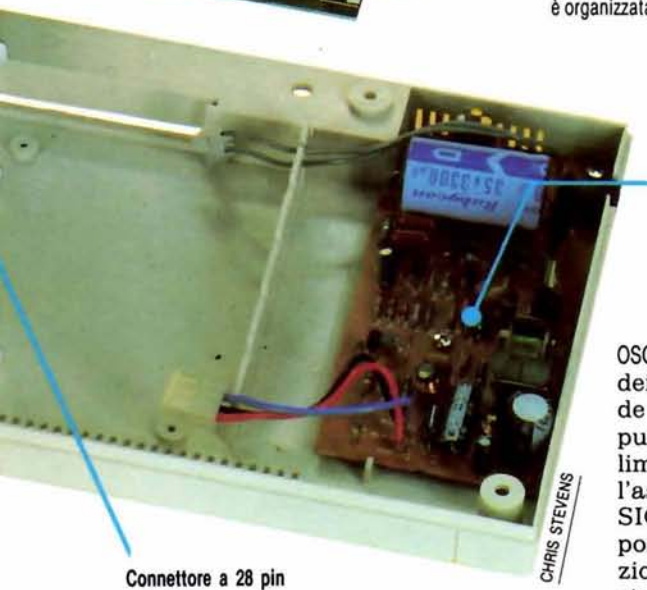
Viene impiegato l'ormai noto chip 6502A, con una temporizzazione di 1.79 MHz

Il chip ULA

Questo circuito è stato appositamente creato per l'Electron ed è la più grande "Uncommitted Logic Array" finora realizzata. Oltre alla CPU, all'ULA, alla ROM ed alla RAM, sono presenti soltanto altri otto circuiti integrati, per la maggior parte porte logiche

La RAM

I byte vengono letti dalla CPU in due metà: i primi quattro bit provengono da altrettanti chip di RAM, i successivi da altri quattro chip. In altri apparecchi, la RAM è organizzata in gruppi di otto bit



CHRIS STEVENS

Alimentatore stabilizzato

L'Electron funziona ad un voltaggio abbastanza atipico: 19 volt. Questo è il circuito di adattamento e di stabilizzazione della tensione

Connettore a 28 pin

Questa zona del circuito stampato, benché predisposta al probabile inserimento di una ROM aggiuntiva, non è, per il momento, utilizzata

OSCLI, che consente di inviare direttamente dei comandi al sistema operativo: chi possiede una certa esperienza di programmazione può sfruttarla per eliminare alcune delle limitazioni del BASIC. Anche il software dell'assembler, caratteristica esclusiva del BASIC del BBC, è stato notevolmente ampliato e possiede particolari istruzioni per la definizione delle variabili e per la stampa delle stringhe.

In complesso, l'Electron offre delle prestazioni superiori alla media: le immagini sono stabili e ben definite, i colori buoni. In futuro, con l'aggiunta di unità a dischetti ed altre espansioni, l'Electron può diventare un apparecchio da non sottovalutare.

ELECTRON

DIMENSIONI

mm. 340 x 160 x 65

PESO

—

CPU

6502A

CLOCK

1,79 MHz

MEMORIA

64 Kbyte di ROM, 32 Kbyte di RAM (senza espansioni, per il momento)

VIDEO

32 linee di 80 caratteri, 8 colori (sfondo e primo piano indipendenti), 127 caratteri predefiniti, 255 definibili da utente

INTERFACCE

Connettore per TV, per monitor RGB, per video composito, per registratore a cassette, connettore per espansioni (non documentato)

LINGUAGGI FORNITI

BASIC BBC fornito di assembler

ALTRI LINGUAGGI DISPONIBILI

I linguaggi della Acornsoft dovrebbero funzionare sull'Electron (FORTH, LISP, ecc.), ma non esiste attualmente la possibilità di collegare le cartucce ROM

ACCESSORI COMPRESI

Trasformatore d'alimentazione, cavo di connessione per TV, un nastro con programma dimostrativo, 2 manuali

TASTIERA

56 tasti tipo macchina da scrivere. Immissione abbreviata dei comandi BASIC. Possibilità di associare funzioni ad un massimo di 10 tasti

DOCUMENTAZIONE

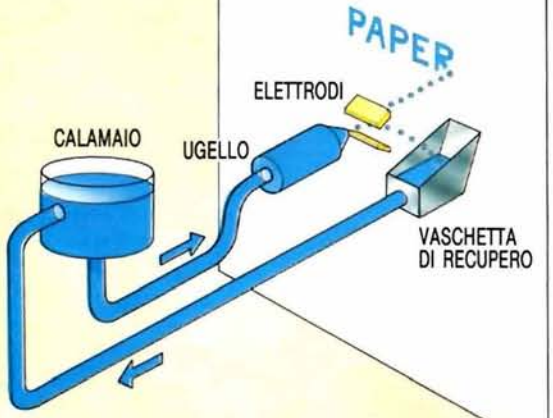
Semplicemente eccellente. Sono esposte moltissime informazioni sia per il principiante che per l'esperto. Ciascun comando ed istruzione BASIC è trattato individualmente ed esiste anche una sezione dedicata al linguaggio Assembler. Le funzioni del sistema operativo sono spiegate egregiamente. L'impiego dell'apparecchio non dovrebbe presentare alcun serio inconveniente anche per i meno esperti

Le stampanti "a getto"

Una stampa veloce, silenziosa e di buona qualità

MISSILI PILOTATI

Le prime stampanti a getto usavano un sistema molto sofisticato ed erano assai costose. All'interno dell'ugello, le minuscole particelle d'inchiostro venivano caricate elettricamente e quindi ininterrottamente "sparate" sulla carta. Per evitare di riempire anche le parti da lasciare in bianco, l'inchiostro veniva opportunamente deviato in basso, verso una piccola vaschetta, e successivamente riciclato



I sistemi di stampa adoperati nelle stampanti sono di vario genere e qualità: i risultati migliori, in genere, si ottengono con quelli ad impatto (le stampanti "a margherita" sono di questo genere), mentre i peggiori provengono dall'impiego di unità di scrittura termiche o elettrostatiche. Tuttavia, il sistema più diffuso, benché rumoroso e di qualità mediocre, è quello che adopera una matrice di aghi (vedere a pag. 74).

Quando apparvero le prime stampanti/plotter, quali la Tandy CGP 115, le limitazioni delle stampanti a matrice d'aghi divennero più appariscenti. Il metodo adottato, in questo caso, consiste in una testina di scrittura, dotata di minuscole penne a sfera, solitamente a quattro colori. Comunque, il tipo di stampante che probabilmente spodesterà tutti gli altri opera su un principio ben diverso: un getto d'inchiostro microscopico, opportunamente controllato. Questi apparecchi sono noti col nome di stampanti a getto.

Ormai ben accreditate in larghi settori commerciali ed industriali, le stampanti a getto si stanno affermando anche nel campo dei personal e degli home computer. Il funzionamento può essere così descritto: l'inchiostro liquido viene pompato da un piccolo deposito; la punta di scrittura (un minuscolo ugello) "spara" minuscole particelle d'inchiostro, che vengono caricate elettricamente. Il meccanismo è, in genere, realizzato in materiale piezoelettrico, capace di sopportare vibrazioni ad alta frequenza, necessarie alla formazione dei caratteri.

Mentre nelle normali stampanti il foglio di carta aderisce ad un rullo di gomma, in questo caso la carta viene a trovarsi tra l'ugello ed una striscia metallica, che viene caricata elettricamente in modo da attirare le particelle d'inchiostro. Questa tecnica può sembrare, a prima vista, poco affidabile, ma, al contrario, i risultati sono molto apprezzabili e gli unici eventuali inconvenienti possono essere l'intasamento dell'ugello o una leggera sbavatura nei caratteri.

La formazione dei caratteri avviene in modo molto simile a quello delle matrici ad aghi: la sequenza di codici ASCII inviata alla stampante viene temporaneamente appoggiata in in buffer (deposito transitorio), finché non arriva il codice di "Ritorno Carrello" o il buffer è pieno. A questo punto, viene esaminato ciascun carattere, prelevando la corri-

Pompa d'innesco

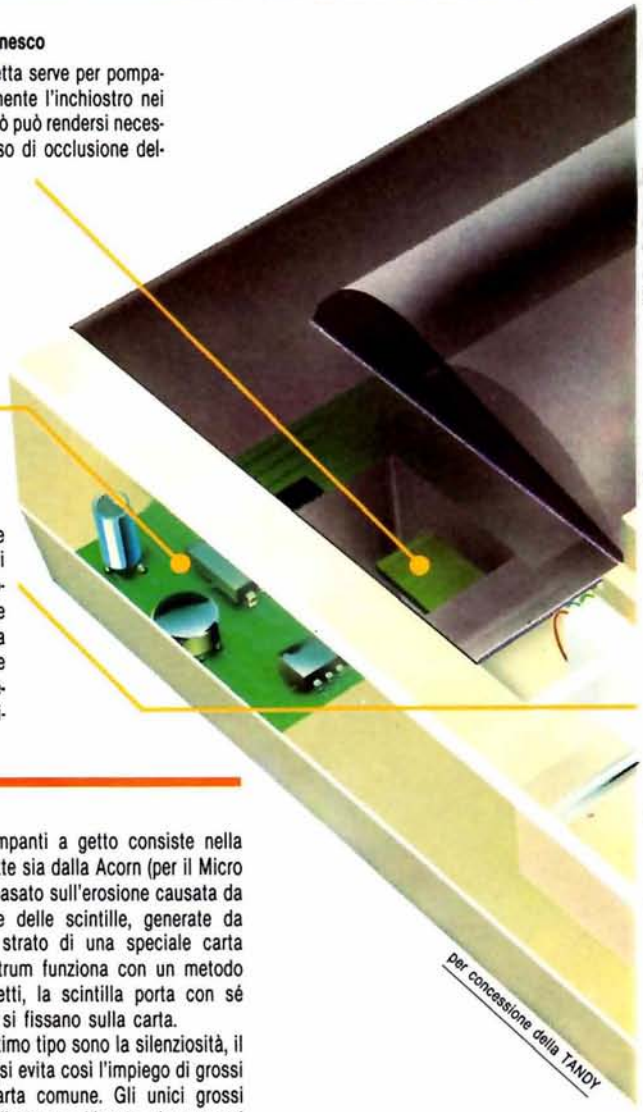
Questa levetta serve per pompare manualmente l'inchiostro nei condotti. Ciò può rendersi necessario in caso di occlusione dell'ugello

Scheda del circuito

Questa stampante è dotata di un proprio microprocessore 6809, memorie RAM e ROM. Tutti i dati in arrivo vengono "bufferizzati" (appoggiati in un deposito temporaneo) in attesa di venire stampati

Posizione di riposo

Il meccanismo di una stampante a getto è molto più delicato di quello delle stampanti tradizionali, e la testina di stampa deve essere "riposta" dopo l'uso. La procedura di avviamento iniziale non è complicata e conviene seguirla scrupolosamente per evitare danni



Per concessione della TANDY

Caratteri ... scintillanti

Un'interessante variazione delle stampanti a getto consiste nella stampa ad "inchiostro secco". Prodotte sia dalla Acorn (per il Micro BBC) che dalla Olivetti, il sistema è basato sull'erosione causata da scintille. Vengono, infatti, impiegate delle scintille, generate da un'alta tensione, per "forare" uno strato di una speciale carta argentata. La stampante dello Spectrum funziona con un metodo simile, mentre, in quella della Olivetti, la scintilla porta con sé minuscole particelle di carbone, che si fissano sulla carta.

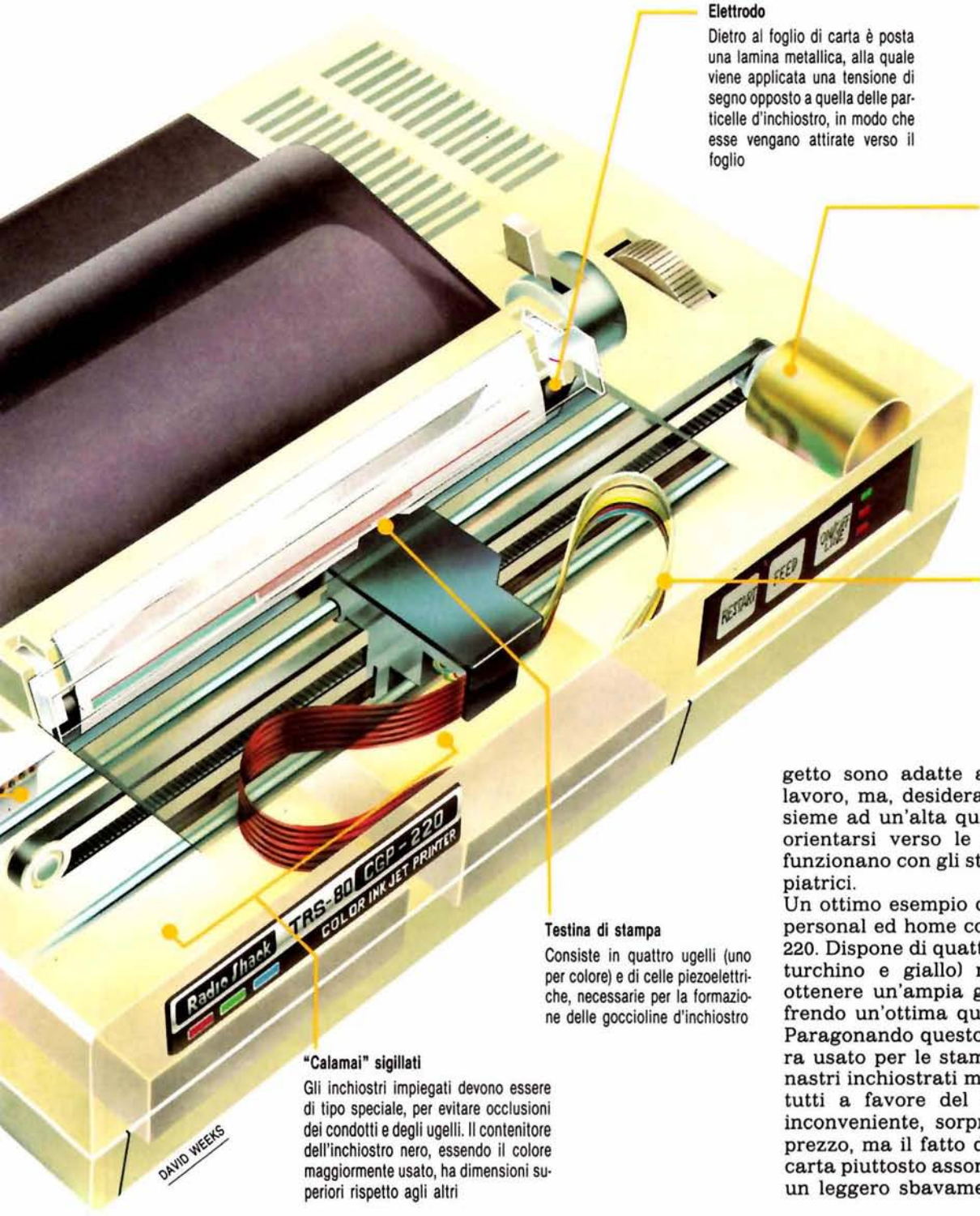
I vantaggi delle stampanti di quest'ultimo tipo sono la silenziosità, il minimo peso della testina di stampa (si evita così l'impiego di grossi motori) e la possibilità di usare carta comune. Gli unici grossi svantaggi sono, invece, la lentezza di stampa (dovuta ai numerosi passaggi necessari per completare una singola riga) ed una leggera sbavatura nei caratteri

spondente sequenza di codici, memorizzati nelle ROM, che formano l'immagine. Ogni carattere è composto da una particolare disposizione di punti, disposti su di una matrice di 8 × 8 elementi. Per la stampa di un'intera linea sono quindi necessari otto passaggi della testina di stampa. Per risparmiare tempo, la stampa avviene in ambedue le direzioni del movimento (stampa bidirezionale). Durante la stampa del primo gruppo di caratteri, estratti via via dal buffer, un secondo buffer viene riempito con i caratteri successivi. Grazie a questi sistemi, le stampanti a getto per applicazioni commerciali sono capaci di produrre un'intera pagina di stampa in pochi secondi.

La qualità di stampa dipende dal tipo di carta impiegato: più la carta è assorbente, maggiore sarà la sbavatura dei caratteri, ma in condizioni ottimali, la qualità di questo tipo di stampante supera di gran lunga quella delle stampanti ad aghi. Le stampanti a getto sono adatte ad una grande mole di lavoro, ma, desiderando un'alta velocità assieme ad un'alta qualità, la scelta dovrebbe orientarsi verso le stampanti a laser, che funzionano con gli stessi principi delle fotocopiatrici.

Un ottimo esempio di stampante a getto per personal ed home computer è la Tandy CGP 220. Dispone di quattro colori (nero, magenta, turchino e giallo) miscelabili tra loro per ottenere un'ampia gamma di sfumature, offrendo un'ottima qualità di stampa.

Paragonando questo sistema con quello finora usato per le stampanti tradizionali, ossia nastri inchiostrati multicolori, i risultati sono tutti a favore del sistema Tandy. L'unico inconveniente, sorprendentemente, non è il prezzo, ma il fatto di dover usare un tipo di carta piuttosto assorbente, il che può causare un leggero sbavamento delle immagini.



Elettrodo
Dietro al foglio di carta è posta una lamina metallica, alla quale viene applicata una tensione di segno opposto a quella delle particelle d'inchiostro, in modo che esse vengano attirate verso il foglio

Motori di trascinamento
Il movimento trasversale della testina di stampa è ottenuto con l'impiego di un normale motore, mentre per il trascinamento verticale della carta si ricorre ad un motore "passo passo"

Condotti flessibili
La maggior parte delle stampanti adopera cavi elettrici flessibili per collegare la testina elettromagnetica ai circuiti. Nelle stampanti a getto, esiste un ulteriore problema: permettere lo scorrimento di più inchiostri colorati in condotti adatti a seguire la testina nei suoi rapidi spostamenti

Testina di stampa
Consiste in quattro ugelli (uno per colore) e di celle piezoelettriche, necessarie per la formazione delle goccioline d'inchiostro

"Calamai" sigillati
Gli inchiostri impiegati devono essere di tipo speciale, per evitare occlusioni dei condotti e degli ugelli. Il contenitore dell'inchiostro nero, essendo il colore maggiormente usato, ha dimensioni superiori rispetto agli altri

A prova di suono

Effetti audio sul Dragon 32

Il Dragon 32 è dotato di un solo oscillatore ad onda quadra per la generazione degli effetti sonori, ma le semplici istruzioni dello "Extended Colour BASIC" della Microsoft consentono la costruzione di sequenze musicali eseguibili con un solo comando. Sfortunatamente, non è previsto un generatore di rumore che tornerebbe utile per creare i tanti effetti sonori richiesti dai giochi.
Il formato del comando SOUND è il seguente:

```
SOUND A,D
```

dove: A = Altezza (da 1 a 255) e D = Durata (da 1 a 255). L'altezza dei suoni non è molto precisa e non rispecchia fedelmente il progredire della scala musicale: approssimativamente, il do centrale corrisponde al valore 89 mentre il la a 440 Hz al valore 159. Una durata di valore 16 equivale, grosso modo, ad un secondo; raddoppiando il valore si raddoppia anche la durata.
Questo breve programma dimostra l'uso di SOUND per generare un effetto sonoro che potremmo definire il decollo di un UFO:

```
10 FOR A = 10 TO 170 STEP 10
20 FOR D = 16 TO 1 STEP -1
30 SOUND A,D
40 NEXT D
50 NEXT P
```

Con l'istruzione PLAY si determina l'altezza, la durata ed il volume di un suono, ma è anche possibile specificare un insieme di note (sotto forma di stringa) da suonare in successione, e ciò facilita la composizione di piccoli brani:

```
PLAY "T;O;V;L;N;P"
```

dove: T = Tempo (da T1 a T255); O = Ottava (da O1 a O5); V = Volume (da V0 a V15); L = Lunghezza della nota (da L1 a L255); N = Valore della nota (da 1 a 12 o la lettera corrispondente alla nota) e P = Pausa (da P1 a P255).
I punti e virgola, tra i parametri, servono unicamente ai fini della chiarezza e si possono tranquillamente omettere. Inoltre, l'ordine dei parametri non è critico. I valori T, O, V ed L rimangono inalterati, fino a diversa indicazione. Se non specificati, i parametri assumono i seguenti valori standard: T2, O2, V15, L4 e P0. I parametri L e P possono essere considerati come "note" e "frazioni di note", ossia L1 e P1 rappresentano la nota intera, L2 e P2 una nota di metà valore, ecc. Lo scandire delle note è

Immagini d'autore

Ancora grafica sul BBC Modello B

Il BBC non dispone dell'intera gamma di comandi per la grafica ad alta risoluzione, comune in altri microcomputer: non esiste, per esempio, un comando CIRCLE o PAINT. Tuttavia, con qualche linea di BASIC si riesce a sopperire a queste mancanze.
Il sistema di coordinate, sullo schermo, rimane sempre invariato, qualunque sia il modo grafico di funzionamento e l'origine è posta nell'angolo in basso a sinistra. Il principale comando grafico è:

```
MOVE x,y
```

che consente di muovere il cursore grafico dal punto in cui si trova fino al punto di coordinate x ed y. Non viene, tuttavia, tracciata alcuna linea di congiunzione tra i punti. Si noti che il cursore grafico può spostarsi indipendentemente da quello usato per le scritte (cursore del testo). Un altro comando è:

```
DRAW x,y
```

e serve, appunto, per tracciare linee da un punto all'altro dello schermo. Il comando

```
PLOT k,x,y
```

invece, è un comando complesso e le sue funzioni dipendono dal valore attribuito a k:

Valore di k	Funzione
0	movimento relativo al punto precedente
1	traccia linea dall'origine (nel colore usato in primo piano)
2	come sopra, ma in colore inverso

stabilito dal parametro T (T1 = durata lunga, T255 = durata breve). Inoltre, la durata della nota può essere modificata inserendo dei puntini: L1... oppure L5..., dove a ciascun puntino corrisponde un aumento di durata pari a metà del valore della nota. In altre parole L1... = 1 + 1/2 + 1/2 + 1/2 = 2 1/2 note, mentre L5 = 1/5 + 1/10 = 3/10.
Non esiste un metodo preciso per stabilire i rapporti tra le note e le durate, per cui è necessario procedere per tentativi.
Il parametro O stabilisce l'ottava da usare per la successiva nota: l'ottava O1 inizia dal do a 131 Hz, mentre la O5 termina col si a 2093 Hz. Il do centrale è la nota iniziale dell'ottava O2. All'interno di un'ottava, le note si possono specificare in due modi. Si può usare un numero identificativo:

1	2	3	4	5	6
Do	Do#	Re	Re#	Mi	Fa
7	8	9	10	11	12
Fa#	Sol	Sol#	La	La#	Si

Ciò permette di far corrispondere la nota ad una variabile numerica. In alternativa, si può usare direttamente la lettera corrispondente alla nota (nella notazione anglosassone).
Facciamo un esempio: il seguente comando suona un Fa (6) nell'ottava O2 per una durata di metà valore (L2) al volume standard V15. Viene quindi eseguita una pausa di un quarto di valore (L1) a volume V20. Il Tempo ha valore T3:

```
PLAY "T3;L2;6;P4;O3;V20;L1;A#"

    (Fa)   ( La# )

        pausa
```

Inoltre, i parametri T, O, V ed L possono essere variati di quantità prefissate aggiungendo un suffisso:

Suffisso	Effetto
+	Somma 1 al valore corrente
—	Sottrae 1 al valore corrente
>	Moltiplica il valore corrente per 2
<	Divide il valore corrente per 2

Il formato è: T+, T—, T>, o T< per ciascun parametro.
Una caratteristica del Dragon 32 è la possibilità di impiegare più stringhe per comporre un unico brano e ciò facilita la ripetizione di frammenti identici, ad esempio:

```
10 A$ = "F;A#;G"
20 B$ = "C;D#;F;P4;XA$;"
30 PLAY B$
```

In questo caso, viene definita una stringa A$ che è adoperata all'interno di B$, specificando la sequenza XA$. Il motivo risultante è composto da Do-Re#-Fa-P4-Fa-La#-Sol. Il punto e virgola, dopo XA$, deve essere incluso.

3	come sopra, ma nel colore dello sfondo
4	identico al comando MOVE
5	identico al comando DRAW
6	identico a DRAW ma nel colore inverso
7	identico a DRAW ma nel colore dello sfondo

Attribuendo a k dei valori superiori a 7, lo schema si ripete, ma si aggiungono nuovi effetti (ad esempio, le linee sono tratteggiate). I valori di k compresi tra 80 ed 87 si riferiscono a funzioni piuttosto particolari: PLOT80,x,y unisce il punto x,y ai due precedentemente "plottati", per formare un triangolo, il cui interno viene colorato con il colore prescelto per il primo piano. In parte, questa funzione sopperisce alla mancanza del comando PAINT. Il comando VDU x equivale a scrivere PRINT CHR$(x). Abbiamo visto, nella prima parte dedicata alla grafica del BBC, che l'istruzione VDU è seguita da alcuni valori. VDU v,w,x,y,z equivale a:

```
PRINT CHR$(v);CHR$(w);CHR$(x);CHR$(y);CHR$(z)
```

La VDU permette d'intervenire su quella parte del sistema operativo che controlla la visualizzazione. Sebbene usate principalmente in BASIC, le funzioni ottenute dall'istruzione VDU sono accessibili anche da altri linguaggi. Definire nuovi caratteri, nel BBC, è abbastanza semplice e l'istruzione VDU 23 provvede a questo. Abbiamo visto a pagina 334, che i normali caratteri ASCII vengono costruiti su griglie di 8 × 8 punti (o pixel). I pixel visibili sono rappresentati dal valore binario 1, mentre quelli invisibili dal valore 0. Ciascuna riga di otto bit può quindi essere convertita in un numero decimale ed un intero carattere viene definito con una serie di otto numeri. L'istruzione VDU 23 permette di ridefinire i caratteri aventi codice compreso tra 224 e 255. Il seguente programma, per esempio:

```
10 REM DEFINIZIONE DI UN CARATTERE
20 MODE 2
30 VDU 23,240,16,56,124,146,16,16,16,0
40 PRINT CHR$(240)
50 END
```

definisce e visualizza un nuovo carattere (una freccetta), attribuendogli il codice 240. Le istruzioni VDU 24 e VDU 28 controllano, rispettivamente, la creazione sullo schermo di "finestre" per la grafica o per i testi. Usandoli, la visualizzazione viene limitata a particolari zone dello schermo. La definizione di una finestra si ottiene semplicemente specificando le coordinate dell'angolo in basso a sinistra e dell'angolo in alto a destra.

Disegniamo un fiore

Il seguente programma genera, sullo schermo, l'immagine di un fiore, usando la risoluzione grafica MODE 1. Si noti come si ottiene la colorazione dell'interno dei triangoli che formano i petali:

```
10 REM FIORE
20 CLS
30 MODE 1
40 FOR D = 1 TO 3
50 A = 600 : B = 500
60 MOVE A,B
70 FOR C = 1 TO 550 STEP3
80 GCOLO,RND(3)
90 S=(C/(RND(5)+10))
100 X=S*5*SIN(C/16)+A
110 Y=S*5*COS(C/16)+B
120 PLOT85,X,Y
130 NEXT C
140 NEXT D
150 END
```

Mediante l'uso delle funzioni trigonometriche SIN e COS (seno e coseno) si genera l'immagine della spirale. Una tale relazione tra le coordinate x ed y, normalmente, produrrebbe un cerchio, ma nel ciclo FOR...NEXT il raggio C viene progressivamente incrementato. Il centro della spirale può essere spostato modificando le sue coordinate, ossia i valori A e B

Un "giro" di prova

Un'elementare verifica dei programmi è sempre utile

L'ultima lezione era terminata con un dilemma: come far sì che un programma, al suo primo "lancio", legga un file se questo non esiste ancora. Infatti, la prima operazione della nostra rubrica computerizzata è quella di leggere i dati conservati su disco o cassetta e trasferirli in una matrice o in più variabili. Una soluzione potrebbe consistere nell'eseguire delle scritture ogni qualvolta il programma viene lanciato, ma una simile tecnica è pericolosa in quanto si potrebbero cancellare i dati già registrati. Del resto, il tentativo di "aprire" un file inesistente causerebbe un errore, al quale il BASIC reagirebbe bloccando l'esecuzione del programma.
Fortunatamente, esiste un metodo abbastanza pratico per risolvere il problema. Molti dei "package" di software in commercio sono corredati di uno o più programmi che effettuano l'installazione iniziale. In genere, questi moduli servono per memorizzare, una volta per tutte, le caratteristiche del terminale e della stampante adottati in un particolare sistema, oltre a creare i file occorrenti al funzionamento del package (si ricordi che i file contenenti dati, al contrario di quelli che contengono programmi, sono accessibili da più programmi, come detto a pagina 316).
Possiamo quindi scrivere un breve programma "d'installazione" (chiamato anche "set-up") che, scrivendo una serie fittizia di dati nel file, permetta di usare con tranquillità la routine RDINFL fin dalla prima volta. Ovviamente, questi dati fittizi devono essere tali da non interferire col normale funzionamento della rubrica ed essere riconosciuti immediatamente come fittizi. Un valido esempio di dato fittizio potrebbe essere la stringa "@PRIMO", poiché è molto improbabile che un normale nominativo inizi con una simile sequenza di caratteri. Occorre, quindi, modificare leggermente la routine RDINFL, cosicché, all'apertura del file, controlli l'esistenza di questa speciale stringa prima di procedere oltre. Volendo, si può impiegare un carattere diverso da @, purché si sia certi che esso non farà mai parte di un nominativo. Ecco, dunque, il programma di set-up:

```
10 REM CREAZIONE DEL FILE DEI DATI
20 REM DA USARE PER LA RUBRICA COMPUTERIZZATA
30 REM SCRIVE UN RECORD 'FITTIZIO'
40 REM RICONOSCIUTO DALLA ROUTINE RDINFL
50 REM
60 REM
70 OPEN "O",#1,"RUBR.DAT"
80 PRINT #1,"@PRIMO"
90 CLOSE #1
100 END
```

Come già accennato precedentemente nel corso, le istruzioni relative al trattamento dei file possono variare da una versione di BASIC all'altra, ma i principi non cambiano. In primo luogo, il file deve essere "aperto", indicando se l'operazione da eseguire riguarda una lettura o una scrittura e, contemporaneamente, assegnando al file un "canale" di comunicazione. Quest'ultima specifica permette di adoperare più di un file simultaneamente. Infine, occorre specificare il nome da noi assegnato al file.
La linea 70 contiene tutte queste informazioni nella versione BASIC della Microsoft, ma l'istruzione OPEN è molto simile anche in altre versioni (per la OPEN del Micro BBC vedere a pagina 319).
Esaminiamola più dettagliatamente: OPEN (apri) indica che intendiamo "aprire" un file; la "O", abbreviazione di OUT, significa che il file viene aperto in uscita, ossia in scrittura; #1 è il numero del canale assegnato al file (utilizzato nelle successive operazioni) e "RUBR.DAT", infine, è il nome da noi assegnato al file.
La linea 80 non fa che scrivere un singolo record nel file. Nella gran parte dei BASIC, la sintassi, relativa alle operazioni di scrittura su file, è identica a quella impiegata nelle operazioni di visualizzazione, salvo indicare il canale del file (nel nostro caso #1).
La linea 90 "chiude" (CLOSE) il file. Teoricamente, un file può restare aperto per tutta l'esecuzione del programma, ma in questa condizione esso è molto vulnerabile (un'improvvisa mancanza di corrente, ad esempio, potrebbe danneggiare irrimediabilmente tutti i dati). Conviene quindi chiudere al più presto un file se non si ha più intenzione di operare su di esso.
Esiste una certa confusione tra i termini file e record, quando riferiti ad un computer, particolarmente aggravata quando si tratta di programmi per database. In un database, un file costituisce un insieme di informazioni tra loro collegate. Per analogia, in un ufficio un file può esser paragonato al cassetto di uno schedario (quello del PERSONALE, ad esempio). Questo file conterrà numerosi record (le schede individuali) ed ogni record, a sua volta, sarà composto da una serie di campi (gruppi d'informazioni), quali: il NOME, il SESSO, l'ETÀ, lo STIPENDIO, ecc.
Trasferendo l'archivio del PERSONALE in uno gestito da computer, la struttura dei dati non verrebbe un gran che alterata, anzi, sarebbe identica.
Un file sequenziale di un computer, invece,

raccoglie una serie di dati e ciascun elemento costituisce un record. Pertanto, il record di un file, nella generalità dei casi, può non corrispondere esattamente ad un record di un database.
È compito del programma leggere i dati dal file ed assegnarli a delle variabili, organizzandoli per formare la struttura logica di un record. Si eviti, dunque, di confondere il record di un file con quello di un database.
Il programma di set-up, eseguito una volta per tutte, non viene più usato. Addirittura, usarlo nuovamente causerebbe certamente la perdita dei dati eventualmente inseriti nell'archivio. Esamineremo questo rischio parlando della versione modificata della routine RDINFL.
Quando viene eseguito il programma della rubrica computerizzata, esso non può sapere, a priori, se il contenuto dell'archivio "RUBR.DAT" è valido oppure no: la prima operazione della routine RDINFL consiste nell'apertura del file e nella lettura del primo record in esso contenuto. Questo record, contrariamente a quanto si potrebbe credere, non viene trasferito in un elemento di una matrice, ma in una speciale variabile alfanumerica, che abbiamo chiamato TEST$.
Prima di tentare la lettura di un successivo record, viene fatta una verifica per vedere se TEST$ contiene la stringa "@PRIMO". Se questo è il caso, il programma conclude che non esistono altri record nel file ed è inutile tentare ulteriori letture e relativi trasferimenti nelle variabili. Di conseguenza, il file può esser chiuso e l'esecuzione passare ad altre parti del programma. Ad ogni buon conto, qualsiasi altra funzione del programma, che non sia l'inserimento di un nominativo, non ha alcun senso. Tanto vale, quindi, sfruttare il valore di TEST$ per decidere se visualizzare il menù delle opzioni o saltare direttamente all'inserimento del primo nominativo.
Qualora TEST$ non contenga la stringa "@PRIMO", allora il programma potrà ragionevolmente supporre che esiste una serie di dati validi nel file, che devono essere trasferiti nelle apposite matrici. Questa è la subroutine RDINFL modificata:

```
1400 REM SUBROUTINE RDINFL
1410 OPEN "I",#1,"RUBR.DAT"
1420 INPUT #1, TEST$
1430 IF TEST$ = "@PRIMO" THEN GOTO 1530: REM ARCHIVIOVUOTO
1440 LET CAMNOM$(1) = TEST$
1450 INPUT #1,CAMMOD$(1),CAMVIA$(1),CAMLOC$,(1)
1460 INPUT #1,CAMNAZ$(1),CAMTEL$(1),CAMIND$(1)
1470 LET DIMENSIONE = 2
1480 FOR L = 2 TO 50
1490 INPUT #1,CAMNOM$(L),CAMMOD$(L),CAMVIA$(L)
1500 INPUT #1,CAMLOC$(L),CAMNAZ$(L),CAMTEL$(L),CAMIND$(L)
1510 REM SPAZIO PER UNA CHIAMATA ALLA SUBROUTINE DIMEN
1520 NEXT L
1530 CLOSE #1
1540 RETURN
```

La linea 1420 assegna un singolo record dal file "RUBR.DAT" alla variabile TEST$. La linea successiva verifica se il contenuto di TEST$ è "@PRIMO": se lo è viene fatto un salto (GOTO) alla linea che chiude il file, ritornando poi al programma principale mediante una RETURN. Non viene fatto alcun ulteriore tentativo di lettura. Non essendo presente alcun dato, l'esecuzione può passare alla routine INITIL, la quale chiamerà la SETFLG. Quest'ultima, per il momento, provvede soltanto ad assegnare il valore 1 a DIMENSIONE se TEST$ = "@PRIMO". Ecco una prima versione di questa subroutine, dove si può notare la presenza di numerose REM, che verranno in seguito sostituite con opportune istruzioni di inizializzazione:

```
1600 REM SUBROUTINE SETFLG
1610 REM MODIFICA LE FLAG DOPO RDINFL
1620 REM
1630 REM
1640 IF TEST$ = "@PRIMO" THEN LET DIMENSIONE = 1
1650 REM
1660 REM
1670 REM
1680 REM
1690 RETURN
```

L'esecuzione ritorna a INITIL e da qui al programma principale, MAINPG, che chiama la subroutine GREETS. Quest'ultima, che provvede a visualizzare un messaggio introduttivo, non necessita di alcuna modifica rispetto a quanto già presentato.
La successiva subroutine chiamata dal programma principale è CHOOSE. Una piccola modifica a questo modulo, rispetto a quello riportato a pagina 357, permette di "forzare" l'immissione di un nominativo, qualora il file dei dati sia vuoto.

```
3500 REM SUBROUTINE CHOOSE
3510 REM
3520 IF TEST$ = "@PRIMO" THEN GOSUB 3860
3530 IF TEST$ = "@PRIMO" THEN RETURN
3540 REM MODULO 'CHMENU'
3550 PRINT CHR$(12)
3560 PRINT "MENU DELLE OPZIONI:"
3570 PRINT
3580 PRINT
3590 PRINT
3600 PRINT "1. TROVARE UN RECORD (USANDO UN NOME)"
3610 PRINT "2. TROVARE UN RECORD (USANDO PARTE DEL NOME)"
3620 PRINT "3. TROVARE UN RECORD (USANDO LA LOCALITÀ)"
3630 PRINT "4. TROVARE UN RECORD (USANDO LE INIZIALI)"
3640 PRINT "5. ELENCO DI TUTTI I RECORD"
3650 PRINT "6. IMMISSIONE DI UN RECORD"
3660 PRINT "7. VARIAZIONE DI UN RECORD"
3670 PRINT "8. ANNULLAMENTO DI UN RECORD"
3680 PRINT "9. FINE LAVORO"
3690 PRINT
3700 PRINT
3710 REM MODULO 'INCHOI'
3720 REM
3730 LET L = 0
3740 LET I = 0
3750 FOR L = 0 TO 1
3760 PRINT "QUALE (1-9)"
3770 FOR I = 1 TO 1
3780 LET A$ = INKEY$
3790 IF A$ = " " THEN I = 0
3800 NEXT I
3810 LET SCEL = VAL(A$)
3820 IF SCEL < 1 THEN LET L = 0 ELSE LET L = 1
3830 IF SCEL > 9 THEN LET L = 0
3840 NEXT L
3850 RETURN
```

Sono state aggiunte due linee. La prima contiene una verifica sul contenuto di TEST$:

se questo è "@PRIMO", allora non esistono dati validi nel file e si può passare immediatamente alla subroutine FIRSTM (linea 3860), che, dopo aver visualizzato un opportuno messaggio, assegni il valore 6 alla variabile SCEL (corrispondente alla subroutine AGGREC), consentendo l'immissione del primo record. Al ritorno dalla subroutine FIRSTM, l'esecuzione passa alla linea 3530, dove il contenuto di TEST$ viene di nuovo controllato, permettendo di tornare al programma chiamante senza eseguire il resto di CHOOSE.
Questa seconda verifica, che potrebbe sembrare superflua, consente invece di evitare la presentazione del menu delle opzioni. È stato evitato l'impiego di una GOTO, inserendo direttamente una RETURN, ma il risultato sarebbe identico scrivendo: IF TEST$ = "@PRIMO" THEN GOTO 3850. L'uso incontrollato di istruzioni GOTO rende difficoltosa la rilettura di un programma, e, curiosamente, i programmatori d'oltre oceano indicano l'abuso di GOTO con l'espressione "spaghetti coding" (traducibile più o meno come "spaghettata di codice").
Prima di passare alla subroutine FIRSTM, vorremmo precisare che la GOTO nella linea 1430 non contraddice quanto appena detto: infatti, costituisce l'eccezione che conferma la regola. Le istruzioni GOTO non dovrebbero mai essere usate per uscire da cicli FOR...NEXT (il valore di alcune variabili potrebbe risultare imprevedibile) né per uscire da una subroutine (ciò potrebbe generare degli errori nel ritorno al programma chiamante), né per saltare ad una parte del programma molto lontana (ciò rende quasi impossibile la rilettura di un listato di programma).
La subroutine FIRSTM è molto semplice ed elementare: viene "ripulito" lo schermo e visualizzato un messaggio che invita ad immettere il primo nominativo. La linea 3870 assegna il valore 6 alla variabile SCEL, cosicché, quando l'esecuzione giunge a EXECUT, viene automaticamente chiamata la subroutine AGGREC. Ecco FIRSTM:

```
3860 REM SUBROUTINE FIRSTM
3870 LET SCEL = 6
3880 PRINT CHR$(12) : REM PULIZIA SCHERMO
3890 PRINT
3900 PRINT TAB(8);"NON ESISTE ALCUN RECORD NEL"
3910 PRINT TAB(8);"FILE. OCCORRE IMMETTERE IL"
3920 PRINT TAB(8);"PRIMO NOMINATIVO"
3930 PRINT
3940 PRINT TAB(5);"(PREMERE UNO SPAZIO PER CONTINUARE)"
3950 FOR B = 1 TO 1
3960 IF INKEY$ <> " " THEN B = 0
3970 NEXT B
3980 PRINT CHR$(12): REM PULIZIA SCHERMO
3990 RETURN
```

Occorre apportare due piccole ma importanti modifiche alla routine AGGREC presentata precedentemente: la nuova versione è a pagina 379. Dopo aver trasferito il contenuto dei campi nelle rispettive matrici, viene incrementato il valore della variabile DIMENSIONE. Inoltre (vedere linee 10090 e 10100), il contenuto di TEST$ viene posto uguale ad una stringa nulla. La variabile DIMENSIONE è molto importante poiché viene usata in più parti del programma per stabilire il numero del record sul quale si sta operando. Inizialmente, DIMENSIONE è posta uguale a 0, nella subroutine CREMAT. Successivamente, in SETFLG, ad essa viene assegnato il valore 1, qualora TEST$ contenga "@PRIMO", consentendo alla subroutine AGGREC l'immissione del primo record del file.
La linea 10090 incrementa il valore di DIMENSIONE, cosicché, se e quando viene eseguita nuovamente un'immissione di dati, il numero di record sarà quello giusto. Infine, AGGREC assegna una stringa nulla a TEST$, in modo da avvertire il programma che un nuovo elemento è stato inserito (anche se non ancora memorizzato nel file) e che, nell'eseguire nuovamente CHOOSE, l'esecuzione non dovrà saltare automaticamente alla subroutine per l'inserimento. Infatti, se TEST$ non venisse azzerata, il programma entrerebbe in un ciclo senza fine, ripetendo continuamente l'immissione del primo elemento e l'unico modo per uscirne sarebbe spegnere il computer (con conseguente perdita di dati).
Invece, il controllo eseguito alle linee 3520 e 3530 permetterà la regolare presentazione del menu delle opzioni. Cosa accade del valore di DIMENSIONE dipende dall'opzione selezionata. Fin qui, abbiamo visto che DIMENSIONE contiene il valore 1 se il file è inizialmente vuoto e che questo valore viene incrementato ad ogni nuova immissione, ma se nel file fossero già presenti alcuni nominativi? Per rispondere, dobbiamo riesaminare la subroutine RDINFL.
La linea 1420 legge il primo elemento del file in TEST$. Se questo non è uguale a "@PRIMO", allora si tratta di un nominativo valido. I record del file hanno sempre il medesimo ordine, ossia: CAMNOM, CAMMOD, CAMVIA, CAMLOC, CAMNAZ, CAMTEL, CAMIND, CAMNOM, CAMMOD, ecc. Pertanto, occorre trasferire il record nei rispettivi elementi delle varie matrici e ciò avviene dalla linea 1440 in poi. Fatto ciò, il valore di DIMENSIONE può aumentare a 2 (ossia il numero del "prossimo" record da trasferire).
A questo punto, un ciclo con indice che varia da 2 a 50 trasferisce i restanti record nelle rispettive matrici. Si ricordi che abbiamo fissato in 50 il numero massimo di nominativi e le istruzioni DIM rispecchiano questo limite. Tuttavia, sarà piuttosto improbabile che il file, almeno le prime volte, contenga tutti e 50 gli elementi, cosicché occorre stabilire, in qualche modo, quando abbiamo esaurito gli elementi da trasferire, sia per attribuire un corretto valore a DIMENSIONE, sia per interrompere la fase di lettura.
Ecco perché abbiamo previsto una chiamata ad una subroutine DIMEN alla linea 1510, che svilupperemo più avanti. Possiamo scegliere fra tre soluzioni. Potremmo utilizzare il primo record del file per contenere il valore DIMENSIONE. La subroutine RDINFL potrebbe quindi esser modificata per leggere questo valore ed usarlo come limite massimo in un ciclo di lettura. Il secondo metodo, senz'altro migliore perché non interferisce con la verifi-

ca sul contenuto di TEST$ (linea 1430), consiste nell'accodare al file, una volta scritti tutti i record, un ultimo elemento di chiusura (ad esempio una stringa "@FINE"). In questo modo potremmo modificare RDINFL per stabilire, mediante un confronto, se e quando è stato letto l'ultimo record.
Il terzo metodo, infine, consiste nell'impiegare il valore di una speciale variabile EOF (End Of File = fine del file), presente in alcune versioni di BASIC.
Normalmente, il valore di EOF è 0: esso diviene 1 (o comunque un valore diverso da 0) quando, durante la lettura di un file, si giunge alla fine di esso. Potremmo allora inserire un controllo del tipo:

```
FINCHÉ EOF(N) = 0 (N = canale del file)
  LEGGI UN RECORD DAL FILE
FINE
```

In alcune versioni di BASIC, pur non esistendo EOF, è possibile operare un controllo equivalente esaminando il contenuto di una particolare locazione di memoria. Ad ogni buon conto, considerato che quest'ultimo metodo non è applicabile in generale, ne evitiamo l'uso.
Al solo scopo di esercitarsi, a seconda del BASIC impiegato, si potrebbe modificare la subroutine RDINFL per sperimentare i tre metodi descritti.
In linea generale, trattare file di lunghezza fissa semplifica la stesura di un programma, ma, se la possibilità di elaborare archivi "dinamici" (ossia di estensione variabile) viene prevista fin dalle prime fasi di analisi, sarà semplice apportare qualche modifica, in seguito, per aumentare le prestazioni del programma.

```
4000 REM SUBROUTINE EXECUT
4010 REM
4019 IF SCEL = 6 THEN GOSUB 10000 : REM VEDERE TESTO
4020 REM NORMALMENTE ON SCEL GOSUB...
4030 REM
4040 REM 1 —> FNDREC
4050 REM 2 —> FNDNMS
4060 REM 3 —> FNDTWN
4070 REM 4 —> FNDINT
4080 REM 5 —> LSTREC
4090 REM 6 —> AGGREC
4100 REM 7 —> MODREC
4110 REM 8 —> DELREC
4120 REM 9 —> EXPROG
```

La subroutine EXECUT, normalmente, non conterrebbe la linea 4019, e la linea 4020 sarebbe:

```
ON SCEL GOSUB numero linea, numero linea, ...
```

oppure sarebbe presente una serie di linee:

```
IF SCEL = 1 THEN GOSUB (numero linea)
IF SCEL = 2 THEN GOSUB (numero linea) ecc.
```

La linea 4019 è stata inserita col solo scopo di usare la subroutine in sede di verifica, anche se non tutti i vari moduli che la compongono sono stati codificati.

```
10 REM 'MAINPG'
20 REM * INITIL *
30 GOSUB 1000
40 REM * GREETS *
50 GOSUB 3000
60 REM * CHOOSE *
70 GOSUB 3500
80 REM * EXECUT *
90 GOSUB 4000
100 END

1000 REM SUBROUTINE * INITIL *
1010 GOSUB 1100: REM SUBROUTINE * CREMAT * (CREAZ. MATRICI)
1020 GOSUB 1300: REM SUBROUTINE * RDINFL * (LETTURA FILE)
1030 GOSUB 1320: REM SUBROUTINE * SETFLG * (ASSEGNA FLAG)
1040 REM
1050 REM
1060 REM
1070 REM
1080 REM
1090 RETURN

1100 REM SUBROUTINE * CREMAT * (CREAZIONE MATRICI)
1110 DIM CAMNOM$(50)
1120 DIM CAMMOD$(50)
1130 DIM CAMVIA$(50)
1140 DIM CAMLOC$(50)
1150 DIM CAMNAZ$(50)
1160 DIM CAMTEL$(50)
1170 DIM CAMIND$(50)
1180 REM
1190 REM
1200 REM
1210 LET DIMENSIONE = 0
1220 LET RMOD = 0
1230 LET SVED = 0
1240 LET CORR = 0
1250 REM
1260 REM
1270 REM
1280 REM
1290 REM
1300 RETURN

10000 REM SUBROUTINE * AGGREC *
10010 PRINT CHR$(12) : REM PULIZIA SCHERMO
10020 INPUT "NOME ";CAMNOM$(DIMENSIONE)
10030 INPUT "INDIRIZZO ";CAMVIA$(DIMENSIONE)
10040 INPUT "CITTA' ";CAMLOC$(DIMENSIONE)
10050 INPUT "NAZIONE ";CAMNAZ$(DIMENSIONE)
10060 INPUT "N. TELEFONICO ";CAMTEL$(DIMENSIONE)
10070 LET RMOD = 1 : REM FLAG 'RECORD IMMESSO'
10080 LET CAMIND$(DIMENSIONE) = STR$(DIMENSIONE)
10090 LET DIMENSIONE = DIMENSIONE + 1
10100 LET TEST$ = ""
10110 REM INSERIRE CHIAMATA A * MODNOM *
10120 REM
10130 REM
10140 REM
10150 RETURN
```

Basic & Basic

SPECTRUM

Poiché lo Spectrum non dispone di istruzioni per la memorizzazione di intere matrici, occorre impiegare la forma SAVE-DATA, illustrata a pagina 318. Di conseguenza, la subroutine RDINFL sarà completamente diversa. Una versione completa per questo apparecchio viene presentata più avanti nel corso. Ma i possessori di Spectrum potrebbero esercitarsi nel tentativo di risolvere il problema di memorizzare i dati e di stabilire quanti elementi sono contenuti nel file.

INKEY$

Si veda "Basic & Basic" a pagina 175.

OPEN CLOSE

Si veda "Basic & Basic" a pagina 319.

Powers e Hollerith

Una delle rivalità nel mondo dei computer



James Powers

Gli apparecchi di Powers erano puramente meccanici ed adatti ad una sola applicazione. Nonostante ciò, la concorrenza nei confronti delle macchine di Hollerith fu enorme



Herman Hollerith

Hollerith inventò i lettori elettromeccanici di schede, in seguito divenuti apparecchi tabulatori

per concessione della I.B.M.

Le macchine inventate da Herman Hollerith (vedere a pag. 240) per elaborare i risultati del censimento americano del 1890 furono la base di macchine per l'elaborazione di dati più generici, chiamate "tabulatori". Fino a quando non vennero introdotti i primi computer ad uso commerciale, negli anni '50, i tabulatori erano strumenti indispensabili per lo sviluppo dell'industria e del commercio. Per esempio, un grande magazzino di Pittsburgh, negli anni '30, sperimentò un sistema di fatturazione automatizzato, nel quale 250 terminali erano collegati, per via telefonica, con un tabulatore centrale. La vendita di ogni articolo, il cui prezzo e codice erano riportati su di una scheda, veniva automaticamente registrata consentendo l'emissione della fattura. La conferma dell'operazione avveniva mediante una sorta di telescrivente.

Lo stimolo principale, nello sviluppo dei tabulatori, provenne proprio dal settore commerciale. Il monopolio di Hollerith venne a cessare quando, nel 1910, l'ufficio del censimento affidò a James Powers l'incarico di progettare macchinari alternativi. Powers offrì un sistema di tabulatori completamente meccanici, evitando così d'interferire con i brevetti di Hollerith per macchine elettromeccaniche. La rivalità tra questi due personaggi (e tra le ditte che essi fondarono) perdurò nel tempo con l'evolversi degli apparecchi per l'elaborazione dei dati.

Nel 1902 Hollerith progettò uno speciale pannello di connessioni (simile a quello impiegato nei vecchi centralini telefonici) che permetteva di selezionare quali delle colonne di una scheda dovessero essere sommate e visualizzate. Egli introdusse, quindi, una prima possibilità di programmazione, totalmente estranea agli apparecchi meccanici di Powers, concepiti, al contrario, per una sola applicazione alla volta. Nel 1924 Powers brevettò un sistema di rappresentazione numerica ed alfabetica su schede perforate e Hollerith reagì prontamente inventando quelle che sarebbero in seguito divenute le comuni schede ad 80 colonne. Le schede venivano lette mediante apposite spazzole metalliche, collegate ai circuiti dell'apparecchio.

I primi tabulatori erano soltanto capaci di eseguire somme sui dati immessi ed accumulare i totali, ma ben presto vennero introdotte anche altre funzioni più complesse. Diversamente dai computer, che venivano allora progettati quasi esclusivamente per applicazioni scientifiche, i tabulatori erano principalmente indirizzati all'elaborazione d'informazioni. Nacquero così tabulatori per le più svariate applicazioni: dal calcolo di complesse tabelle numeriche all'analisi delle onde e, addirittura, all'astronomia (il pianeta Plutone venne scoperto grazie ai calcoli fatti su quei tabulatori). La IBM brevettò, tra l'altro, un tabulatore in grado di gestire oltre 10.000 conti correnti bancari.

I TABULATORI

I tabulatori, nel loro periodo di maggior successo (agli inizi degli anni '50), consistevano di otto unità separate. I dati venivano immessi mediante un perforatore di schede, in grado di elaborare 200 schede all'ora. Un altro apparecchio eseguiva una verifica dei dati immessi dall'operatore e le schede logorate venivano duplicate da un altro apparecchio. Un altro meccanismo, infine, provvedeva a stampare un messaggio esplicativo sui dati contenuti in ciascuna colonna.

Il tabulatore vero e proprio accumulava i totali delle somme relative a ciascuna colonna di dati, ed era capace di elaborare circa 9.000 schede all'ora. Spesso esso era collegato ad un "moltiplicatore", che consentiva di eseguire funzioni più complesse. Una speciale unità di confronto (il "collator") permetteva di confrontare i dati di due diverse pile di schede o di fondere in un unico gruppo due pile di schede. L'apparecchio riordinatore, inoltre, consentiva di riordinare e selezionare le schede, separandole in 13 diverse pile.

Le diverse operazioni del tabulatore venivano selezionate mediante appositi codici, perforati su speciali schede (colorate vivacemente, per riconoscerle con facilità). In tal modo, poteva essere variata, ad esempio, la colonna sulla quale effettuare la somma ed, automaticamente, veniva stampato il risultato intermedio raggiunto. Nei censimenti, questa capacità permetteva, ad esempio, di conteggiare separatamente il numero di abitazioni in una particolare zona del territorio. Alcune delle caratteristiche tipiche dell'elaborazione dati coi tabulatori vennero adottate nei primi linguaggi per computer



BBC HULTON PICTURE LIBRARY

Fantascienza e realtà

Gli scrittori di fantascienza hanno spesso profetizzato nuove tecnologie



© COLUMBIA WARNER

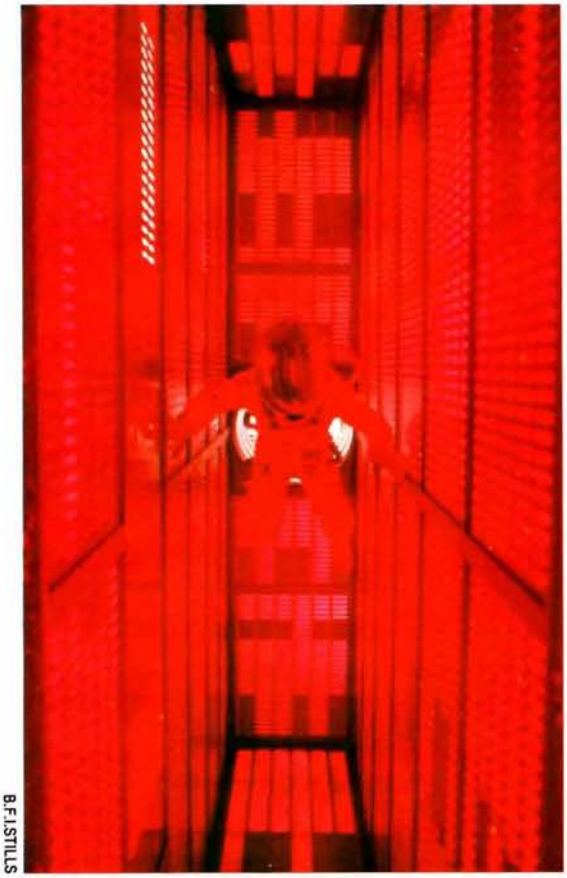
Superman III
Un computer truffatore è il coprotagonista del terzo film della serie Superman. Richard Pryor interpreta la parte di un furfante che si costruisce una fortuna, rubando mezzo "cent" per ogni transazione effettuata da un computer di una banca. Questa parte della trama prende spunto da alcuni casi di frode realmente accaduti. Il film si conclude con la distruzione del più grande computer del mondo, costruito con fini decisamente criminali

Molte conquiste tecniche e scientifiche sono state immaginate da scrittori di fantascienza o da registi di film molto prima che esse fossero effettivamente realizzabili. Arthur C. Clarke, l'autore di *2001 Odissea nello spazio*, per primo propose l'idea di un satellite geostazionario in un articolo pubblicato dalla rivista *Wireless World* all'inizio del 1950, vent'anni prima della sua concreta realizzazione. Similmente, il breve racconto *Waldo*, scritto da Heinlein, descriveva dei manipolatori controllati a distanza molto prima della progettazione di mani robotizzate. In effetti molti inventori e scienziati sono stati ispirati dalla creatività degli scrittori di fantascienza e dai registi di film.

Tuttavia, i computer della fantascienza molto spesso hanno scarsa attinenza con la realtà: nel futuristico film *Rollerball*, ad esempio, un computer capace di riconoscere le voci e di parlare, ha l'aspetto di un recipiente cubico pieno di un misterioso liquido. Computer più realistici tendono ovviamente ad essere meno emozionanti e poco interessanti anche se spesso sono stati usati nei film come arredi scenografici. Non c'è dubbio che i film prodotti negli anni '60 e '70, che presentavano computer del tutto simili ad autentici elaboratori, hanno contribuito ad educare la gente mostrando che cosa in realtà fossero queste nuove e quasi mitiche macchine.

Gli scrittori dotati di una fertile immaginazione cominciarono a fantasticare sui computer poco tempo dopo lo sviluppo, da parte di Charles Babbage (vedere a pag. 220) della pionieristica Macchina Analitica, intorno alla metà del diciannovesimo secolo. Nel 1879, Edward Page Mitchell scrisse un racconto intitolato *L'uomo più intelligente del mondo*, nel quale descriveva come una macchina calcolatrice fosse stata impiantata nel cervello di un uomo non particolarmente dotato, trasformandolo in un genio. Sotto vari punti di vista le idee di Mitchell hanno precorso progressi scientifici poi verificatisi. Innanzi tutto egli mise in risalto l'idea della miniaturizzazione: la macchina calcolatrice presa in considerazione era abbastanza piccola da trovar posto nel cranio dell'"incapace" e tuttavia era così potente da dotarlo di un'intelligenza superiore. Inoltre egli formulò l'ipotesi di collegare un computer ad un corpo umano. Soltanto oggi, a distanza di circa un secolo dalla pubblicazione di quel racconto, cominciano a venir perfezionate le tecniche necessarie per collegare semplici

apparecchi elettromeccanici al sistema nervoso centrale di un essere vivente.
Di solito, pochi scrittori hanno una conoscenza approfondita della struttura di un computer, anche se molti di essi sono ingegneri di notevole esperienza ed alcuni usano computer nel proprio lavoro. Nonostante ciò, molti scrittori hanno dimostrato di saper descrivere in modo convincente un viaggio intergalattico pur senza possedere approfondite conoscenze d'astrofisica o di missilistica. Inoltre non c'è ragione per cui gli scrittori non possano fare ipotesi sulle possibili caratteristiche delle future generazioni di computer pur senza conoscere bene le macchine oggi in uso.



B.F.I.STILLS

Pazzia spaziale

Il computer HAL, protagonista del romanzo di Arthur Clarke "2001 Odissea nello spazio", è un ottimo esempio dei supercomputer onnipotenti che spesso compaiono nei film di fantascienza. Nel racconto, HAL è l'unico a conoscere i veri obiettivi della missione spaziale, ignoti persino agli stessi membri dell'equipaggio. Seguendo una propria logica, la macchina si autoconvince di poter fare a meno degli uomini. Sembra certo che Clarke abbia scelto il nome HAL perché le lettere che lo compongono precedono nell'alfabeto quelle della sigla industriale IBM

Queste ipotesi hanno, per così dire, dato vita ad un computer fantascientifico standard che presenta, rispetto ai modelli attuali, una serie di caratteristiche del tutto irreali. Innanzi tutto, nella memoria di questo ideale computer è generalmente archiviato tutto lo scibile umano e qualsiasi informazione può essere istantaneamente richiamata, seguendo processi analoghi a quelli del cervello umano. Il computer HAL, nel film *2001 Odissea nello spazio*, è un tipico esempio di queste macchine superintelligenti.
Inoltre, il supercomputer più comunemente descritto dagli scrittori di fantascienza è onnipresente, anche se ciascun individuo ha l'impressione di essere il solo a dialogare con esso in quel momento. L'uso della voce come output (senza che nulla faccia pensare ad un prodotto sintetico costituito da fonemi posti in sequenza) ed il riconoscimento delle voci (che infallibilmente permette di riconoscere l'interlocutore dalle sue specifiche caratteristiche vocali), sono entrambi requisiti essenziali di questa supermacchina. Il riconoscimento visivo di un oggetto e la capacità di sintetizzare cibo (a partire dai suoi elementi chimici di base) rappresentano altre comuni caratteristiche.
Il supercomputer ora descritto possiede di solito una psicologia tipicamente umana e questa caratteristica lo fa apparire come un essere superiore. Tuttavia la "psiche" del computer può talvolta divenire decisamente crudele e pericolosa. Nel film *Dark star*, ad esempio, una bomba controllata da un computer aveva la psicologia di un killer psicopatico.
Inteso in questi termini, il supercomputer appartiene certamente al regno della fantasia. Tuttavia, alcune delle caratteristiche ora descritte si possono già trovare in certi moderni sistemi d'elaborazione. Sono già disponibili memorie di grande capacità e con rapidi tempi d'accesso; all'inizio degli anni '80 sono state prodotte memorie "gigabyte" (capaci di 1000 milioni di byte) e le più veloci macchine commerciali sono in grado di elaborare informazioni a velocità superiori a 10 milioni di informazioni per secondo. Inoltre, nel campo dell'output parlato, è stata quasi raggiunta la perfezione esibita nei film, dove i computer dialogano coi loro operatori. La qualità dell'output parlato dipende quasi esclusivamente dalla quantità di memoria disponibile, dalla velocità d'elaborazione e dai tempi di programmazione.
Il riconoscimento della voce è invece difficile da ottenere perché ciascun individuo è dotato di caratteristiche vocali assolutamente uniche. Due persone possono parlare la stessa lingua, ma il computer avrà l'impressione che esse parlino linguaggi completamente diversi.
Il riconoscimento visivo degli oggetti compie in questi anni i suoi primi passi, ma la tecnologia è in rapido sviluppo. Quando abbiamo parlato dei robot industriali (vedere a pag. 281), abbiamo visto che i maggiori progressi nel riconoscimento degli oggetti sono stati conseguiti utilizzando speciali telecamere e che un robot è oggi capace di scegliere uno specifico articolo tra oggetti di vario tipo. L'efficacia di un sistema di riconoscimento visivo dipende direttamente dalle dimensioni del "vocabolario visivo", a sua volta funzione dell'ampiezza di memoria e del potere d'elaborazione. Per quanto riguarda la sintesi del cibo, non si può preparare un piatto che abbia l'aspetto esteriore della carne, delle patate o del pesce fritto, ma è certamente possibile riprodurre il gusto e l'odore di questi alimenti, anche se i computer non hanno ancora la capacità di sintetizzarli partendo dagli elementi chimici elementari.
Non tutti gli autori hanno attribuito poteri fenomenali alle loro macchine fantascientifiche. Ad esempio, nel racconto *Stand on Zanzibar*, pubblicato nel 1969, John Brunner de-



© BBC STILLS

La cabina del Dottor Who

In parte veicolo ed in parte intelligenza artificiale, il "Tandis" del Dottor "Who" è una macchina fantascientifica che a buon diritto va annoverata tra quei computer di cui abbiamo parlato. Le domande le vengono rivolte attraverso una tastiera ed un'unità video; le sue capacità intellettive e le informazioni contenute nella sua memoria sembrano essere illimitate

scrive l'ipotetico mondo dell'anno 2010 nel quale i problemi della sovrappopolazione e della fame hanno ormai raggiunto livelli critici. Il computer che egli ci presenta, chiamato "Shalmaneser", possiede grande capacità di memoria e notevole velocità d'elaborazione (essendo collegato a tutti gli apparecchi televisivi della terra), ma il dialogo con esso assomiglia a quello adoperato oggi con alcuni computer:

PROGRAMMA RIFIUTATO
Q ragione per rifiuto
ANOMALIE IN RILEVAMENTO DATI SUOLO
Q definire Q specificare
DATI IN SUCCESSIVE CATEGORIE INACCETTABILI.
CULTURA INTERAZIONE SOCIALE STORIA COMMERCIO
Q accettare informazioni ricevute
DOMANDA SENZA SIGNIFICATO ED INATTUABILE

In questo caso, Brunner non ha fatto altro che rielaborare, ovviamente in chiave futuristica, il linguaggio comune, applicandolo ad una macchina. Anche altre sue ipotesi sono realistiche e non sorprende che la novella abbia vinto i più importanti premi per la letteratura fantascientifica.
Nei film e nei racconti più recenti i computer non svolgono più la sola funzione d'arredo scenografico o anche di "personaggi" secondari ma, molto spesso, interpretano un ruolo fondamentale nell'intreccio del racconto stesso. Un esempio di questo tipo si ha nel film *Tron* di Walt Disney. Accenniamo a questo eccezionale film (il cui titolo deriva da una funzione mnemonica, chiamata TRace-ON) perché è un esempio di animazione computerizzata (vedere a pag. 181).
L'azione si sviluppa sia nel mondo reale sia all'interno di un computer. I personaggi della realtà sono ingegneri di software, programmatori di sistemi ed altre persone del mondo dei computer, ma, all'interno della macchina, le singole parti del programma e del sistema operativo si trasformano in personaggi e l'architettura interna della macchina diviene lo scenario nel quale si sviluppa l'azione.
In altri racconti i computer non compaiono come protagonisti, ma non c'è dubbio che, senza la presenza di superelaboratori molto potenti, la trama di certi racconti non potrebbe reggere. *1984* di George Orwell e *Brave new world* di Aldous Huxley sono racconti di questo tipo. Entrambi descrivono un mondo futuro, dominato da un ristretto gruppo di potere, che opprime tutta la popolazione. Indubbiamente, il successo ottenuto da questi due racconti è dovuto ai timori del grande pubblico sui possibili abusi di potere che, potenzialmente, il computer può favorire.
Una completa analisi della letteratura fantascientifica, in particolare per quanto concerne la rappresentazione dei computer, non è possibile in questa sede, ma va ricordato un ultimo racconto, nel quale vengono presentate delle ipotesi veramente singolari. Si tratta di *Siles Goat-Boy* di John Barth: un lungo romanzo, di oltre 800 pagine, incentrato sull'idea che il suo autore è un supercomputer chiamato WESCAC. La trama narra di un incidente capitato, appunto, a questo computer "scrittore".
Occorre, infine, ricordare che non è stata la sola letteratura fantascientifica a descrivere in modo creativo il mondo dei computer. Tra le migliaia di libri specializzati non fantascientifici, scritti sui computer e sui problemi della computerizzazione, uno in particolare si distingue per la sua ottima qualità narrativa. Si tratta del libro *Soul of a new machine* (L'anima di una nuova macchina) di Tracy Kidden, nel quale si racconta la storia della nascita dell'Eagle, un microcomputer a 32 bit prodotto dalla Data General. Anche se, apparentemente, i protagonisti sono gli ingegneri ed i manager coinvolti nel progetto, il lettore avverte chiaramente che la "star" del romanzo è proprio il computer.

War Games

Un giovane utente di computer che cerca di comunicare con un altro computer attraverso la rete telefonica pubblica, penetra involontariamente nella memoria del computer principale dell'apparato di difesa NATO. Convinto di essersi collegato ad un nuovo gioco, egli inizia a giocare per scoprire soltanto in seguito di aver dato inizio alla Terza Guerra Mondiale



© UNITED ARTISTS

I generatori di programmi

Si possono scrivere programmi in grado di generare altri programmi o di correggere gli errori di programmazione

Sono in corso molte ricerche che hanno come obiettivo lo sviluppo di programmi capaci di generarne altri e di sistemi operativi in grado di correggere gli errori presenti nei codici scritti dagli uomini.
Per fare un esempio, lo scarno messaggio SYNTAX ERROR, che si incontra con una certa frequenza usando il BASIC negli home e personal computer, poiché si limita a segnalare un errore, senza fornire ulteriori informazioni utili. I compilatori, presenti su elaboratori di grandi dimensioni, forniscono invece maggiori indicazioni sulla natura dell'errore incontrato. Ad esempio, un messaggio d'errore potrebbe avere questa forma:

```
1090 LET A=(C*2+F$)*((FG—C)*TH+1))
                    ↑              ↑
ERRORE: 1) ACCOPPIAMENTO ERRATO, VARIABILE STRINGA
           F$ NON CONSENTITA
        2) ULTIMA PARENTESI CHIUSA NON PREVISTA
```

È comunque possibile acquistare ROM aggiuntive o cartucce di software che amplino la gamma di comandi BASIC disponibili, particolarmente quelli relativi allo sviluppo ed alla ricerca degli errori dei programmi. Questi comandi BASIC comprendono:

HELP – visualizza la linea di programma ed evidenzia con precisione il punto nel quale l'esecuzione del programma si è bloccata. Questa posizione spesso corrisponde ad un errore di sintassi.
DUMP – stampa una lista di tutti i nomi delle variabili e del loro contenuto. Questo comando è utile per capire a quale punto dell'elaborazione sia stato incontrato l'errore.
TRACE – visualizza in una "finestra", situata in un angolo dello schermo, il numero o i numeri di linea via via eseguiti. Questo comando permette all'utente di seguire il flusso del programma e permette di verificare, inoltre, che le subroutine vengano eseguite nella corretta sequenza.

La stesura di programmi che consentano ad un computer di individuare e di correggere errori di programmazione introdotti dall'uomo non è affatto semplice, anche se, per un certo tipo di errori, non esistono grandi difficoltà. Ad esempio, sappiamo che in BASIC tutte le linee di programma devono iniziare con una parola chiave (anche se qualche macchina permetterà di saltare la parola LET). Quindi se una linea comincia con PRUNT o PRONT, sarà facile dedurne che il comando corretto doveva essere PRINT. Nel Corso di Programmazione BASIC, abbiamo approfondito il concetto di "confronto approssimativo" (ossia di algoritmi che permettano di trovare l'insieme di parole che maggiormente somiglia ad una data frase): questo principio può essere applicato anche alle parole chiave di un programma. In alternativa, l'interprete potrebbe semplicemente utilizzare un elenco degli errori di battitura più comuni, assieme alla forma corretta equivalente. In questo caso, la validità di ogni eventuale sostituzione, suggerita dal computer, potrebbe esser confermata dall'operatore.
Tuttavia, per procedure di correzione più complesse, la correzione automatica presenta difficoltà molto maggiori. Nell'esempio prima mostrato, F$ è stato battuto al posto di F, FS o F4? Oppure si tratta di un errore di tutt'altro genere? Se facessimo esaminare il listato completo ad un programmatore esperto, egli dovrebbe essere in grado di individuare l'errore e di correggerlo. Il suo ragionamento si baserebbe su due fattori fondamentali: il contesto in cui appare la linea di programma e la sua personale esperienza.
Stranamente, questa tecnica è usata più per correggere testi, che per verificare codici di programmi. Gli "spelling checker" (programmi che controllano la corretta compitazione delle parole), ad esempio, analizzano un testo e mettono in evidenza tutte le parole non rintracciabili in un dizionario (spesso composto da oltre 50.000 parole), memorizzato su un disco. Molti di questi "pacchetti" di programmi hanno la capacità di apprendere nuove parole (ad esempio la sigla di un'azienda o un nome proprio) e di inserirle nel dizionario. I "pacchetti" più sofisticati offrono anche la possibilità di sostituire parole, scritte in modo errato, con il corretto equivalente. Sono stati progettati anche dei "word processor" in grado di applicare queste stesse procedure alla grammatica ed allo stile letterario: essi rilevano, per esempio, gli errori di punteggiatura o la ripetizione di parole in una breve frase oppure metafore poco chiare ed aggettivi usati in modo inadeguato. Programmi di questo tipo lavorano esaminando il contesto di ogni frase, confrontandolo con una "libreria" di frasi usate in precedenza.
Molto si è fatto anche per sviluppare sistemi capaci di generare nuovi programmi, piuttosto che correggere quelli esistenti.
Nel 1981 venne prodotto un pacchetto di software, che scatenò una delle più dure battaglie commerciali mai combattute tra le case produttrici di microcomputer. Questo pacchetto, astutamente chiamato "The Last One" (ossia: "L'Ultimo"), veniva presentato

come un sistema software in grado di creare qualsiasi tipo di programma di cui si potesse aver bisogno. In realtà, anche se il package è utile nello sviluppo di particolari tipi di programmi, soprattutto destinati ad applicazioni commerciali, la soluzione è ancora lontana dalle condizioni ideali. Esistono in commercio numerosi prodotti di questo genere per microcomputer commerciali e qualcuno anche per home computer: simili programmi sono chiamati "generatori di programmi".
Esaminiamo ora i principi teorici su cui si basano questi generatori di programmi. Consideriamo questo banale esempio:

```
10 PRINT "CHE COSA SI DESIDERA VISUALIZZARE SULLO
   SCHERMO?"
20 INPUT A$
30 PRINT "IL PROGRAMMA È:"
40 PRINT "10 PRINT";CHR$(34);A$;CHR$(34)
```

Se rispondiamo SALVE alla domanda, il programma (che dovrebbe essere in grado di funzionare sulla maggior parte degli home computer) visualizza quanto segue:

```
IL PROGRAMMA È
10 PRINT "SALVE"
```

Adottando una simile tecnica per l'input, il calcolo e l'output del programma che si desidera sviluppare, allora potremo scrivere da soli un semplice generatore di programmi. Se tutte le domande che il generatore rivolge all'utente sono espresse in modo chiaro, anche un operatore alle prime armi potrà sviluppare un semplice programma usando un generatore così progettato. I generatori di programmi disponibili in commercio usano proprio questo genere di procedure.
Molte applicazioni commerciali si basano su una combinazione di 5 diversi processi: l'input dei dati, l'output su video o su stampante, l'immagazzinamento di dati in un file, la ricerca ed il calcolo. Il generatore non fa altro che impiegare dei sottoprogrammi standard, ma molto adattabili, per ciascuna di queste procedure. Lo stesso generatore chiederà che vengano specificati l'esatta struttura dei dati, i calcoli da eseguire su questi dati ed i formati di visualizzazione desiderati sulla stampante o sul video.
Ottenute queste ed altre informazioni, il generatore cambierà i valori di alcune variabili nei sottoprogrammi e li combinerà assieme per creare un nuovo programma "su misura" per la particolare applicazione.
I generatori di programmi stanno diventando sempre più sofisticati, ma è improbabile che possano sostituire i programmatori umani nell'immediato futuro, in quanto possiedono ancora qualche limitazione. Innanzi tutto le tecniche descritte sono adatte ad applicazioni commerciali basate su transazioni, ad esempio la contabilità o la gestione delle scorte. Generalmente questi sistemi non possono essere usati per scrivere "word processor" o programmi di giochi. Secondariamente, dato che i generatori di programmi debbono far uso di sottoprogrammi standard flessibili, i listati prodotti non sono quasi mai ottimizzati (in termini di velocità d'esecuzione e di uso della memoria) quanto i programmi scritti da un esperto programmatore. Inoltre, i programmi creati dai generatori non sono di solito così interattivi e facili all'impiego, quanto quelli sviluppati in maniera tradizionale. Ad esempio, essi raramente sfruttano appieno le capacità grafiche dei computer.
Infine, i generatori di programmi oggi in commercio possono sostituirsi all'uomo solo nella fase finale del lavoro di programmazione, ossia nella effettiva codifica delle istruzioni: l'utente deve ancora svolgere una notevole mole di lavoro per definire l'esatto formato dei dati, l'input e l'output necessari. Senza dubbio le fasi iniziali della programmazione sono le più difficili ed hanno bisogno di competenze specifiche, diverse da quelle necessarie per la stesura fisica dei programmi. Molte grandi industrie impiegano degli specialisti (gli analisti di sistema) per stendere un'accurata analisi di specifiche procedure, che viene successivamente trasposta, dai programmatori, in un particolare linguaggio di programmazione.

UNA SEMPLICE APPLICAZIONE



I programmi commerciali vengono generati specificando il contenuto di ciascun file (da tenere costantemente aggiornato) e la struttura di tutte le transazioni e delle stampe che dovranno essere elaborate. A questo punto, l'utente specifica le relazioni esistenti tra i vari gruppi di dati. Lo schema in figura mostra le fasi iniziali di un'analisi nella quale sono schematizzate le operazioni di un semplice sistema di contabilità

LIZ DIXON

Un arcobaleno di accessori

Alcuni computer sono superaccessoriati: lo Spectrum è uno di questi



Accoppiatore acustico
Il Micro-Myte 60, qui illustrato, permette la comunicazione tra due computer

I Microdrive
Il Microdrive utilizza cartucce che contengono un anello continuo di nastro magnetico, ciascun punto del quale passa sotto la testina di lettura/scrittura ogni 6 secondi. Il trasferimento delle informazioni avviene ad una velocità di 6 Kbyte per secondo (quattro volte superiore a quella ottenibile con un normale registratore a cassette) ed è possibile utilizzare fino ad otto Microdrive contemporaneamente (come nella foto) per una capacità complessiva di 700 e più Kbyte

Tastiera
La tastiera FDS, prodotta dalla Fuller, è dotata di tasti di funzione e di una barra spaziatrice di grandi dimensioni

Joystick
Utilizzando l'Interfaccia 2 della Sinclair, è possibile collegare una coppia di joystick qualsiasi, purché dotata di interfaccia Atari, indipendentemente dal suo principio di funzionamento

Presentato agli inizi del 1982, lo Spectrum della Sinclair si è immediatamente imposto come un prodotto dal prezzo molto competitivo. Dopo appena un anno di produzione, le vendite dello Spectrum (circa 600.000 unità) hanno rappresentato più del 50% del numero complessivo di home computer venduti in Gran Bretagna. Un simile risultato ha sorpreso persino i suoi fabbricanti!
Lo Spectrum costituisce un notevole perfezionamento del precedente computer della Sinclair, lo ZX81: infatti è dotato di 16 o 48 Kbyte di RAM come standard, di 8 colori per il margine del video, lo sfondo ed i testi, di discrete capacità grafiche in alta risoluzione, di una tastiera migliore (anche se di formato non standard) ed, infine, della possibilità di produrre semplici suoni. Tuttavia, tutte queste novità non hanno affatto impedito la produzione, da parte di fabbricanti indipendenti, di una vasta gamma di accessori. La stessa Sinclair ha seguito questa strada producendo sia memorie di massa, i Microdrive, sia una serie di interfacce per l'innesto di cartucce ROM e di joystick.

Il "RAM pack"
La versione di base dello Spectrum, a 16 Kbyte, può essere potenziata con l'aggiunta di una cartuccia di memoria RAM da 32 Kbyte



Monitor TV

Poiché lo Spectrum non prevede un'uscita su monitor, è difficile ottenere grafica di alta qualità. Qui è mostrato il Profeel della Sony, un apparecchio televisivo modulare con apparato di ricezione separato dal monitor

Stampante seriale

Oltre che alla stampante ZX, lo Spectrum può essere collegato a normali stampanti a matrice d'aghi o a margherita, utilizzando la porta RS232 o una scheda d'interfaccia Centronics. L'interfaccia 1, propria della Sinclair, prevede un collegamento di tipo seriale

Registratore a cassette

Lo Spectrum può tanto scrivere quanto leggere da un normale registratore a cassette, anche quando è collegato al Microdrive. Ciò permette di utilizzare senza problemi il software per lo Spectrum reperibile in commercio

Sintesi musicale

Il "Tricorde" della Petron è un ottimo esempio di sintetizzatore musicale utilizzabile con lo Spectrum. Questo apparecchio, come lo stesso nome suggerisce, è un'unità a tre voci e, abbinato allo Spectrum a 48 Kbyte, permette di suonare fino a 6134 accordi di tre note

Sintesi di voce

Lo "Sweet Talker" della Cheetah è un sintetizzatore basato su di un sistema allofonico semplificato per produrre la voce. Le parole vengono costruite combinando le sillabe che le compongono. Questo apparecchio usa 64 allofoni e 4 intervalli di tempo di diversa durata

La tastiera

La tastiera standard dello Spectrum è stata notevolmente migliorata rispetto a quelle dello ZX80 e dello ZX81, tuttavia non è l'ideale per chi sia abituato alle tastiere di tipo tradizionale

Interfaccia 1

Progettata appositamente per gestire i Microdrive, questa interfaccia possiede anche dei circuiti che permettono il collegamento a stampanti, a plotter oppure a modem. Essa consente, inoltre, di collegare un certo numero di Spectrum tra loro, per formare una rete locale

Interfaccia 2

Alcune case produttrici indipendenti hanno messo in vendita interfacce programmabili per joystick, ma l'apparecchio prodotto dalla stessa Sinclair è dotato di uno zoccolo per ROM, che permette di installare ed usare più facilmente linguaggi alternativi e giochi

Stampante ZX

La Sinclair produce una stampante a basso costo che risponde alle necessità di un'utenza domestica interessata soprattutto a stampare listati di programmi. Questa unità adotta un particolare meccanismo di abrasione termica che permette di stampare su una carta rivestita da una sottile lamina d'alluminio

I chip ULA

Questi circuiti integrati sono l'ideale complemento della CPU

Fra i molti progressi nella progettazione elettronica che sono derivati dal "boom" dei microcomputer, uno dei più significativi è stato lo sviluppo di un particolare chip chiamato "Uncommitted Logic Array" (o, più brevemente, ULA). Questa importante innovazione, quasi ignorata dal grande pubblico, rappresenta un notevole salto tecnologico, al punto che oggi è possibile costruire computer (o altri apparecchi) molto sofisticati utilizzando non più di quattro componenti principali: una CPU, alcune RAM, alcune ROM ed un chip ULA, che permette di collegare tutte queste unità tra loro.

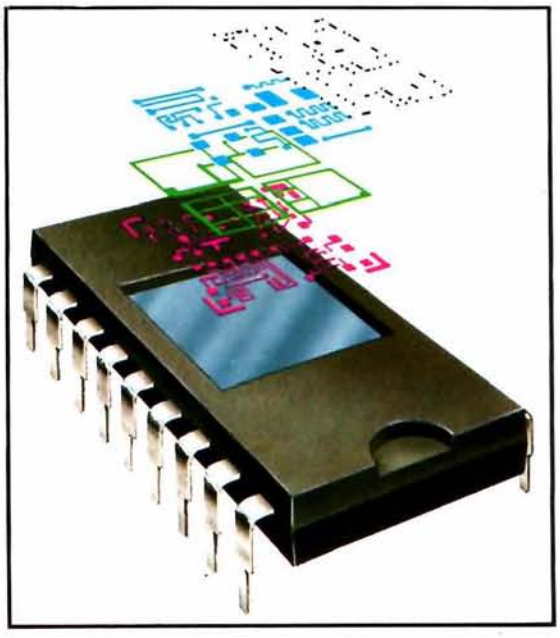
Ma che cos'è un Uncommitted Logic Array? Letteralmente, il nome sta a significare "matrice logica non vincolata" ed, in effetti, non si tratta che di un insieme di porte logiche, inizialmente indipendenti tra loro, ma che può essere opportunamente sfruttato per realizzare quasi tutte le operazioni che il progettista desideri. L'ULA può essere considerato uno sviluppo della ROM, in quanto, in entrambi i casi, il contenuto dei chip viene definito dal fabbricante e non dall'utente.

Una ROM o un ULA, prima di essere programmati, contengono soltanto un certo numero di semplici circuiti elettronici o "celle" non collegate tra loro e che non possono svolgere alcuna utile funzione. Tutti i chip vengono fabbricati sovrapponendo più strati di materiali semi-conduttori (vedere a pag. 122). L'ultimo strato è di solito composto di materiale conduttore e consente i collegamenti elettrici tra le varie celle. È la grande quantità di combinazioni, tra le varie connessioni possibili, che consente un'alquanto notevole flessibilità d'impiego degli ULA e, benché ogni singola cella del circuito sia piuttosto semplice (una coppia di transistor o soltanto una resistenza), più celle vengono collegate tra loro mediante il substrato conduttore per comporre circuiti molto più complessi, come un flip-flop (vedere a pag. 305). Circuiti di questo tipo, chiamati "moduli", utilizzano generalmente meno di 6 celle e poiché un ULA di notevoli dimensioni può contenere alcune migliaia di celle, i moduli stessi possono venir collegati tra loro per ottenere circuiti più complessi (registri, contatori, circuiti di temporizzazione, ecc.). Un ULA può essere programmato per l'esecuzione di una gamma di funzioni assai differenziate: dalla sintesi di suoni al controllo dei tempi di esposizione, della messa a fuoco e del motore di una macchina fotografica. Inoltre, un ULA non ha praticamente alcun bisogno di collegamenti elettrici esterni, fatta eccezione per una batteria, un interruttore e per alcuni sensori o pulsanti di controllo.

Come era logico attendersi, nella progettazione dei collegamenti tra celle che compongono un ULA viene largamente impiegato il computer: un mini computer, utilizzando uno speciale programma CAD (Computer Aided Design = progettazione assistita da computer), costruisce un diagramma, in codice, della struttura logica richiesta. Il sistema poi disegna e quindi codifica una mappa del sistema logico desiderato. Il risultato viene riprodotto mediante una stampante grafica ed, infine, un plotter provvede a tracciare il progetto definitivo. Quest'ultimo viene poi elaborato da un computer di maggiori dimensioni, che ne verifica la validità, lo confronta col progetto logico originale e controlla che non vi siano errori. Il progetto viene infine sottoposto all'analisi di un programma, fornito dal committente, che simula il suo funzionamento. A progetto ultimato, il computer fornisce la fotoincisione per la produzione dello strato finale di materiale conduttore.

È difficile prevedere quali sviluppi avranno in futuro gli ULA: l'idea di inserire un gruppo di semplici circuiti nel chip di silicio e di offrire all'utente la possibilità di definirne le reciproche interazioni, è così stimolante che potrebbe divenire il metodo più diffuso per costruire la maggior parte dei circuiti. Tuttavia, tenendo conto degli attuali livelli tecnologici, la produzione di ULA risulta economica solo quando si supera un quantitativo di almeno 2000 circuiti identici. La PROM (Memoria Programmabile di Sola Lettura), l'EPROM (una PROM riprogrammabile), l'EEPROM e l'EAPROM (PROM riprogrammabili mediante campo elettrico) sono tutte unità di memoria alternative alla ROM, che possono essere programmate o riprogrammate a piacimento dall'utente in possesso di un'adeguata apparecchiatura. Tra non molto, compariranno in commercio anche dei chip ULA capaci di una pari flessibilità d'impiego.

STEVE CROSS



Lo strato conduttore

Tutti i chip sono costituiti da strati di materiale semiconduttore, individualmente incisi mediante un procedimento fotochimico, che permette di modellare i vari elementi del circuito. Lo strato finale (conduttore) definisce le connessioni elettriche tra i vari elementi. Un ULA è composto da un insieme di elementi logici (porte), che si possono combinare tra loro per formare un circuito logico complesso

Tre giochi di simulazione

Questo genere di applicazioni è considerato molto educativo: vediamone qualche esempio

Venti

Questo programma di simulazione, prodotto dalla Longmans per il Micro della BBC, mette l'utente al comando di un veliero. Sul video appare una cartina dei cinque continenti ed il veliero, rappresentato da un piccolo punto luminoso, viene governato usando, come riferimento, i punti cardinali. L'utente sceglie il giorno di partenza, quello d'arrivo e la data del primo giorno di navigazione. La velocità della nave dipende dalla forza e dalla direzione del vento: questi dati vengono visualizzati nella parte inferiore del video con la posizione del veliero (espressa in latitudine e longitudine), il quadrante da cui spira il vento (ad esempio "da ovest"), la data, la rotta percorsa e la lunghezza del viaggio.
Consideriamo la rotta più diretta tra Londra e Rio de Janeiro e supponiamo di prendere il mare il primo di gennaio. Con rotta sud, avanziamo rapidamente, sino a raggiungere l'Equatore, sospinti dai venti atlantici che spirano da ponente; qui ci imbattiamo in tre giorni di bonaccia. Finalmente, comincia a soffiare un vento da sud-ovest, ma questo fatto crea un nuovo problema: come sarà possibile far rotta verso sud-ovest con un vento che soffia proprio da quella direzione? La soluzione corretta è "bordeggiare" in direzione sud ed ovest alternativamente sino a raggiungere Rio, dopo aver percorso 15.480 km. (9.620 miglia marine) in 207 giorni.
Altre rotte presentano pericoli maggiori: uragani, iceberg e naufragi sono solo alcuni dei rischi che occorre affrontare.

Simulazione di volo

Nella simulazione di volo, spesso presente in apparecchi da "sale giochi", il giocatore diviene il pilota di un piccolo aeroplano. Sulla parte superiore dello schermo viene riprodotto graficamente il paesaggio, mentre in quella inferiore è visualizzato il pannello degli strumenti di volo, comprendente numerosi quadranti, manometri e spie luminose: tutti strumenti da sorvegliare costantemente.
A differenza di quanto accade nelle versioni da "sala giochi", in questo caso è possibile familiarizzare con le procedure di controllo iniziando la simulazione con l'aereo già in volo. Per sapere dove il velivolo si trovi in un particolare momento, si può visualizzare una mappa aerea, completa di posizione dell'aereo, dei radiofari e delle piste d'atterraggio. La miglior procedura di navigazione consiste nell'affidarsi all'avvistamento automatico di un determinato radiofaro e quindi seguire una rotta d'avvicinamento sino a raggiungere un perfetto allineamento con esso. Il radiofaro è simboleggiato da un punto lampeggiante sul quadrante del radiogoniometro e si sposta sullo schermo fino a trovarsi al centro del quadrante. Raggiunta questa posizione, si dovrà volare "dritto di prua".
L'atterraggio è la manovra più difficile. È necessario essere perfettamente allineati alla pista ed avvicinarsi alla giusta velocità, altezza ed angolazione. Anche se tutti questi parametri sono corretti, è ancora possibile commettere un banale ma fatale errore: scordarsi d'abbassare i carrelli!

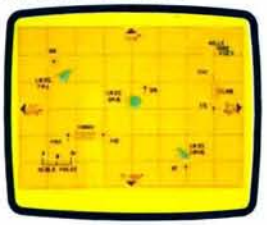
Simulazione fisiologica

Questo programma simula il cervello umano e l'attività che esso svolge nel controllare le varie funzioni corporee: lo scopo è quello di tenere in vita il "corpo" per un periodo di 50 minuti! La prima cosa da fare è inserire nel computer l'età ed il sesso della ipotetica persona, oltre all'attività che le si vuol far svolgere. Il sonno è la più semplice da simulare perché il corpo consuma una quantità minima d'energia. Altre attività possono essere: camminare, scalare una roccia e correre, che è la più dispendiosa in termini d'energia. I parametri da tenere sotto controllo sono il volume ed il ritmo respiratorio, il livello di sudorazione. È necessario quindi definire alcuni valori iniziali; ad esempio, un volume respiratorio di 2,5 litri, un ritmo respiratorio di 15 inspirazioni al minuto ed una sudorazione di 3 grammi al minuto.
Sul video appaiono 5 grafici, che si riferiscono alle varie funzioni corporee, ed un orologio. Col passare del tempo i grafici mostrano come il corpo risponda all'attività in corso e l'operatore può intervenire su ciascun grafico per evitare che venga superata la soglia di pericolo. Se, ad esempio, la temperatura corporea sale troppo, è possibile sospendere per breve tempo l'attività ed accrescere la sudorazione per ottenerne l'abbassamento. Se questo tentativo fallisce (e la traccia che rappresenta la temperatura oltrepassa il livello di guardia), sullo schermo può comparire la secca diagnosi: "IL SOGGETTO È MORTO". Questo programma è stato prodotto dalla Heinemann Educational Software per il Micro BBC Modello B.

VENTI



SIMULAZIONE DI VOLO



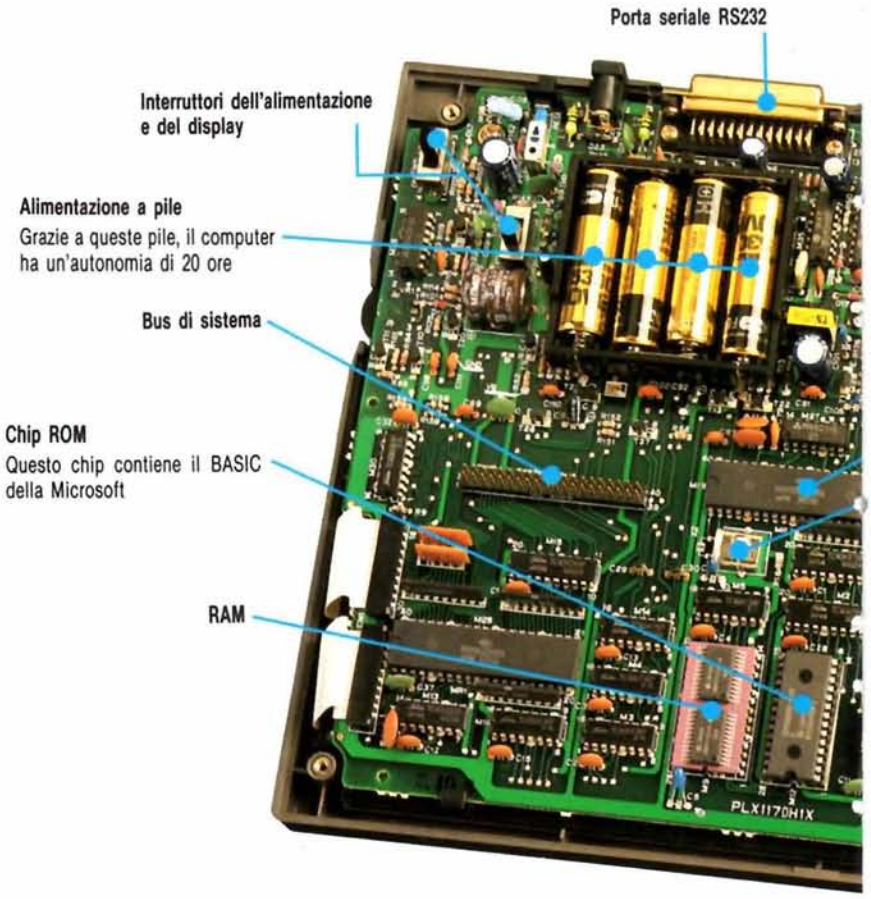
SIMULAZIONE FISIOLOGICA

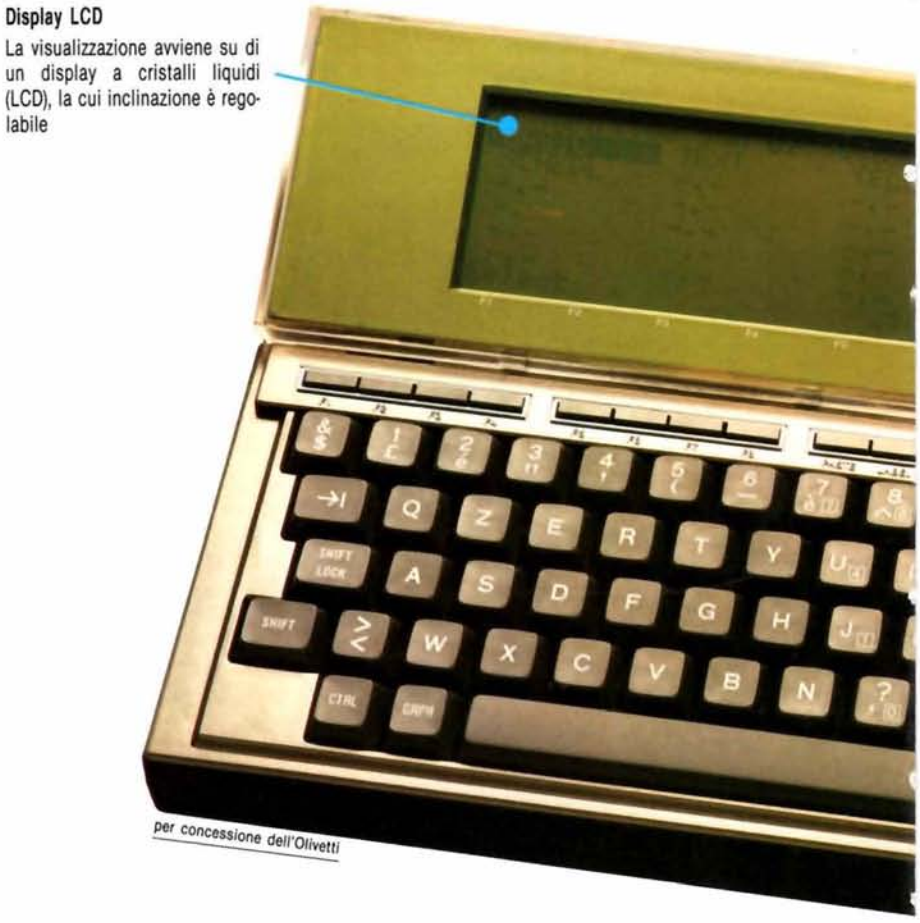


IAN McKINNELL

L'Olivetti M10

Anatomia di un microcomputer a diffusione europea: tecnologie a confronto





per concessione dell'Olivetti

L'Olivetti M10 appartiene alla stessa classe di portatili di cui fanno parte anche i molto simili Tandy Modello 100 ed Epson HX-20. In effetti, sia il Tandy che l'Olivetti sono stati commissionati alla medesima ditta giapponese, la Kyocera.

L'M10 si presenta come un'unità molto compatta, dalla linea assai elegante. La tastiera è di dimensioni standard e la qualità dei tasti è senz'altro ottima. La visualizzazione avviene su di un display a cristalli liquidi, la cui posizione (a differenza di altri portatili) è regolabile per ottenere la massima leggibilità anche in condizioni di luminosità non ottimali. Tra la tastiera ed il display sono situati 16 tasti (8 per funzioni speciali, 8 programmabili da utente). I principali pregi di questo apparecchio sono la possibilità di funzionare con alimentazione a batterie (quattro pile a stilo incorporate) e la capacità di non perdere il contenuto della RAM ad apparecchio spento (è previsto, infatti, un apposito circuito, alimentato autonomamente con pile al nichelcadmio).

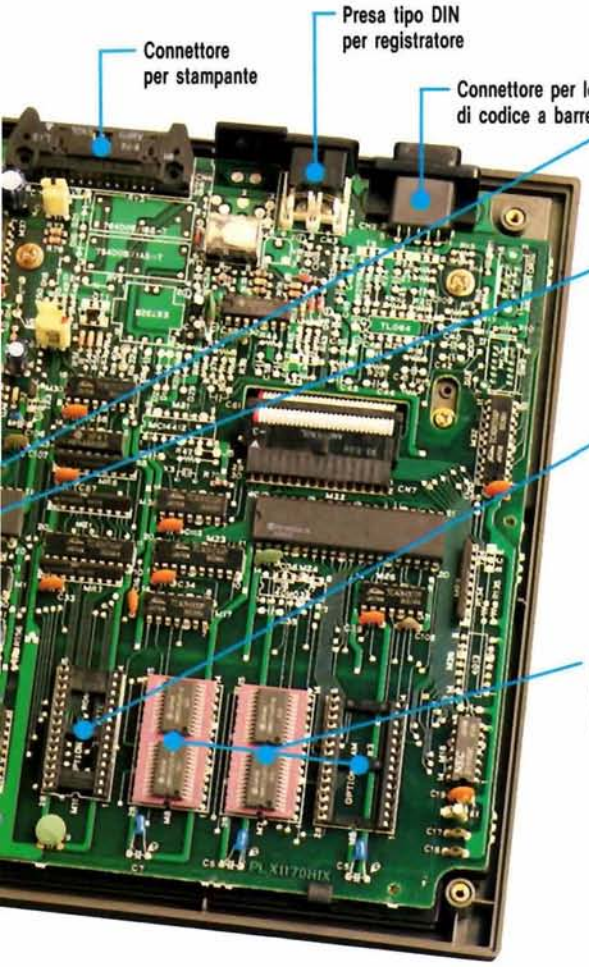
Queste caratteristiche, assieme al BASIC (versione Microsoft), ad un elementare "word processor" incorporato ed alla possibilità di comunicare con altri apparecchi mediante la porta seriale RS232, fanno dell'M10 uno strumento ideale per chi abbia la necessità di lavorare durante spostamenti da un luogo all'altro; l'autonomia si aggira sulle 20 ore.

Interessanti sono la possibilità di comporre immagini grafiche, nonostante le limitate dimensioni del display e quella di produrre note musicali mediante semplici comandi BASIC.

Oltre al software già citato, viene fornito un programma per la gestione di una rubrica elettronica ed una sorta di agenda computerizzata per appuntamenti.

Una caratteristica, non comune ai microcomputer di serie, è la presenza di un connettore per lettore di codice a barre, che può risultare utilissimo, in certe applicazioni, considerato il crescente diffondersi della codifica a barre nell'identificazione di prodotti commerciali.

La fascia più probabile di utilizzatori, per apparecchi di questo tipo, comprende: scrittori, giornalisti, rappresentanti e uomini d'affari in genere. Invece, per quanto concerne la didattica, l'Olivetti M10, pur offrendo prestazioni ragguardevoli, ha una ben nutrita concorrenza da parte di apparecchi più economi-





LA TASTIERA

La tastiera, sia nelle dimensioni che nella disposizione dei tasti, corrisponde più o meno a quella di una normale macchina da scrivere, facilitando l'immissione di lunghi brani di testo. Sono disponibili 16 tasti speciali, dei quali 8 servono per funzioni predefinite, mentre gli altri otto sono programmabili da utente. La disposizione dei tasti è secondo il formato italiano (QZERTY). I tasti sono realizzati in materiale antiriflesso e, per una maggiore comodità di battitura, ogni tasto è dotato di autoripetizione e di conferma sonora dell'avvenuta pressione. Nessun problema inoltre per dattilografi veloci: un apposito "buffer" di memoria evita la perdita di caratteri battuti molto velocemente

OLIVETTI M10

DIMENSIONI

mm. 300 x 210 x 50

PESO

gr. 1.700

CPU

OKI 80C85

CLOCK

2,4 MHz

MEMORIA

32 Kbyte di ROM (estendibile a 64 Kbyte)
8 Kbyte di RAM (estendibile a 32 Kbyte)

VIDEO

Display a cristalli liquidi con 8 linee di 40 caratteri, o 64 x 240 punti per la grafica. Posizione regolabile

INTERFACCE

Connettore per registratore a cassette, connettore seriale RS232C, connettore per lettore di codice a barre, uscita per stampante parallela

LINGUAGGI FORNITI

BASIC versione Microsoft

ALTRI LINGUAGGI DISPONIBILI

—

ACCESSORI COMPRESI

Manuale d'uso

TASTIERA

Formato QZERTY, 8 tasti funzione più altri 8 tasti speciali

DOCUMENTAZIONE

Il fatto che la documentazione sia in italiano è, indubbiamente, un pregio non indifferente rispetto a quella fornita dalla maggior parte degli altri apparecchi prodotti all'estero. L'uso del BASIC viene ampiamente illustrato, istruzione per istruzione, in modo sufficientemente chiaro anche per i meno esperti.

ci, dotati di maggiori capacità, sia grafiche che musicali, e di una più ampia gamma di linguaggi.
L'Olivetti ha già realizzato una nuova versione di computer portatile il cui progetto è parzialmente derivato dall'apparecchio in esame, dimostrando la tendenza, anche nelle grandi case costruttrici, ad orientarsi verso computer sempre più piccoli e versatili.

Suoni spettacolari

L'Oric-1 offre discrete possibilità nel controllo degli effetti sonori

I comandi MUSIC e PLAY

Il seguente programma utilizza i comandi MUSIC e PLAY per suonare un accordo di do maggiore (do, mi e sol) usando, a rotazione, i vari tipi d'inviluppo:

```
10 REM ****************
20 REM *ACCORDO*
30 REM ****************
40 MUSIC 1,4,1,0 : REM DO
50 MUSIC 2,3,8,0 : REM MI
60 MUSIC 3,3,5,0 : REM SOL
70 FOR E = 1 TO 7 : REM
   *SELEZIONA INVILUP-
   PO*
80 PLAY 7,0,E,750 : REM
   *SUONA L'ACCORDO*
90 PLAY 0,0,0,0 : REM *CHIU-
   DE L'ACCORDO*
100 WAIT 50 : REM *PAUSA*
110 NEXT E : REM *INVILUP-
   PO SEGUENTE*
```

L'Oric-1 è dotato di una notevole capacità di generare effetti audio. L'apparecchio offre un'estensione musicale di 7 ottave ed impiega 3 oscillatori, un generatore di rumori e 7 livelli d'inviluppo (vedere a pag. 276), selezionabili per modificare i suoni, che vengono riprodotti mediante l'altoparlante del televisore al quale il computer è collegato.
Il BASIC dell'Oric-1 mette a disposizione un gruppo di comandi relativi ai suoni: ZAP, PING, SHOOT ed EXPLODE. Il seguente esempio chiarisce come essi vengano utilizzati e mostra anche l'uso del comando WAIT per inserire nel programma delle pause, scandite in centesimi di secondo (in questo caso 2 secondi):

```
10 ZAP:WAIT 200
20 EXPLODE
30 GOTO 10
```

Il comando SOUND ha il seguente formato:

SOUND C,P,V

dove C = Canale o numero dell'oscillatore (1—6), P = Tonalità (10—5000) e V = Volume (0—15). Il Canale, definito dai valori 1, 2 o 3, seleziona uno dei tre oscillatori (4, 5 e 6 sono equivalenti ma selezionano anche il rumore). La Tonalità è leggermente imprecisa: 10 corrisponde alla nota più alta (approssimativamente a 10 KHz) e 5000 a quella più bassa (a 100 Hz). Il Volume massimo corrisponde a 15, ma il livello più soddisfacente è 6. Se a V viene attribuito il valore 0, il controllo è effettuato dall'inviluppo di volume definito nel comando PLAY.
Il maggior limite del comando SOUND è l'impossibilità di definire la durata di una nota. Ciò significa anche che questo comando non può determinare la propria conclusione. L'unico modo per cessare l'emissione di un suono è di usare un nuovo comando PLAY ma con tutti i parametri posti a zero. Il comando MUSIC è il migliore per definire con precisione i suoni. La sua semplice struttura permette di interpretare con facilità un programma musicale anche piuttosto complicato. Il formato è il seguente:

MUSIC C,O,N,V

dove C = Canale (1, 2 o 3), O = Ottava (0—6), N = Nota (0—12) e V = Volume (0—15). Questo comando funziona in modo assai simile a SOUND. Il Canale seleziona gli oscillatori 1, 2 o 3 (ma col comando MUSIC non è possibile selezionare il rumore) ed il Volume si estende da 0 (nel qual caso il controllo di volume viene effettuato dal comando PLAY) a 15. L'Ottava permette di definire una specifica ottava nella quale suonare la nota (N), che fa parte del comando. Se all'Ottava viene attribuito il valore 0, le note più basse verranno suonate a partire da una frequenza di 32,7 Hz. Un valore O = 6 consente un'estensione fino a 3951,07 KHz. I 12 valori di N (Nota) corrispondono alle note musicali secondo la seguente tabella:

1	2	3	4	5	6
do	do#	re	re#	mi	fa
7	**8**	**9**	**10**	**11**	**12**
fa#	sol	sol#	la	la#	si

Immagini alla ribalta

La grafica dello Spectrum è di facile impiego

Lo Spectrum si è dimostrato un eccellente punto di partenza per coloro che sono interessati alla grafica ad alta risoluzione ed al colore. La semplicità del suo uso rende la progettazione grafica facilmente realizzabile anche per chi abbia poca dimestichezza con la programmazione.
I colori per i caratteri, lo schermo ed il margine vengono selezionati con i comandi chiamati rispettivamente INK, PAPER e BORDER. Inoltre, l'utente può definire fino a 21 nuovi simboli grafici.
La visualizzazione avviene su 24 righe di 32 colonne. Le ultime due righe, comunque, sono riservate a messaggi provenienti dal computer o a righe di comandi immessi da tastiera. Pertanto, l'area utile risulta essere di 22 × 32 caratteri. In alta risoluzione ciò equivale a 176 × 256 pixel. Una delle caratteristiche più utili dello Spectrum è la sua capacità di combinare sul video immagini in alta risoluzione e testi: questa possibilità permette di creare dei grafici corredati di scritte esplicative, grafici a barre e via dicendo. Dopo aver costruito un'immagine sullo schermo, è possibile "salvarla" su nastro e, successivamente, richiamarla in memoria all'occorrenza. Il co-

Per suonare un la a 440 Hz sul canale 1 a volume 6, il formato del comando è il seguente:

MUSIC 1,3,10,6

Comunque, per utilizzare al meglio tutte le potenzialità dell'Oric-1, il miglior modo di procedere è usare i comandi MUSIC e PLAY assieme. Il comando PLAY ha questo formato:

PLAY C,N,E,P

dove C = Canale (0—6), N = Rumore (0—7), E = Inviluppo (1—7) e P = Periodo o Fase d'Inviluppo. Canale e Rumore permettono selezioni più complesse di quelle viste in precedenza:

Numero	Canale	Rumore
0	Tutti Osc. off	off
1	Osc. 1	+Osc. 1
2	Osc. 2	+Osc. 2
3	Osc. 1 & 2	+Osc. 1 o 2
4	Osc. 3	+Osc. 3
5	Osc. 1 & 3	+Osc. 1 o 3
6	Osc. 2 & 3	+Osc. 2 o 3
7	Osc. 1, 2 & 3	+Osc. 1, 2 o 3

I comandi MUSIC o SOUND definiti in precedenza con volume selezionato a 0, possono venir suonati (PLAY) assieme, secondo il numero di canale scelto, per ottenere accordi di 3 note. Il Rumore, come parte del comando, definisce quale oscillatore emetterà rumore e suoni fusi assieme (mixati). L'Inviluppo seleziona uno dei 7 inviluppi di volume prefissati per una o più note.
Queste possibilità di scelta sono descritte nel Manuale dell'Oric-1.
La sola variabile di controllo sull'Inviluppo è il Periodo d'Inviluppo (P), grazie alla quale si può definire la durata complessiva di un inviluppo (da 0 a 32767).
Questa durata varia da un inviluppo all'altro, ma ad un valore di 5000 corrisponde una durata di circa 2 secondi.
I comandi per la generazione di effetti audio, nell'Oric-1, sono facili da usare e dimostrano che è stata prestata molta cura nel fornire funzioni di semplice applicazione. Il solo altro computer dotato sia di un BASIC altrettanto funzionale sia di un controllo d'inviluppo è il Micro della BBC.

Il comando SOUND
Il comando SOUND usato per produrre il rumore di un'astronave che atterra:

```
10 REM ****************
20 REM *ATTERRAGGIO*
30 REM ****************
40 FOR P=10 TO 3000 STEP
   10
50 SOUND 2,P,6
60 PLAY 2,0,1,1
70 NEXT P
80 WAIT 75
90 PLAY 0,0,0,0
100 END
```

mando SCREEN$ serve a questo scopo, oltre che per copiare sulla stampante l'immagine presente sullo schermo.
In bassa risoluzione, il comando PRINT AT consente di posizionare (sia verticalmente sia orizzontalmente) il cursore per la visualizzazione di un carattere o simbolo grafico. Sono disponibili anche alcune comode variazioni: visualizzazione in modo inverso, lampeggiante (FLASH) o brillante (BRIGHT). Molto utile (e piuttosto insolita) è la possibilità di sovrapporre (mediante il comando OVER) un carattere ad un altro, già presente in qualsiasi locazione dello schermo. Questo procedimento è particolarmente stimolante quando vengono combinati testi ed immagini grafiche in alta risoluzione, ma è anche possibile scrivere su di un grafico senza cancellarlo. Tuttavia, occorre prestare una certa attenzione nell'impiego di questo metodo, in quanto la gestione dei colori può diventare piuttosto complicata.
Il video è controllato da due aree di memoria: una è relativa ai caratteri da visualizzare, mentre nella seconda vengono registrati gli attributi di ogni singolo carattere (ossia l'elenco di informazioni impostate con i comandi INK, PAPER, FLASH, ecc.). Questi attributi vengono codificati in un singolo byte ed è possibile esaminare, in BASIC, gli attributi di ogni singola cella dello schermo usando il comando ATTR.

Il BASIC dello Spectrum prevede i comandi:

PLOT x,y

che attribuisce al pixel di coordinate (x,y) il colore specificato dall'ultimo comando INK.

DRAW x,y,p

Il comando DRAW traccia una linea dalla posizione attuale del cursore al punto indicato dalle coordinate specificate nel comando. Se è presente anche un terzo numero (p), la linea si trasforma in un arco. Questo numero è di solito una frazione di π (3,14159...). Se a p sarà attribuito il valore di π, la linea tracciata sarà un semicerchio. Variando il valore di p, si ottiene una maggiore o minore curvatura dell'arco. Con valori di p positivo o negativo, si possono tracciare archi ai lati della ipotetica retta che unisce i punti di riferimento.

CIRCLE x,y,r

Il comando CIRCLE traccia un cerchio di centro (x,y) e raggio r. Nella maggior parte dei dialetti BASIC, con il comando CIRCLE è anche possibile schiacciare una circonferenza per trasformarla in un'ellisse ma, purtroppo, lo Spectrum non prevede questa possibilità.
Un inconveniente abbastanza fastidioso, nell'uso del colore in alta risoluzione, proviene dal fatto che è possibile indicare un solo colore INK per ogni cella video di 8 × 8 pixel. Di conseguenza, se due linee di diverso colore s'intersecano, i pixel della cella nella quale avviene l'incrocio assumeranno il colore INK della linea che viene tracciata per seconda.

Sorrida, prego...
Il programma chiarisce l'uso in alta risoluzione dei comandi PLOT, CIRCLE e DRAW e disegna un viso che "sorride":

```
10 REM *VISO SORRIDENTE*
20 CLS
30 BORDER 6
40 PAPER 6
50 INK 2
60 CIRCLE 122,88,50
70 CIRCLE 97,108,5
80 CIRCLE 147,108,5
90 PLOT 92,68
100 DRAW 152,68,PI/3
110 END
```

I "sensi" del computer

Ad un computer si possono collegare vari tipi di sensori, adatti a processi di controllo

I microprocessori vengono impiegati in molti elettrodomestici: lavatrici, tostapane, videoregistratori ed impianti di riscaldamento e via dicendo. Non deve quindi sorprendere il fatto che esista la possibilità di collegare tra loro questi apparecchi, ponendoli sotto il controllo di un sistema centralizzato. I sistemi di questo tipo si dividono in tre categorie: i sistemi "dedicati", i sistemi "ad interruzione" ed i sistemi di controllo "su rete".

I sistemi dedicati si trovano comunemente in commercio, ma è anche possibile autocostruirseli. Essi possono lavorare in abbinamento anche con un normale home computer ed usano particolari interfacce per collegarsi e controllare unità elettriche o elettromeccaniche come l'impianto luce od un termostato.

Nella seconda categoria di sistemi di controllo sono impiegati dispositivi ad interruzione collegati, ad esempio, all'impianto di riscaldamento o a rilevatori di fumo o di calore. Quando uno di questi apparati rileva un evento anormale, vengono inviati al computer dei segnali che provocano l'interruzione del programma in corso d'esecuzione. Sebbene un sistema di questo genere debba funzionare ininterrottamente, una momentanea mancanza di corrente può non risultare dannosa, in quanto gli apparecchi sotto controllo sono, in genere, dotati di una parziale capacità di autoregolazione.

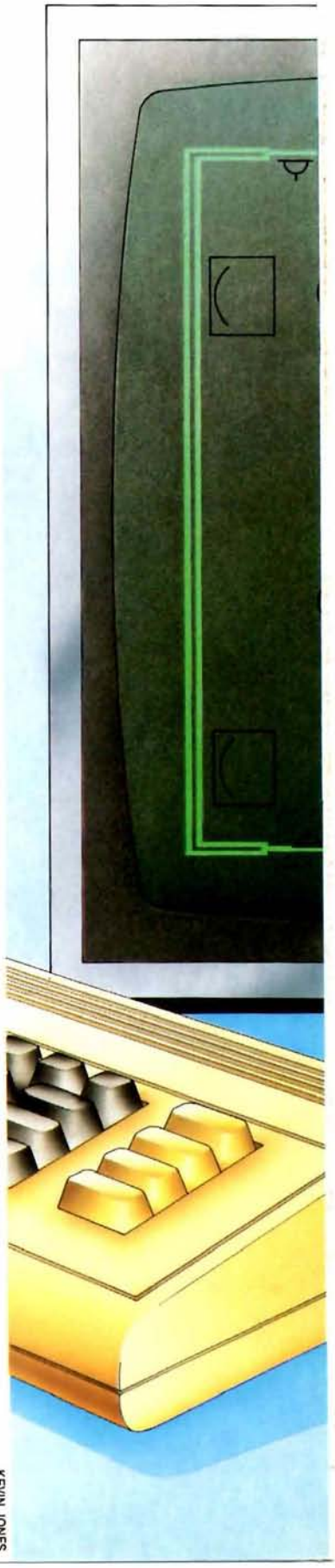
Il principio di funzionamento è abbastanza elementare: nel computer sono memorizzati numerosi programmi (uno per ciascun apparecchio collegato), oltre a quello in corso d'esecuzione. Supponiamo che, mentre ci stiamo divertendo col nostro gioco preferito, il rilevatore di fumo invii un messaggio d'interruzione: in risposta a questo segnale, il computer sospende il programma di gioco (conservando però tutti i dati relativi alla situazione di gioco) e fa partire il programma specifico, associato al rilevatore di fumo. A questo punto, sul video potrebbe apparire un messaggio che avverte della presenza d'un principio d'incendio; oppure, se in quel particolare momento il computer non è in uso, potrebbe automaticamente suonare un campanello d'allarme. Portate a termine le operazioni necessarie a rimuovere lo stato d'emergenza (ad esempio: il focolaio è stato individuato e spento), è possibile riprendere il gioco (o altro programma) dall'esatto punto in cui era stato interrotto. In altre situazioni, invece, l'intervento dell'operatore non è indispensabile, e le operazioni vengono svolte dal computer automaticamente, addirittura in apparente simultaneità con il programma in corso, la cui esecuzione, praticamente, non risente affatto dell'interruzione.

Alla terza categoria, infine, appartengono i sistemi di controllo su rete, nei quali i segnali necessari al controllo dei vari elettrodomestici viaggiano sul normale impianto elettrico. Ovviamente, questo tipo di sistema deve essere istallato da personale qualificato, essendo in gioco pericolose tensioni.

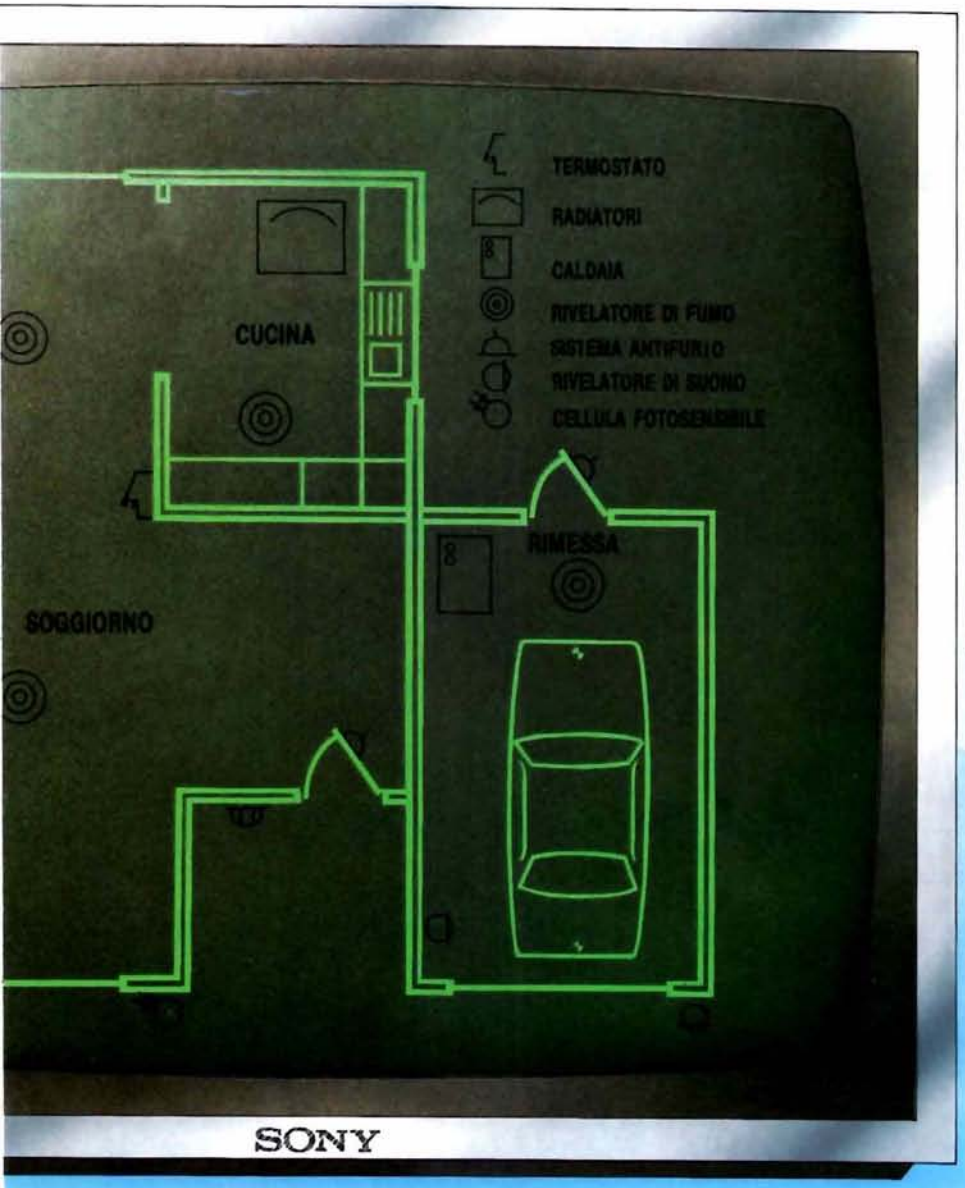
Ciascun apparato, dislocato in prossimità dell'apparecchio da controllare, è identificato da un codice (in pratica un "indirizzo"), che permette di attivarlo o disattivarlo mediante un segnale ad alta frequenza generato da una centralina (o da un computer). Il sistema, inoltre, può essere arricchito da un apparecchio di controllo a distanza: in questo caso, usando la normale linea telefonica, si possono inviare comandi anche da luoghi molto distanti (per esempio dall'ufficio o dal luogo di villeggiatura).

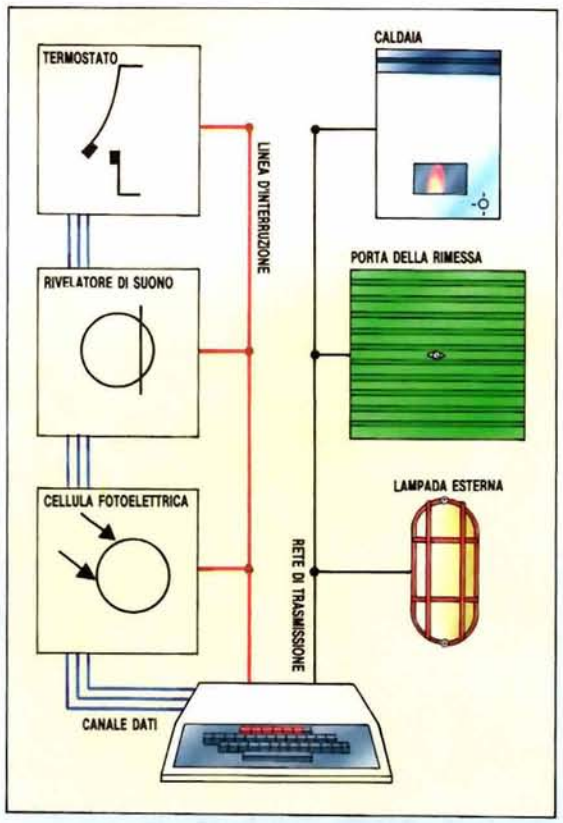
Tutti i sistemi descritti si trovano normalmente in commercio, anche se quelli dell'ultima categoria sono di scarsa applicabilità in Italia. Infatti, questi sistemi funzionano soltanto sugli impianti elettrici nei quali esistono la linea di "terra" e le linee di "fase" e di "neutro" sono identificabili con assoluta certezza e ciò, purtroppo, non accade nella maggior parte degli impianti domestici italiani.

I sensori possono essere di qualunque tipo: dai semplici microinterruttori del tipo usato per gli allarmi antifurto ai più complessi microcircuiti digitali per la misurazione della temperatura. Ci sono diversi home computer capaci di "sostenere" simili livelli d'espansione, ad esempio: l'Apple II, il Commodore 64 ed il BBC Modello B. Altre macchine, invece, richiedono notevoli modifiche per ottenere risultati accettabili. I costi maggiori per l'istallazione di un sistema computerizzato di controllo degli apparecchi domestici, sono costituiti dall'hardware necessario per collegare il computer all'impianto elettrico principale. Mentre è probabile che la maggior parte del lavoro necessario al possessore del computer, per mettere a punto un sistema di questo tipo, sia dedicato alla progettazione e stesura del software. Poiché questi sistemi si basano essenzialmente sulla rapidità della risposta (non serve a niente che l'allarme antincendio scatti dopo che la casa è andata a fuoco!), i programmi di controllo devono essere scritti in codice macchina. Listati di programmi di questo tipo non sono ancora facilmente reperibili, ma lo saranno in un prossimo futuro.



KEVIN JONES





"Indirizzi" per elettrodomestici

Questo schema illustra due tecniche che permettono ad un computer di controllare un certo numero di elettrodomestici. Quando uno dei tre sensori (sulla sinistra) ha qualcosa da segnalare, esso invia un segnale elettronico lungo la "linea d'interruzione". Questo segnale giunge direttamente al microprocessore, che sospende temporaneamente qualsiasi programma in corso e passa il controllo del sistema ad una procedura specifica. Questa interpreterà tutte le informazioni inviate dal sensore sul canale dati ed agirà di conseguenza. Gli apparecchi sulla destra sono collegati alla rete ed il computer può metterne in funzione uno qualsiasi inviando una serie d'informazioni composta, ad esempio, dal codice specifico per la porta del garage e dall'istruzione per farla aprire



CENTRALINA DI CONTROLLO

La figura riporta una schematica rappresentazione di una casa, visualizzata sullo schermo di un computer. Nei sistemi di controllo per impianti industriali e nei sistemi di sicurezza computerizzati si usano metodi di questo tipo. Naturalmente, se il funzionamento è "ad interruzione", non occorre visualizzare simili immagini, in quanto il software svolge tutte le funzioni di controllo "nell'ombra", ossia senza causare, apparentemente, la sospensione del programma in corso d'esecuzione sul computer. In un futuro molto prossimo, gli edifici verranno progettati includendo, oltre al normale impianto elettrico, anche delle reti interne di controllo centralizzato

Velocità di ricerca

Riordinare una matrice in BASIC richiede tempo ma velocizza la ricerca

Fin qui abbiamo sviluppato la maggior parte del programma per il nostro piccolo database, ma non è stata ancora definita con precisione la procedura di memorizzazione e di richiamo su archivio magnetico, né il sottoprogramma per la gestione del campo CAMMOD$. Il programma completo è riportato al termine di questa lezione.

Le linee da 10250 a 10330 provvedono a convertire i caratteri del nominativo in maiuscole. Nelle linee da 10350 a 10370 vengono contati i caratteri, cercando al tempo stesso eventuali spazi di separazione. La posizione dell'ultimo spazio incontrato viene memorizzata nella variabile S.

Le linee da 10400 a 10420 provvedono a trasferire, uno alla volta, i caratteri, convertiti in maiuscole, nella stringa CNAM$. Tutti i caratteri non alfabetici vengono però ignorati: il punto (ASCII 46), l'apostrofo (ASCII 39), lo spazio (ASCII 32) e così via. Un simile processo avviene tra la linea 10450 e 10470, ma in questo caso i caratteri vengono trasferiti in SNAM$.

Qualora N$ contenga una sola parola (GIOVANNI, ad esempio), la variabile S varrà 0 e tutti i caratteri verranno trasferiti in SNOM$.

Le linee da 10490 a 10500 occorrono per azzerare le variabili usate nelle subroutine appena descritte, prima di un loro nuovo impiego. Ciò si rende indispensabile ogniqualvolta venga adoperata una procedura del tipo LET X$ = X$ + Y$. Se omettessimo l'azzeramento della variabile X$, ad ogni nuovo "giro", continueremmo ad aggiungere caratteri ad X$ indefinitivamente. Si noti che la variabile SCEL viene azzerata nella routine AGGREC, poiché vogliamo assicurarci che l'operatore sia costretto ad inserire almeno un record soltanto nel caso che il file sia vuoto (ossia solamente la prima volta che il programma viene lanciato).

Avendo approntata una routine per l'inserimento dei nuovi record, ci occorre adesso una procedura per trasferire i dati tra la memoria del computer e quella di massa (nastro o disco). Il metodo più semplice consisterebbe nello scrivere tutti i record in un file (RUBR.DAT nel nostro esempio), prelevandoli dalle varie matrici e senza modificarne la sequenza. Un simile sistema porterebbe però a notevoli perdite di tempo nella fase di ricerca di un particolare nominativo, poiché, se i record non si susseguono con un preciso ordine logico, occorrerà iniziare dal primo record ed effettuare una ricerca sequenziale su tutto il file, per trovare l'elemento desiderato. Qualora, invece, avessimo previsto di memorizzare i nominativi in ordine alfabetico, la subroutine di ricerca potrebbe risultare molto più efficace. Se, per esempio, l'ultimo nominativo inserito è stato GIUSEPPE BANCHI, ossia CAMMOD$(DIMENSIONE−1) = BANCHI GIUSEPPE, la ricerca potrebbe iniziare tranquillamente all'inizio della matrice ordinata. Purtroppo, sia il riordino sia la ricerca di dati sono operazioni che richiedono notevoli quantità di tempo ed occorre stabilire, in fase di progetto, a quale intendiamo dare priorità.

Nella nostra "rubrica computerizzata" abbiamo scelto di eseguire un riordino alfabetico dei nominativi dopo ogni nuovo inserimento e prima di trasferirli su memoria di massa, in base alla considerazione che la funzione adoperata più frequentemente, in una rubrica, è la consultazione e non l'inserimento di nuovi nominativi. Così facendo, sacrifichiamo del tempo al termine di una fase d'inserimento, ma otteniamo una ricerca molto più rapida ed efficace.

La variabile RMOD viene appunto impiegata per stabilire se è stato aggiunto un nuovo nominativo nel database. Inizialmente, essa vale 0, per indicare che non è stato effettuato alcun nuovo inserimento. Qualora venga eseguita una qualsiasi operazione di modifica (come ad esempio l'aggiunta di un nuovo nominativo), ad RMOD viene assegnato il valore 1. In quelle parti di programma, nelle quali occorre determinare se il file è stato modificato, basta pertanto consultare il valore di RMOD. Per esempio, la subroutine EXPROG (per uscire dal programma) controlla il valore di RMOD nella linea 11050, per decidere se il file deve essere nuovamente "salvato" prima di tornare al sistema operativo. Anche nella subroutine di ricerca occorre verificare RMOD. Se il valore è 1, significa che l'ordine dei record è stato alterato da un'altra operazione e che, prima della ricerca, è necessario riordinare i dati. Se, invece, il valore è 0, si può passare immediatamente alla ricerca. Ovviamente, al termine della procedura di riordino, da noi indicata col nome SRTREC, il valore di RMOD viene sempre azzerato (vedere la linea 11320). È opportuno commentare brevemente SRTREC, prima di passare oltre. Il metodo di riordino adottato è quello noto col nome di "bubble sort" (vedere a pag. 286): il più semplice, ma anche il più lento di tutti. Lo abbiamo scelto, appunto, per la sua semplicità, ma niente vieta di sostituire questo modulo, in seguito, quando avremo appreso altri sistemi più veloci e sofisticati. Questa decisione, comunque, dipende dal numero di elementi da riordinare. Il tempo necessario per un bubble sort di n elementi è direttamente proporzionale al quadrato di n. Ossia, suppo-

nendo che il riordino di due elementi richieda 4 millisecondi, quello di quattro elementi richiede ben 16 millisecondi, mentre per 50 elementi occorrono due secondi e mezzo e per 1.000 elementi più di 16 minuti. Per la nostra applicazione, un'attesa di due o tre secondi può essere ancora accettabile.
Ecco una delle ragioni per le quali abbiamo limitato a 50 il numero totale di nominativi memorizzabili. Più avanti nel corso, vedremo come aumentare questo limite, sfruttando "dinamicamente" gli archivi magnetici.
Ma torniamo alla nostra subroutine: in essa vengono confrontati il contenuto di CAMMOD$(L) e CAMMOD$(L+1) e viene effettuato uno scambio tra i due soltanto se CAMMOD$(L) è maggiore (alfabeticamente) di CAMMOD(L+1). Il campo relativo all'indice, anche se non attualmente utilizzato, viene opportunamente modificato nelle linee 11490 e 11570. Ad ogni scambio effettuato, la variabile S viene posta ad 1. Alla linea 11290 viene controllato il valore di S: se questo è zero (ossia se non è stato effettuato alcuno scambio) il riordino è completo ed anche la variabile RMOD viene azzerata, altrimenti viene fatto un altro "giro" di confronti.
La routine EXPROG inizia alla linea 11000 controllando se siano state apportate modifiche ai nominativi dal lancio del programma (linea 11050: IF RMOD = 0 THEN RETURN). Se non ve ne sono state, non occorre memorizzare di nuovo le matrici, cosicché l'esecuzione ritorna (RETURN) al programma principale, ossia alla linea 100, che verifica il valore di SCEL. Se questo è 9, in quanto abbiamo optato per l'uscita dal programma, allora viene semplicemente eseguita la linea 110, che contiene una END. Se, al contrario, nella linea 11050 il valore di RMOD risulta essere 1, c'è la possibilità che l'ordine dei nominativi non sia più nella corretta sequenza, e, pertanto, occorre eseguire un riordino (vedere la linea 11070), prima di memorizzare le matrici.
Il sottoprogramma di memorizzazione SAVREC viene chiamato nella linea 11090 ed il relativo modulo inizia alla 12000. Nel listato principale, questo modulo è scritto nella versione BASIC della Microsoft: si presti, dunque, particolare attenzione se viene impiegato un altro dialetto (vedere Basic & Basic). In breve, la linea 12030 "apre" il file RUBR.DAT, assegnando ad esso il canale di comunicazione numero 1. La linea 12050 determina il limite massimo di ripetizioni nel ciclo di scrittura dei record. Esso corrisponde a DIMENSIONE—1, in quanto DIMENSIONE contiene sempre il numero del "prossimo" record da inserire.
Si noti il particolare formato delle linee 12060 e 12070: ogni campo è separato dall'altro con una virgola: ",". Questo carattere è necessario nella maggioranza dei BASIC (e viene effettivamente scritto nel file), poiché occorre, in qualche modo, separare gli elementi tra loro. Le istruzioni PRINT# e INPUT# sono analoghe alla PRINT ed alla INPUT usate nelle operazioni su video. Nel caso di un'immissione da tastiera, INPUT X,Y,Z richiede che l'operatore digiti tre valori separandoli con una virgola, ad esempio: 10,12,15<CR>. Ad X viene assegnato il valore 10, ad Y il valore 12 ed a Z il valore 15. Omettendo le virgole, il BASIC non avrebbe modo di distinguere i tre elementi, ed assegnerebbe alla prima variabile, X nel nostro esempio, il valore 101215. Analogamente, in una lettura da un file eseguita con l'istruzione INPUT#, il BASIC deve essere in grado d'individuare ciascun elemento da assegnare alle variabili, altrimenti i dati verrebbero trasferiti disordinatamente nelle rispettive variabili. Qualora, poi, la lettura del file avvenisse all'interno di un ciclo (come nel nostro programma), i dati del file si esaurirebbero molto prima della conclusione del ciclo e ciò causerebbe un errore del tipo INPUT PAST END (tentativo di lettura oltre la fine del file).
Una volta memorizzati tutti i record nel file, durante la subroutine SAVREC, l'esecuzione ritorna alla linea 90 del programma principale. Nella linea 100 viene controllato il valore di SCEL, per vedere se l'ultima scelta eseguita dall'operatore è stata l'uscita dal programma. In questo caso, SCEL vale 9 e l'esecuzione passa alla linea 110, che contiene l'istruzione END. Invece, se SCEL contiene un valore diverso, viene di nuovo eseguita la visualizzazione del menu delle opzioni.
Ultimo punto da notare è la subroutine FLSIZE, che inizia alla linea 12500. Questa rappresenta un'alternativa alle linee 1510 e 1520. Infatti, il funzionamento del programma, nella versione presentata, dipende dalla presenza di una funzione EOF (linea 1520) che segnala la condizione di "fine-file". Tutti i BASIC dispongono di un sistema per individuare quando si è raggiunta la fine di un file: in alcuni casi viene impiegata la funzione EOF(x), in altri occorre esaminare il contenuto di una particolare locazione di memoria, mediante una PEEK. La subroutine FLSIZE, pertanto, è stata inclusa come ultima risorsa, qualora non si disponga di altri metodi per rilevare la fine del file. In tal caso, occorre sostituire il contenuto della linea 1510 con GOSUB 12500 ed eliminare la linea 1520.

Basic & Basic



Prima di lanciare il programma della rubrica, occorre creare uno speciale file su nastro, usando il seguente programma:

```
10 REM CREAZIONE DEL FILE INIZIALE
20 DIM Z$(1,30)
30 LET Z$(1) = "@PRIMO"
40 SAVE "NFLD" DATA Z$()
50 STOP
```

Eseguito il programma, si riavvolge il nastro e s'impartisce il comando VERIFY "NFLD" DATA Z$(), per verificare l'esatta memorizzazione dei dati. Questa procedura di verifica, anche se raddoppia i tempi di memorizzazione, andrebbe sempre eseguita, prima di spegnere il computer.
Qui di seguito vengono riportate le linee di programma da modificare per l'uso della rubrica computerizzata sullo Spectrum:

```
1100 REM "CREMAT"
1110 DIM N$(50,30)
1120 DIM M$(50,30)
```

SPECTRUM

```
1130 DIM S$(50,30)
1140 DIM T$(50,15)
1150 DIM C$(50,15)
1160 DIM R$(50,15)
1170 DIM X$(50,30)
1180 DIM B$(30)
1190 DIM Z$(30)
1250 LET Z$ = "@PRIMO"
:
1400 REM SUBROUTINE "RDINFL"
1410 LOAD "NDFL" DATA N$()
1420 IF N$(1) = Z$ THEN LET Q$ = Z$ :
     RETURN
1430 LOAD "MFLD" DATA M$()
1440 LOAD "SFLD" DATA S$()
1450 LOAD "TFLD" DATA T$()
1460 LOAD "CFLD" DATA C$()
1470 LOAD "TELFLD" DATA R$()
1480 LOAD "NDXFLD" DATA X$()
1490 REM CHIAMATA A FLSIZE
1500 GOSUB 12500
:
1540 RETURN
:
1640 IF Q$ = Z$ THEN LET SIZE = 1
:
3520 IF Q$ = Z$ THEN GOSUB 3860:RETURN
:
3810 LET SCEL = CODE A$ — 48
:
10090 LET Q$ = ""
:
10200 REM SUBROUTINE "MODNOM"
:
10250 LET D$ = N$(SIZE):LET P$ = ""
10260 FOR L = 1 TO LEN(D$)
10270 LET A$ = D$(L)
10280 LET T = CODE A$
10290 IF T > 97 THEN LET T = T—32
10300 LET A$ = CHR$ T
10310 LET P$ = P$ + A$
10320 NEXT L
10330 LET D$=P$:LET P$="":LET A$="":-
      LET T=LEN(D$)
10340 REM LOCALIZZA ULTIMO SPAZIO
10350 FOR L = 1 TO T
10360 IF D$(L) = " " THEN LET S=L:LET L=T
10370 NEXT L
10380 REM RIMUOVE CARATTERI ESTRANEI
10390 REM E SALVA IL NOME IN P$
10400 FOR L = 1 TO S—1
10410 IF CODE(D$(L)) > 64 THEN LET P$=-
      P$+D$(L)
10420 NEXT L
10430 REM RIMUOVE CARATTERI ESTRANEI
10440 REM E SALVA IL COGNOME IN A$
10450 FOR L = S+1 TO LEN(D$)
10460 IF CODE (D$(L)) > 64 THEN LET
      A$=A$+D$(L)
10470 NEXT L
10480 LET M$(SIZE) = A$ + " " + P$
10490 LET P$ = "": LET A$ = ""
10510 RETURN
```

N.B. A causa del particolare modo col quale vengono trattate le stringhe nello Spectrum, la subroutine appena presentata separa il nome dal cognome al primo spazio incontrato e non all'ultimo.

```
12000 REM SUBROUTINE "SAVREC"
12030 SAVE "NDFL" DATA N$()
12040 SAVE "MFLD" DATA M$()
12050 SAVE "SFLD" DATA S$()
12060 SAVE "TFLD" DATA T$()
12070 SAVE "CFLD" DATA C$()
12080 SAVE "TELFLD" DATA R$()
12090 SAVE "NDXFLD" DATA X$()
:
12150 RETURN
:
12500 REM SUBROUTINE "FLSIZE"
12510 LET SIZE = 50
12520 FOR L = 1 TO 50
12530 IF N$(L) = B$ THEN LET SIZE=L:LET
      L=50
12540 NEXT L
12560 RETURN
```

LYNX

Il Lynx non dispone dell'istruzione STR$. Vedere Basic & Basic a pag. 357. Per il dimensionamento delle stringhe, vedere Basic & Basic a pag. 275.

EOF

Sul Commodore 64 e sul Vic-20 sostituire la linea 1520 con:

```
1520 IF ST AND 64 THEN LET L = 50
```

Sul Dragon 32 eliminare la linea 1520 ed inserire:

```
1485 IF EOF(—1) THEN GOTO 1510
```

Sul Micro BBC si sostituisca:

```
1520 IF EOF# X THEN LET L = 50
```

dove X è il valore numerico usato per l'istruzione OPENOUT (vedere a pag. 318).

OPEN
CLOSE

Vedere a pag. 319.

```
10 REM 'MAINPG'
20 REM == INITL ===
30 GOSUB 1000
40 REM == GREETS ==
50 GOSUB 3000
60 REM == SCELTA ==
70 GOSUB 3500
80 REM == EXECUT ==
90 GOSUB 4000
100 IF SCEL () 9 THEN 60
110 END
1000 REM SUBROUTINE == INITIL ==
1010 GOSUB 1100 : REM == CREMAT == (CREAZIONE MATRICI)
1020 GOSUB 1400 : REM == RDINFL == (LETTURA DEL FILE)
1030 GOSUB 1600 : REM == SETFLG == (ASSEGNAZIONE DELLE FLAG)
1040 REM
1050 REM
1060 REM
1070 REM
1080 REM
1090 RETURN
1100 REM SUBROUTINE == CREMAT == (CREAZIONE MATRICI)
1110 DIM CAMNOM$(50)
1120 DIM CAMMOD$(50)
1130 DIM CAMVIA$(50)
1140 DIM CAMLOC$(50)
1150 DIM CAMNAZ$(50)
1160 DIM CAMTEL$(50)
1170 DIM CAMIND$(50)
1180 REM
1190 REM
1200 REM
1210 LET DIMENSIONE = 0
1220 LET RMOD = 0
1230 LET SVED = 0
1240 LET CORR = 0
1250 REM
1260 REM
1270 REM
1280 REM
1290 REM
1300 RETURN
1400 REM SUBROUTINE == RDINFL == (LETTURA DEL FILE)
1410 OPEN "I",#1,"RUBR.DAT"
1420 INPUT #1,TEST$
1430 IF TEST$ = "@PRIMO" THEN GOTO 1540:REM CHIUDE
     E RITORNA
1440 LET CAMNOM$(1) = TEST$
1450 INPUT #1,CAMMOD$(1),CAMVIA$(1),CAMLOC$(1),CAMNAZ$(1),
     CAMTEL$(1)
1460 INPUT #1,CAMIND$(1)
```

```
1470 LET DIMENSIONE = 2
1480 FOR L = 2 TO 50
1485 IF EOF(1) = -1 THEN LET L = 50 : GOTO 1530
1490 INPUT #1,CAMNOM(L),CAMMOD$(L),CAMVIA$(L),CAMNAZ$(L)
1500 INPUT #1,CAMTEL$(L),CAMIND$(L)
1510 LET DIMENSIONE = DIMENSIONE + 1
1530 NEXT L
1540 CLOSE #1
1550 RETURN
1600 REM SUBROUTINE == SETFLG == (ASSEGNAZIONE DELLE FLAG)
1610 REM LE FLAG VENGONO ASSEGNATE DOPO LA LETTURA DEL FILE
1620 REM
1630 REM
1640 IF TEST$ = "@PRIMO" THEN LET DIMENSIONE = 1
1650 REM
1660 REM
1670 REM
1680 REM
1690 RETURN
3000 REM SUBROUTINE == GREETS == (MESSAGGIO INTRODUTTIVO)
3010 PRINT CHR$(12) : REM PULIZIA DELLO SCHERMO
3020 PRINT
3030 PRINT
3040 PRINT
3050 PRINT
3060 PRINT TAB(12);"RUBRICA COMPUTERIZZATA"
3070 PRINT TAB(13);"PRESENTATA NEL CORSO"
3080 PRINT TAB(14);"'IL MIO COMPUTER'"
3090 PRINT
3100 PRINT TAB(5);"(PREMERE UNO SPAZIO PER CONTINUARE)"
3110 FOR L = 1 TO 1
3120 IF INKEY$ () " " THEN L = 0
3130 NEXT L
3140 PRINT CHR$(12)
3150 RETURN
3500 REM SUBROUTINE == SCELTA == (MENU DELLE OPZIONI)
3510 REM
3520 IF TEST$ = "@PRIMO" THEN GOSUB 3860: REM (FILE VUOTO)
3530 IF TEST$ = "@PRIMO" THEN RETURN
3540 REM == MENU ==
3550 PRINT CHR$(12)
3560 PRINT TAB(20);"MENU DELLE OPZIONI"
3570 PRINT
3580 PRINT
3590 PRINT
3600 PRINT "1. TROVARE UN RECORD (USANDO IL NOME)"
3610 PRINT "2. TROVARE UN RECORD (USANDO PARTE DEL NOME)"
3620 PRINT "3. TROVARE RECORD (USANDO LA LOCALITA')"
3630 PRINT "4. TROVARE UN RECORD (USANDO LE INIZIALI)"
3640 PRINT "5. ELENCO DI TUTTI I RECORD"
3650 PRINT "6. INSERIMENTO DI UN RECORD"
3660 PRINT "7. MODIFICA DI UN RECORD"
3670 PRINT "8. ANNULLAMENTO DI UN RECORD"
3680 PRINT "9. USCITA DAL PROGRAMMA"
3690 PRINT
3700 PRINT
3710 REM == INCHOI ==
3720 REM
3730 LET L = 0
3740 LET I = 0
3750 FOR L = 1 TO 1
3760 PRINT "QUALE (1 - 9)"
3770 FOR I = 1 TO 1
3780 LET A$ = INKEY$
3790 IF A$ = "" THEN I = 0
3800 NEXT I
3810 LET SCEL = VAL(A$)
3820 IF SCEL ( 1 THEN L = 0
3830 IF SCEL ) 9 THEN L = 0
3840 NEXT L
3850 RETURN
3860 REM SUBROUTINE == FIRSTM == (FILE VUOTO)
3870 LET SCEL = 6
3880 PRINT CHR$(12)
3890 PRINT
3900 PRINT TAB(8);"L'ARCHIVIO E' VUOTO ED OCCORRE"
3910 PRINT TAB(8);"INIZIARE INSERENDO IL PRIMO"
3920 PRINT TAB(8);"RECORD DEL FILE"
3930 PRINT
3940 PRINT TAB(5);"(PREMERE UNO SPAZIO PER CONTNUARE)"
3950 FOR B = 1 TO 1
3960 IF INKEY$ () " " THEN B = 0
3970 NEXT B
3980 PRINT CHR$(12) : REM PULIZIA DELLO SCHERMO
3990 RETURN
4000 REM SUBROUTINE == EXECUT ==
4010 REM
4020 IF SCEL = 6 THEN GOSUB 10000
4030 REM
4040 REM 1 E' == FNDREC ==
4050 REM 2 E' == FNDNMS ==
4060 REM 3 E' == FNDTWN ==
4070 REM 4 E' == FNDINT ==
4080 REM 5 E' == LSTREC ==
4090 IF SCEL = 6 THEN GOSUB 10000
4100 REM 7 E' == MODREC ==
4110 REM 8 E' == DELREC ==
4120 IF SCEL = 9 THEN GOSUB 11000
4130 REM
4140 RETURN
10000 REM SUBROUTINE == AGGREC ==
10010 PRINT CHR$(12) : REM PULIZIA DELLO SCHERMO
10020 INPUT "NOMINATIVO ";CAMNOM$(DIMENSIONE)
10025 IF LEN(CAMNOM$(DIMENSIONE)) = 0 THEN 10020
10030 INPUT "INDIRIZZO ";CAMVIA$(DIMENSIONE)
10040 INPUT "LOCALITA' ";CAMLOC$(DIMENSIONE)
10050 INPUT "NAZIONE ";CAMNAZ$(DIMENSIONE)
10060 INPUT "NUMERO DI TELEFONO ";CAMTEL$(DIMENSIONE)
10070 LET RMOD = 1 : REM FLAG DI 'RECORD AGGIUNTO'
10080 LET CAMIND$(DIMENSIONE) = STR$(DIMENSIONE)
10090 LET TEST$ = ""
10100 GOSUB 10240 : REM == MODNOM ==
10110 LET SCEL = 0
10120 LET DIMENSIONE = DIMENSIONE + 1
10130 REM
10140 REM
10150 RETURN
10200 REM SUBROUTINE == MODNOM ==
10210 REM CONVERTE IL CONTENUTO DI CAMNOM$ IN MAIUSCOLE
10220 REM RIMUOVE CARATTERI ESTRANEI E MEMORIZZA:
10230 REM COGNOME+SPAZIO+NOME IN CAMMOD$
10240 REM
10250 LET N$ = CAMNOM$(DIMENSIONE)
10260 FOR L = 1 TO LEN(N$)
10270 LET TEMP$ = MID$(N$,L,1)
10280 LET T = ASC(TEMP$)
10290 IF T )= 97 THEN T = T - 32
10300 LET TEMP$ = CHR$(T)
10310 LET P$ = P$ + TEMP$
10320 NEXT L
10330 LET N$ = P$
10340 REM LOCALIZZA L'ULTIMO SPAZIO
10350 FOR L = 1 TO LEN(N$)
10360 IF MID$(N$,L,1) = " " THEN S = L
10370 NEXT L
10380 REM RIMUOVE CARATTERI ESTRANEI
10390 REM E SALVA NOME IN CNAM$
10400 FOR L = 1 TO S - 1
10410 IF ASC(MID$(N$,L,1)) ) 64 THEN CNAM$ = CNAM$ +
      MID$(N$,L,1)
10420 NEXT L
10430 REM RIMUOVE CARATTERI ESTRANEI
10440 REM E SALVA CGNOME IN SNAM$
10450 FOR L = S + 1 TO LEN(N$)
10460 IF ASC(MID$(N$,L,1)) ) 64 THEN SNAM$ = SNAM$ +
      MID$(N$,L,1)
10470 NEXT L
10480 LET CAMMOD$(DIMENSIONE) = SNAM$ + " " + CNAM$
10490 LET P$ = "":LET N$ = "":LET SNAM$ = "": LET CNAM$ = ""
10500 RETURN
11000 REM SUBROUTINE == EXPROG ==
11010 REM RIORDINA E MEMORIZZA IL FILE
11020 REM SE SONO STATE APPORTATE MODIFICHE
11030 REM AD ALMENO UN RECORD (RMOD = 1)
11040 REM
11050 IF RMOD = 0 THEN RETURN
11060 REM
11070 GOSUB 11200 : REM == SRTREC ==
11080 REM
11090 GOSUB 12000 : REM == SAVREC ==
11100 RETURN
11200 REM SUBROUTINE == SRTREC ==
11210 REM ESEGUE UN RIORDINO ALFABETICO
11220 REM SUL CAMPO CAMMOD$, ED AGGIORNA
11230 REM CAMIND$
11240 REM
11250 LET S = 0
11260 FOR L = 1 TO DIMENSIONE - 2
11270 IF CAMMOD$(L) ) CAMMOD$(L+1) THEN GOSUB 11350
11280 NEXT L
11290 IF S = 1 THEN 11250
11300 REM
11310 REM
11320 LET RMOD = 0 : REM AZZERA FLAG 'RECORD MODIFICATO'
11330 REM
11340 RETURN
11350 REM SUBROUTINE == SWPREC == (SCAMBIO RECORD)
11360 LET TEMNOM$ = CAMNOM$(L)
11370 LET TEMMOD$ = CAMMOD$(L)
11380 LET TEMVIA$ = CAMVIA$(L)
11390 LET TEMLOC$ = CAMLOC$(L)
11400 LET TEMNAZ$ = CAMNAZ$(L)
11410 LET TEMTEL$ = CAMTEL$(L)
11420 REM
11430 LET CAMNOM$(L) = CAMNOM$(L+1)
11440 LET CAMMOD$(L) = CAMMOD$(L+1)
11450 LET CAMVIA$(L) = CAMVIA$(L+1)
11460 LET CAMLOC$(L) = CAMLOC$(L+1)
11470 LET CAMNAZ$(L) = CAMNAZ$(L+1)
11480 LET CAMTEL$(L) = CAMTEL$(L+1)
11490 LET CAMIND$(L) = STR$(L)
11500 REM
11510 LET CAMNOM$(L+1) = TEMNOM$
11520 LET CAMMOD$(L+1) = TEMMOD$
11530 LET CAMVIA$(L+1) = TEMVIA$
11540 LET CAMLOC$(L+1) = TEMLOC$
11550 LET CAMNAZ$(L+1) = TEMNAZ$
11560 LET CAMTEL$(L+1) = TEMTEL$
11570 LET CAMIND$(L+1) = STR$(L+1)
11580 LET S = 1
11590 REM
11600 RETURN
12000 REM SUBROUTINE == SAVREC ==
12010 REM
12020 REM
12030 OPEN "O",#1,"RUBR.DAT"
12040 REM
12050 FOR L = 1 TO DIMENSIONE - 1
12060 PRINT #1,CAMNOM$(L);",";CAMMOD$(L);",";CAMVIA$(L);
      ",";CAMLOC$(L)
12070 PRINT #1,CAMNAZ$(L);",";CAMTEL$(L);",";CAMIND$(L)
12080 NEXT L
12090 REM
12100 REM
12110 REM
12120 REM
12130 CLOSE #1
12140 REM
12150 RETURN
12500 REM SUBROUTINE == FLSIZE ==
12510 IF CAMNOM$(L) = "" THEN LET L = 50
12520 IF CAMNOM$(L) = "" THEN RETURN
12530 LET DIMENSIONE = DIMENSIONE + 1
12540 REM
12550 REM
12560 RETURN
```

Vannevar Bush

L'inventore di un interessante apparecchio per il calcolo di equazioni differenziali



Molti considerano Vannevar Bush il padre del computer. Egli partecipò attivamente allo sviluppo dei computer, a partire dal 1930, quando progettò un analizzatore differenziale meccanico: questa invenzione incentivò la ricerca che ha portato allo sviluppo degli attuali computer digitali.

Bush nacque vicino a Boston, Massachusetts, l'11 marzo 1890 e, seguendo le orme del padre, studiò ingegneria. Laureatosi, lavorò per breve tempo alla General Electric Company, quindi divenne lettore presso il college frequentato da studente. Successivamente, seguì dei corsi di perfezionamento all'Università di Harvard ed al Massachusetts Institute of Technology (MIT). Durante la Prima Guerra Mondiale partecipò allo sviluppo di rivelatori sottomarini per le Forze Armate Americane.

Bush progettò la sua prima invenzione, un apparecchio per misurazioni topografiche, quando era ancora studente. Il meccanismo di questo apparecchio, sospeso tra due ruote di bicicletta, permetteva di misurare l'altezza del terreno sul quale la macchina si muoveva e ne riproduceva il profilo sotto forma di un tracciato grafico. L'apparecchio era inoltre dotato di un "integratore", poiché per la determinazione dell'altezza di un qualsiasi punto occorreva conoscere tutti i dati ottenuti fino a quel momento sul tragitto percorso. Bush, divenuto docente al MIT, si dedicò all'approfondimento di uno dei maggiori problemi dell'approvvigionamento energetico, ossia come evitare le interruzioni di erogazione causate da un'improvvisa e massiccia crescita della domanda energetica. Le equazioni matematiche, necessarie alla soluzione di tale problema, erano state scoperte alla fine del XIX secolo dallo scienziato scozzese James Maxwell (1831-1879). Ma poiché la quantità di equazioni da risolvere, in questo particolare problema, era tale da precludere qualsiasi procedimento manuale, Bush si dedicò al progetto di una macchina capace di svolgere questo immane lavoro. Egli si ispirò al lavoro compiuto da Lord Kelvin (1824-1907), uno scienziato inglese che aveva fabbricato una macchina per la risoluzione delle equazioni matematiche relative alla previsione delle maree.

Agli inizi del 1920 Bush costruì una prima macchina che fu chiamata il "telegrafo per prodotto". Questo apparecchio permetteva all'operatore di seguire le linee d'onda che vengono tracciate su un grafico mediante un potenziometro (un apparecchio che traduce uno spostamento fisico in una variazione di voltaggio). Questi segnali elettrici venivano quindi inviati ad un wattmetro appositamente adattato: il disco ruotante, presente in ogni wattmetro, registrava quindi la quantità di corrente consumata dall'integratore dei valori di fluttuazione della corrente e del voltaggio per produrre il "prodotto".

Il successo di questa macchina nella risoluzione di una serie di equazioni simultanee fece intuire che si poteva costruire un nuovo apparecchio capace di risolvere equazioni differenziali di secondo grado ancor più difficili. Bush compì ulteriori ricerche e nel 1931 fu in grado di progettare il primo analizzatore differenziale. Anche questo apparecchio dette ottimi risultati e ne furono costruiti alcuni esemplari in varie nazioni europee. Negli Stati Uniti anche la Scuola di Ingegneria Elettrica Moore, in Pennsylvania, dove venne in seguito costruito il primo computer ENIAC (vedere a pag. 81), ne commissionò uno. Quando il telegrafo di prodotto di Bush poté essere tarato per margini d'errore inferiori all'uno o due per cento, l'analizzatore differenziale produsse risultati con una precisione dello 0,05 per cento. Tuttavia il conseguimento di questo risultato portò ad un aumento dei costi di produzione: ogni diminuzione di un decimo di punto nel margine d'errore, incrementò il costo dell'apparecchio di 10 volte! Con l'introduzione dei computer digitali, comunque, il costo della macchina ed il grado di precisione crebbero in seguito di pari passo.

Nel 1939 Bush divenne rettore della facoltà d'ingegneria e vicepresidente del Carnagie Institute. Le sue notevoli capacità nell'amministrare i cospicui fondi per la ricerca scientifica sono comprovati dalla sua nomina alla presidenza dei Comitati di Ricerche per la Difesa Nazionale avvenuta negli anni seguenti. In questa carica egli divenne responsabile, durante il periodo bellico, del Settore Ricerche per le Forze Armate USA ed in particolare contribuì all'approvazione del "progetto Manhattan" che portò alla creazione della bomba atomica. Bush si ritirò dall'attività pubblica nel 1955 e si dedicò ai suoi hobby preferiti: il canottaggio, l'allevamento dei tacchini e le invenzioni. Morì nel 1974.

L'analizzatore differenziale

Questo apparecchio venne progettato per risolvere un'importante classe di funzioni matematiche comune a molte branche della scienza e dell'ingegneria: le equazioni differenziali di secondo grado. Il metodo era stato inizialmente impostato da Lord Kelvin e si basava sull'impiego dei dati prodotti da un "integratore" (un dispositivo per calcolare l'area racchiusa in una curva) da parte di un secondo integratore. Finché non vennero inventati gli amplificatori, spesso i segnali coinvolti in questo processo erano troppo deboli per essere elaborati più volte. La macchina di Bush, realizzata nel 1931, era interamente meccanica: un complesso insieme di ingranaggi, assi e motori elettrici. L'immissione e l'emissione dei dati avveniva per mezzo di assi ruotanti e mediante un amplificatore dei segnali di torsione.

Nel 1940 venne realizzata una versione più sofisticata dell'analizzatore differenziale, usando componenti elettriche, ma il peso complessivo dell'apparecchio era di oltre cento tonnellate. L'emissione dei dati era in forma digitale, con una precisione di una parte su 10.000. Le informazioni di partenza ed i parametri di controllo venivano inseriti per mezzo di nastri perforati. L'apparecchio venne usato, durante la Seconda Guerra Mondiale, per impieghi balistici e crittografici

Per i più piccoli

Tra gli home computer ed i giocattoli dell'ultima generazione sono poche le differenze



TONY SLEEP

Giovani menti

I bambini sembrano essere più ricettivi degli adulti alle nuove tecnologie: questi ultimi si mostrano restii all'idea di doversi confrontare con nuovi schemi di pensiero. La versatilità dei microprocessori fa sì che non esista, in pratica, un limite minimo d'età per giocattoli elettronici e strumenti didattici

I microprocessori, che costituiscono il "cuore" dei microcomputer, vengono oggi comunemente impiegati in moltissimi apparecchi di uso domestico: nelle macchine da cucire, nelle lavatrici e persino nelle serrature delle porte.

Anche i fabbricanti di giocattoli hanno sperimentato con successo l'uso di microprocessori, soprattutto per il controllo di piccoli modelli di automobili e di treni.

La Texas Instruments (TI), una casa produttrice di computer, ha trovato molto remunerativa la produzione di giocattoli didattici basati su microprocessori. Il primo prodotto della TI in questo campo è stato un elementare modello di calcolatrice, in grado di proporre semplici problemi di aritmetica: il "Little Professor" (piccolo professore). Questo apparecchio ha ottenuto un notevole successo e, nonostante ne esista anche una versione più recente (lo Speak & Maths), le sue vendite non sembrano diminuire.

Lo Speak & Maths è stato il secondo giocattolo didattico della TI nel quale viene impiegato un chip di sintesi vocale, prodotto dalla stessa casa ed usato anche nell'home computer TI99/4A.

Lo Speak & Spell, lanciato nel '78, è un apparecchio didattico per esercitare i bambini nella compitazione delle parole. Possiede un vocabolario di qualche centinaio di parole ed una tastiera alfabetica (dotata di alcuni tasti addizionali) simile a quella dello ZX81. Premendo il tasto Enquiry, l'apparecchio emette un segnale acustico, invitando, a voce, a compitare una parola. Ogni lettera battuta sulla tastiera viene visualizzata da un LED (diodo fotoemittente) e, una volta completata l'immissione, lo Speak & Spell informa, sempre mediante il sintetizzatore vocale, se la parola è stata scritta correttamente oppure no.

Lo Speak & Maths funziona in modo simile, ma propone problemi aritmetici. Questi due prodotti altamente innovativi si sono conquistati una larga fetta di mercato nel campo dei giocattoli didattici ed hanno spinto la Texas Instruments verso un settore tecnologico molto diverso da quello della componentistica elettronica da cui era partita.

Un terzo giocattolo parlante di questa casa, il Touch & Tell, ha forse finalità più ricreative che didattiche. È composto da un certo numero di lamine plastiche stratificate, su ciascuna delle quali è riprodotta una figura, identificata da un codice magnetico. Quando il bambino tocca una particolare area del disegno, il Touch & Tell pronuncia in modo udibile il nome della figura selezionata.

Mentre la sintetizzazione vocale è certamente la tecnica elettronica più sofisticata usata dai fabbricanti di giocattoli e di apparecchi da gioco, il campo di applicazione più vasto è quello delle versioni miniaturizzate dei più famosi giochi da sala giochi. Nel campo dei giocattoli, un altro settore in cui i microprocessori hanno conquistato un vasto mercato è quello dei modelli di macchine e camion semoventi. Il più conosciuto è forse il Big Track, che viene programmato mediante una speciale tastiera a cuscinetto posta nella sua parte superiore. Questo giocattolo ricorda una "tartaruga" (vedere a pag. 176) e può essere guidato anche da un microcomputer attraverso la porta parallela di cui è dotato.

Ci sono altri giocattoli basati su microprocessori: il Simon, che chiede al bambino di ripetere una sequenza di note musicali e luci lampeggianti; il Maximus della Playskool, un istruttore aritmetico simile al Little Professor e tutta una varietà di piccoli robot. Esistono anche dei giocattoli per ragazzi più grandi (ed adulti): l'Electroni-Kit, il Mykit System ed il radionics. Questi, come i nomi suggeriscono, sono kit di costruzione elettronica che utilizzano componenti sigillate in materiale plastico da innestare su di una speciale piastra, in modo da assemblare una varietà di semplici apparecchi funzionanti.



IAN McKINNELL

Electroni-Kit

L'Electroni-Kit è una scatola di montaggio con la quale è possibile fabbricare vari apparecchi elettronici come radio a transistor, amplificatori e così via. I vari componenti sono sigillati in blocchetti di plastica trasparente e vengono innestati su di una speciale piastra (seguendo gli schemi forniti) per ottenere la configurazione desiderata. Il prodotto più sofisticato di questa serie permette la fabbricazione di un rudimentale microcomputer, progettato per insegnare le più semplici operazioni in codice macchina. Tuttavia, questo piccolo apparecchio, con appena 96 byte di memoria, non può essere considerato un vero e proprio home computer



per concessione della PERSONAL COMPUTER WORLD

Maximus
Questo è un prodotto molto sofisticato, fabbricato dalla MB Electronics. Somiglia vagamente ad una calcolatrice, ma in realtà è un gioco basato su di una serie di confronti, come il Simon. Il Maximus può tuttavia seguire procedure differenziate che consentono di confrontare l'ortografia delle parole, note musicali, immagini, parole in rima e forme varie



per concessione della MILTON BRADLEY LTD



Il Simon
Il Simon è una versione elettronica del gioco "Simon dice", molto popolare in Gran Bretagna. Questo apparecchio produce una sequenza casuale di note musicali, ciascuna abbinata ad una luce lampeggiante. I quadranti colorati, posti nella sua parte superiore, si comportano come interruttori sia per i toni che per la luce. L'obiettivo del gioco è riuscire a ripetere in modo corretto la sequenza prodotta dall'apparecchio, via via più complessa



Texas Instruments
La TI è entrata nel mercato dei prodotti didattici verso la metà degli anni '70 con il Little Professor, una specie di calcolatrice, in grado di proporre semplici problemi aritmetici. Alla fine degli anni '70 la Texas Instruments ha prodotto una sorta di "istruttore ortografico" che utilizza un chip di sintesi vocale: premendo un tasto lo Speak & Spell chiede al bambino di compitare una parola sulla tastiera. Più recentemente queste tecniche sono state applicate a semplici giochi aritmetici e ad apparecchi, per i più piccini, che narrano brevi fiabe



per concessione della TEXAS INSTRUMENTS

Robo-1

Il Robo-1 della Tomy è un braccio robot, simile a quelli industriali, ma controllato da due joystick e non sotto il controllo di un programma. Il suo dato più significativo è il costo: meno del 10% del più economico braccio robot didattico da computer. Naturalmente la sua struttura è molto meno robusta: è realizzato in materiale plastico anziché in metallo. Il funzionamento del braccio viene controllato visivamente e non vengono usati precisi motori "passo passo". Tuttavia, con un po' d'ingegno, è possibile collegarlo ad un home computer

TOMY ROBO 1 per concessione della HAMLEYS

Big Track

Benché somigli più o meno ad un robusto carro armato giocattolo, il Big Track è, in realtà, un robot da pavimento sotto mentite spoglie. Completamente autonomo, viene programmato impostando i codici di distanza e di direzione mediante una speciale tastiera posta nella sua parte superiore. Con una piccola modifica è possibile anche collegarlo ad un computer, usando una porta seriale o parallela. Il veicolo può quindi essere guidato da un programma di controllo anche piuttosto sofisticato

per concessione della MILTON BRADLEY LTD

IAN McKINNELL

Gli "spelling checker"

Esistono numerosi programmi per la verifica automatica dell'ortografia e della compitazione delle parole

Una delle applicazioni previste per i computer della quinta generazione, che dovrebbero apparire intorno al 1990, consiste nella traduzione automatica di testi da una lingua all'altra, per esempio dall'inglese al giapponese. La possibilità di effettuare traduzioni, tramite macchine, esiste già per testi relativamente semplici (ad esempio rapporti governativi ed atti di congressi) anche se la prima stesura prodotta dal computer deve essere necessariamente riveduta e corretta a mano. Le esperienze in questo campo hanno spesso portato a spassosi errori: si racconta che la frase "Lo spirito è forte, ma la carne è debole", tradotta dall'inglese al russo e poi nuovamente in inglese da due diversi programmi, abbia dato il seguente risultato: "Il vino è gradevole, ma la carne è guasta!".
Storie di questo tipo mettono in evidenza un importante problema: le difficoltà che un computer incontra quando elabora dati senza comprenderne il significato. Agli studenti d'informatica anglosassoni viene spesso posto il quesito su come un computer potrebbe interpretare il significato delle seguenti frasi:

TIMES FLIES LIKE AN ARROW (il tempo vola come una freccia)
WHITE FLIES LIKE A BANANA (alle mosche bianche piace una banana)

Nonostante il significato di queste due frasi sia totalmente diverso, la loro costruzione è identica, ma nel primo caso FLIES è un verbo (vola), mentre nella seconda è un sostantivo (mosche). L'interpretazione si basa sulla nostra personale esperienza. Disponendo di un'ampia memoria, è possibile dotare un computer di un'esperienza (ovviamente simulata): di ciò si occupa il campo dell'intelligenza artificiale e le ricerche in questo settore sono ancora in via di sviluppo. Il vero nocciolo della questione è la differenza tra sintassi e semantica. La sintassi regola la costruzione delle frasi in un linguaggio ed è una materia che possiamo insegnare, abbastanza facilmente, ai computer, come ben sanno tutti i programmatori in BASIC che spesso incontrano il messaggio SYNTAX ERROR! La semantica, invece, concerne l'interpretazione da dare alle frasi ed alla loro costruzione.
Nel 1950, Noam Chomsky sviluppò le basi dell'attuale teoria sui linguaggi umani e sulle relative regole grammaticali. Pur non essendo direttamente interessato ad applicazioni computerizzate, egli formulò delle teorie che si sono dimostrate incredibilmente attinenti sia alla traduzione mediante computer, che alla scrittura di interpreti e compilatori per i linguaggi di programmazione.
Uno dei sottoprodotti di questa ricerca è stata la creazione di vari strumenti software di sussidio nell'elaborazione di testi. Oltre ai word processor, usati nella creazione, redazione e stampa di un testo, esistono specifici programmi di correzione sia per l'analisi ortografica e la stampa degli errori, sia per la verifica grammaticale e stilistica di brani scritti mediante word processor. Anche se nessuno di questi prodotti appartiene, in senso stretto, alla sfera dell'intelligenza artificiale, è tuttavia istruttivo studiare i principi di funzionamento di questi programmi.
Tutti i programmi di verifica ortografica si avvalgono di un dizionario, memorizzato su disco, che generalmente contiene dalle 25.000 alle 50.000 parole. Molti di questi pacchetti software permettono di aggiungere nuove parole al dizionario, ad esempio, termini di uso non comune, ma necessari in un particolare campo; nomi di ditte o di prodotti di cui si desidera avere una verifica ortografica e via dicendo. La lingua inglese si adatta particolarmente bene a programmi di questo genere, ma sono in corso tentativi di applicare i medesimi metodi anche ad altre lingue.
Sorge, tuttavia, il problema di trovare unità di memoria capaci di contenere un intero dizionario. Ricordiamo qui che, usando il codice ASCII, un byte (8 bit) può contenere un solo carattere alfanumerico. Così, anche ipotizzando una ottimistica media di 5 caratteri per parola, un dizionario di 30.000 parole occuperà 150 Kbyte di memoria: spazio molto superiore alle capacità della maggior parte delle unità a disco usate sugli home computer. Fortunatamente, questo tipo di dati può essere facilmente "compresso" mediante due tecniche principali.
Nella prima, si presume che il dizionario contenga solo lettere minuscole e che i dati numerici ed alcuni simboli di punteggiatura non siano necessari. In questo modo, è possibile costruire un dizionario usando solo 32 differenti caratteri, invece dell'abituale gamma ASCII di 128. Inoltre, è possibile codificare questo insieme ridotto di caratteri, usando codici di soli 5 bit: la parola "computer", ad esempio, viene memorizzata in 40 bit, cioè 5 byte. I primi 5 bit del primo byte conterranno il codice della lettera c, gli ultimi 3 bit del primo byte ed i primi 2 del secondo il codice della lettera o, e così via.
La seconda tecnica, utilizzata anche nei linguaggi di programmazione, viene chiamata

"tokenising" e si basa sulla premessa che alcune combinazioni di caratteri sono così frequenti da poter essere rappresentate mediante simboli che occupano un solo byte di memoria. Questi simboli vengono in qualche modo evidenziati per distinguerli dai caratteri normali. Quasi tutti i BASIC per home e personal computer adottano questa tecnica: tutte le parole chiave, come PRINT o NEXT, vengono memorizzate nella RAM in un singolo byte per risparmiare memoria.
Il programma VizaSpell, impiegato assieme al word processor VizaWrite sul Commodore 64, usa sia la compressione che il tokenising per conservare su disco le 30.000 parole del dizionario, occupando appena 65 Kbyte.
Il compito più laborioso, per un verificatore ortografico, resta tuttavia il confronto di tutte le parole di un testo con quelle presenti nel dizionario. Si potrebbe utilizzare una procedura di ricerca binaria (vedere a pag. 416), ma, anche per testi di solo un migliaio di parole, ciò richiederebbe molte ore di lavoro.
In linea teorica, la correttezza delle parole potrebbe esser verificata dallo stesso word processor, durante la battitura del testo. Ma anche in questo caso i tempi sarebbero inaccettabili oltre a comportare notevoli difficoltà in termini di programmazione. I testi, perciò, devono essere verificati nella loro interezza: il programma esamina il documento, parola per parola, compilando un elenco alfabetico. Molti verificatori ortografici sono anche in grado di produrre un'utile indagine analitica sull'uso delle parole nel testo preso in esame: ciò può servire per eliminare fastidiose ripetizioni. Questo processo riduce al minimo i tempi di verifica, rendendolo quasi costante: circa 4 minuti (nel caso del VizaSpell), indipendentemente dalla lunghezza del testo.

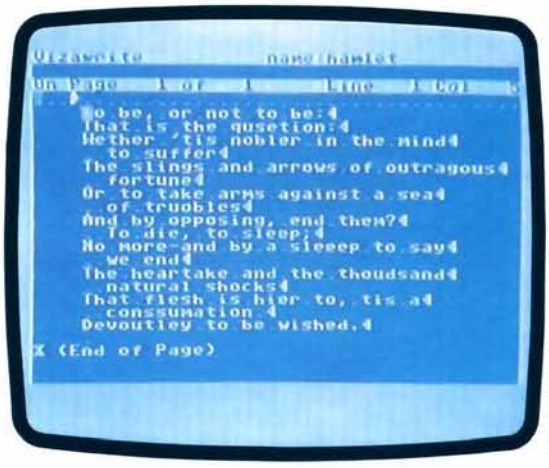


Le parole non presenti nel dizionario vengono raggruppate in un elenco a parte oppure evidenziate nel testo originale. Per ciascuna di esse l'utente ha tre possibilità a disposizione:

1) La parola contiene un errore di battitura e quindi deve essere corretta;
2) La parola è corretta e deve soltanto essere aggiunta al dizionario;
3) La parola è corretta, ma è improbabile che venga usata di nuovo, quindi non occorre aggiungerla al dizionario.


POPPERFOTO

I verificatori di grammatica e di stile funzionano in modo molto simile. I primi lavorano in base ad un limitato numero di regole (ad esempio, controllano che vi sia la lettera maiuscola all'inizio di ogni frase): può accadere, perciò, che alcuni errori grammaticali non vengano individuati. I verificatori di stile sono ancora allo stadio sperimentale, e, per il momento, si limitano a confrontare la formale correttezza di un testo con un discreto numero di frasi, ricorrenti nel linguaggio, conservate in una sorta di dizionario. Generalmente, i programmi di questo tipo suggeriscono costruzioni di frasi più lineari di quelle che si trovano nel testo in esame, oppure evidenziano un uso eccessivamente ripetitivo di frasi identiche o la presenza di periodi lunghi e poco eleganti.
Scrivere semplici programmi di verifica per l'ortografia, la grammatica e lo stile, in linguaggio BASIC, può essere un esercizio interessante anche per programmatori poco esperti, sebbene sia necessaria un'approfondita conoscenza delle funzioni che permettono di manipolare le "stringhe" di caratteri.



IAN McKINNELL

Essere o non essere

Per illustrare il funzionamento di un programma verificatore, il VizaSpell, prendiamo come esempio il monologo di Amleto. Innanzi tutto il testo viene inserito nel computer usando un word processor. Quindi, con qualche semplice comando, viene messo in funzione il verificatore: esso stampa un elenco alfabetico di tutte le parole incontrate, indicandone anche la frequenza. Questo elenco viene poi confrontato col dizionario memorizzato su disco e vengono segnalate le parole non riconosciute. Quando il VizaSpell viene usato per la prima volta, può darsi che il programma segnali anche parole di uso piuttosto frequente: basta allora inserirle nel dizionario

Programmi generatori specializzati

La generazione automatica di programmi applicativi di impiego generico



IAN McKINNELL

Progettiamo un flipper

Il "Pinball Construction Set" è un tipico esempio di "generatore di applicazioni", orientato ai giochi: l'utente progetta l'impostazione logica e grafica di un flipper, selezionando da un menu i vari componenti da includere sul tavolo del flipper

Abbiamo già esaminato (vedere a pag. 384) un particolare genere di programmi per computer che, partendo da una serie di specifiche impartite dall'utente, producono un programma adatto all'applicazione desiderata. Simili "generatori di programmi" sono disponibili per molte marche di microcomputer ad uso commerciale, ma ne esiste anche qualche versione semplificata, adatta alla fascia degli home computer, purché dotati di almeno un'unità a dischetti.

Un modo molto più comune di generare programmi prevede l'uso di pacchetti chiamati "generatori d'applicazione". I programmi prodotti sono però parzialmente dipendenti dal particolare generatore. Ossia, il programma generato deve risiedere nella memoria del computer contemporaneamente al programma generatore. Per meglio evidenziare le differenze tra le due famiglie di generatori, consideriamo la creazione di un programma per la fatturazione.

Usando un generatore di programmi è necessario, innanzitutto, rispondere a tutta una serie di quesiti relativi ai file, ai record, ai campi, alle relazioni matematiche, alle immagini da visualizzare sullo schermo ed al formato dei tabulati di stampa richiesti. A questo punto, il generatore invita ad inserire nell'unità a dischetti un disco vuoto, salvando su di esso il programma generato. Questo procedimento può essere ripetuto, consentendo, ad esempio, di preparare una copia del programma di fatturazione per ciascun reparto della ditta. Viceversa, i generatori di applicazioni, in un primo momento, sembrano dare risultati meno soddisfacenti. Completata la procedura di definizione, più o meno identica a quella appena descritta, le routine necessarie vengono registrate, generalmente, sul medesimo dischetto sul quale si trova il generatore. Anche utilizzando un secondo dischetto, la presenza del programma generatore è indispensabile al funzionamento del programma generato. Si può usare una sola copia del pacchetto originale per produrre numerose applicazioni di diverso tipo, ma queste dovranno sempre essere usate congiuntamente al generatore. Quindi, è possibile, vendere (o regalare) un programma generato soltanto se chi lo riceve possiede un generatore identico. Ovviamente, i generatori sono opportunamente protetti, dalle case produttrici, per rendere molto difficoltosa la produzione di copie non autorizzate.

Un generatore di applicazioni, tutto sommato, non è che un sofisticato programma di uso generico: l'inserimento delle specifiche relative ad una particolare applicazione consiste nell'assegnazione di opportuni valori ad una serie di importanti variabili, contenute nel generatore, che prendono il nome di "parametri". Questi controllano il flusso del programma, la struttura dei dati, le immagini da visualizzare e quelle da stampare. In realtà, quando una particolare applicazione viene memorizzata su dischetto, non viene registrato altro che un elenco completo di questi parametri o variabili (detto anche "modulo d'applicazione").

Alcuni generatori sono ancora più sofisticati, consentendo l'uso di uno pseudo-linguaggio di programmazione ad alto livello (simile a quello adoperato nel corso di Programmazione Basic). In tal caso, lo pseudo-programma viene interpretato dal generatore, producendo automaticamente l'elenco dei parametri, in seguito utilizzato dall'interprete BASIC. Questo procedimento, detto per inciso, obbliga a rivedere i concetti di gerarchia del software esposti a pag. 66.

È abbastanza frequente che i moduli d'applicazione vengano prodotti e commercializzati da case diverse da quelle che hanno progettato il generatore originale. Ad esempio, il

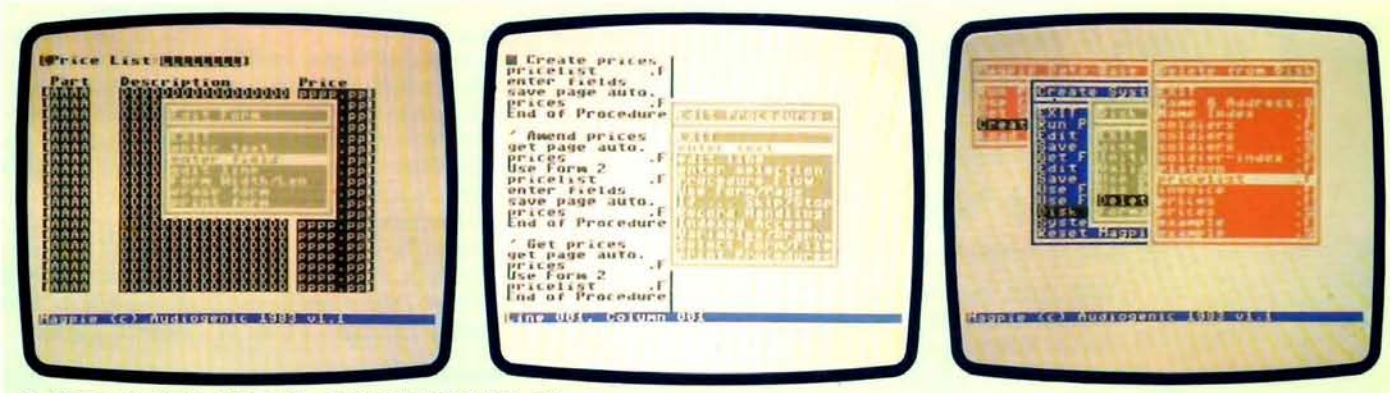
dBase II (uno dei più diffusi e sofisticati programmi per la gestione di database su microcomputer) può esser considerato, in effetti, un generatore di applicazioni, contenente moduli composti da stringhe di comandi di alto livello, tipici per la gestione di database. Con il dBase II è possibile costruire moduli per applicazioni di carattere molto particolare (come ad esempio un sistema di contabilità per agenti di cambio) senza dover scrivere l'intero programma partendo da zero. Considerando la limitata richiesta, sul mercato, di simili applicazioni, un tale pacchetto di contabilità, che gira sotto dBase II, darà migliori risultati di uno scritto in BASIC, in quanto il programmatore può dedicare tutta la propria attenzione alla struttura ed al funzionamento del programma, piuttosto che ai dettagli relativi alla stesura del listato in codice. Le parti del programma più soggette a difetti (ad esempio, la gestione dei file) sono, difatti, scritte dagli autori del generatore e verificate in applicazioni di diverso tipo da migliaia di utenti.

La differenza fondamentale tra generatori di programmi e generatori di applicazioni risiede nel diverso grado di facilità d'impiego. Un programma definitivo, creato da un pacchetto software del primo tipo, viene codificato in modo interamente artificiale (probabilmente in un linguaggio come il BASIC) e risulta quasi sempre di qualità inferiore, per efficienza e stile, rispetto ad un programma stilato da un programmatore umano. Viceversa, con un generatore di applicazioni, addirittura il 99% del programma finale risulta, in pratica, scritto dalla Software House produttrice, probabilmente in codice macchina. È questo il caso del "Silicon Office", uno dei generatori di applicazioni più sofisticati e facili da usare, disponibile per microcomputer ad uso commerciale. I programmi generati si dimostrano molto rapidi e funzionali, dotati di procedure di verifica (per individuare errori dovuti all'operatore). Inoltre, le informazioni vengono presentate con chiarezza, mediante procedure "pilotate" da menu.

Oltretutto, i generatori applicativi non si limitano a realizzare programmi di carattere commerciale. Il miglior esempio di pacchetti non finalizzati ad usi commerciali è il Pinball Construction Set (vedere a pag. 241), per la costruzione di flipper computerizzati, nel quale la formulazione del modulo d'applicazione viene efficacemente facilitata, visualizzando gli elementi in dotazione per la costruzione del flipper.

Effettivamente, la linea di separazione, tra i generatori di programmi e la programmazione orientata, è molto vaga. Abbiamo già affrontato l'argomento (vedere a pag. 242), ma possiamo qui sintetizzarlo brevemente: incoraggiare il programmatore ad implementare le proprie applicazioni specificando esclusivamente gli obiettivi del programma. Un numero sempre più crescente di pacchetti software, pur non facendo parte in senso stretto dei generatori di applicazioni, condividono con questi alcuni aspetti fondamentali. Questi programmi, "pilotati" da parametri, usati per la prima volta, rivolgono all'utente una serie di domande e registrano su disco le relative risposte. Queste informazioni determineranno alcune caratteristiche di funzionamento del programma. Un programma di fatturazione, ad esempio, rivolgerà domande sui dati che la ditta ritiene necessario includere in ogni fattura oppure sui periodi standard di credito che essa concede. Un programma di gioco, invece, chiederà con quanti alieni, basi e missili, il giocatore desidera iniziare il gioco oppure darà la facoltà di progettare la forma grafica da dare agli invasori.

L'attuale tendenza è di offrire all'utente un software di un livello tale da evitare l'apprendimento di linguaggi di programmazione, pur mantenendo un alto grado di flessibilità nel funzionamento. Un obiettivo a cui si punta con decisione è quello di produrre software capace di adattarsi alle esigenze dell'utente, per non costringere quest'ultimo ad adeguarsi, faticosamente e dispendiosamente, al software esistente.



IL "MAGPIE" PER IL COMMODORE 64

IAN McKINNELL

per concessione della SOFT

La prima fase, nella creazione di un'applicazione commerciale, consiste nello specificare l'impostazione grafica delle immagini da visualizzare (siano transazioni, bollettini o un listino prezzi). Inserendo nelle varie colonne speciali lettere (A, D, P), l'utente indica al programma i campi da usare nel database. Poi, vengono specificate le relazioni matematiche e le caratteristiche di elaborazione da applicare ai campi, sotto forma di una lista d'istruzioni in un linguaggio di programmazione di alto livello, in seguito interpretata dal Magpie. In figura vengono mostrate le fasi relative alla lettura di un listino prezzi da disco ed alla modifica di alcuni prezzi. Il funzionamento è basato sull'impiego di menu: quando viene selezionata un'opzione (ad esempio, CREATE), accanto al precedente apparirà un nuovo menu, che mostra tutte le opzioni di CREATE

Suoni incredibili

Il comando ENVELOPE, nel Micro BBC Modello B

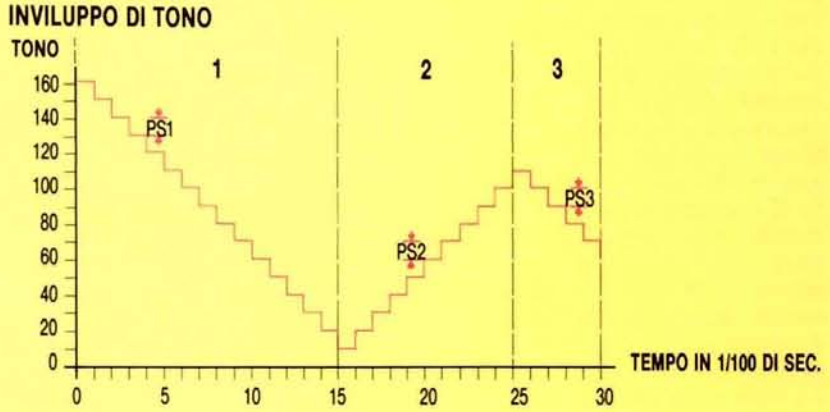
Abbiamo già esaminato il formato del comando SOUND del Micro BBC. Tuttavia, le capacità sonore e musicali di questo microcomputer si manifestano appieno soltanto quando, all'impiego del comando SOUND, viene affiancato quello del poliedrico comando ENVELOPE (inviluppo). Quest'ultimo permette di fondere 4 suoni elementari, creando così una vasta gamma di eccellenti riproduzioni di strumenti musicali tradizionali. Inoltre, ciò consente di perfezionare gli effetti sonori nei giochi, rendendo più realistiche le esplosioni e le sparatorie che spesso compaiono in essi. Il comando ENVELOPE ha la seguente struttura:

ENVELOPE N, T, PS1, PS2, PS3, NS1, NS2, NS3, AR, DR, SR, RR, FAL, FDL

Il primo parametro (N) costituisce il numero identificativo dell'inviluppo, che serve per i comandi SOUND e SOUND &. Un inviluppo, tra quattro selezionabili, può sostituire il parametro di volume fisso (V), selezionabile nel comando SOUND, mediante un numero negativo (da 0 a −15, vedere a pag. 392).

T (da 0 a 127) & (da 128 a 255)

Questo è il controllo principale di temporizzazione per il comando. Seleziona la durata di ogni intervallo nella costruzione dell'inviluppo, scandendo il tempo in centesimi di secondo. Quindi T=5 significa che ogni intervallo d'inviluppo avrà una durata di 5 centesimi di secondo (0,05 secondi). Aggiungendo 128 a questo valore, verrà soppressa la ripetizione automatica dell'inviluppo di tono per cui, attribuendo a T il valore di 5+128=133,



avremo una durata d'intervallo di 5 centesimi di secondo in un inviluppo di tono che si verifica una sola volta in ogni nota.
Il termine inviluppo "di tono" non deve confondersi con l'inviluppo "di volume", precedentemente usato, poiché nel primo è il tono della nota a variare, nel secondo è il volume. L'inviluppo di tono ha scarsa utilità in termini musicali a meno che non si desideri suonare un "vibrato", ma può essere utile per creare interessanti effetti sonori "trillati". Come mostrato nel grafico, l'inviluppo di

Luci sfolgoranti

L'uso di "sprite" sul Commodore 64

Una delle caratteristiche più interessanti del Commodore 64 è la grafica "a sprite". Gli sprite vengono costruiti con lo stesso procedimento usato per definire nuovi caratteri, sebbene le dimensioni siano spesso molto maggiori, avendo a disposizione, infatti, una matrice di 21 × 24 pixel. La visualizzazione degli sprite, però, non avviene secondo lo stesso metodo usato per i caratteri normali: è infatti possibile far muovere anche un solo pixel alla volta, ottenendo effetti grafici di gran lunga più completi. Inoltre, si possono visualizzare fino a 8 sprite simultaneamente, ciascuno dei quali è programmabile, indipendentemente dagli altri, nelle seguenti caratteristiche:

Forma e colore. Sono necessari 63 byte per codificare il disegno di uno spirte in forma binaria. Ad ogni sprite è associato un "puntatore", che contiene l'indirizzo del blocco di 63 byte consecutivi, ove è definita la sua forma. Ciò permette a più sprite di condividere una stessa zona di memoria, ad esempio, nel caso in cui due (o più) sprite siano identici. Inoltre, la forma dello sprite può esser cambiata semplicemente modificando il valore del puntatore, in modo da indirizzare una diversa zona di memoria.
Ogni sprite può assumere uno dei 16 colori disponibili, ma può anche esser composto da più colori, nel qual caso la risoluzione orizzontale si riduce, però, di circa la metà.

Dimensioni e movimento. La forma di uno sprite può esser espansa orizzontalmente, verticalmente, o in entrambe le direzioni simultaneamente, fino a raddoppiarne le dimensioni originarie. La massima dimensione ottenibile è di 48 × 42 pixel. Anche in questo caso c'è un prezzo da pagare: la capacità di risoluzione si riduce della metà nella direzione in cui si è verificata l'espansione.

tono è suddiviso in tre sezioni. Le caratteristiche di ciascuna sezione si definiscono assegnando opportuni valori ai parametri PS ed NS nel seguente modo:

PS1, PS2 & PS3 (da −128 a 128)

PS si riferisce all'intervallo di tono. Quando ha inizio l'emissione del suono associata a questo comando, il suo tono viene selezionato dal comando SOUND. PS1 definisce il cambiamento di tono positivo o negativo per intervallo nella prima sezione, PS2 vale per la seconda e PS3 per la terza. Come il comando SOUND, PS è suddiviso in "quarti di semitono".

NS1, NS2 & NS3 (da 0 a 255)

NS è relativo al numero d'intervalli per sezione ed assieme a PS definisce il ritmo di cambiamento del tono in una sezione ed anche la durata dell'intero inviluppo di tono. I valori di PS ed NS per l'esempio precedente sono i seguenti:

T=1 PS1 = −10 NS1 = 15
PS2 = +10 NS2 = 10
PS3 = −15 NS3 = 5

In questo caso il tono viene selezionato da SOUND=160. Da ciò risulta:

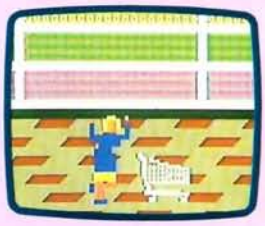
ENVELOPE 1,1,−10,10,−15,15,10,5,0,0,0,0,0,0

La durata dell'inviluppo è data dalla formula (NS1+NS2+NS3)×T che, nel caso in questione, si traduce in: (15+10+5)×1 = 0,3 secondi. Di solito l'inviluppo di tono si ripeterà automaticamente per tutta la durata della nota a meno che questo effetto non venga disattivato dal parametro di temporizzazione T.
Nella prossima parte di Suoni e Luci torneremo ancora sulle capacità del Micro BBC e descriveremo il funzionamento dell'inviluppo di volume.

Priorità e collisione. Per illustrare il concetto di priorità ricorriamo ad un semplice esempio. Supponiamo di aver definito due sprite: un treno ed una casa. Nell'animazione dell'immagine, possiamo scegliere se far passare il treno in piano collocato tra l'osservatore e la casa (la casa verrà temporaneamente nascosta al passaggio del treno) oppure in un piano situato dietro la casa (il treno scomparirà alla vista passando dietro alla casa). Ciò consente la creazione di effetti tridimensionali stupefacenti. Ad ogni sprite è associato un codice (da 0 a 7), che definisce, appunto, la priorità: uno sprite con priorità inferiore sembrerà muoversi in un piano situato tra l'osservatore e tutti i piani contenenti sprite con priorità maggiore.
Se due sprite occupano lo stesso punto dello schermo, questo evento viene segnalato ad un registro di collisione. Esaminando il contenuto di questo registro (col comando PEEK, ad esempio), il programmatore può individuare quali sprite hanno partecipato alla collisione. In un altro registro vengono memorizzate le collisioni tra gli sprite ed i caratteri dello sfondo.
Sfruttando convenientemente tutte queste informazioni, è possibile scrivere, in BASIC, efficientissimi programmi di controllo per giochi nei quali la rapidità dei movimenti grafici è di capitale importanza. Sfortunatamente, nel BASIC del Commodore non sono disponibili speciali comandi per controllare le caratteristiche degli sprite: occorre quindi ricorrere ad un largo impiego di comandi PEEK e POKE. Alternativamente, si può acquistare una speciale versione di BASIC (il "Simon's BASIC"), appositamente progettato a questo scopo.

Il "Simon's BASIC"

Con modica spesa si può acquistare una cartuccia, che permette al programmatore BASIC di estendere le capacità grafiche in alta risoluzione ed il controllo sugli sprite. La cartuccia è corredata da un manuale, nel quale vengono spiegati i 114 comandi aggiuntivi, che comprendono comandi per abilitare il modo di alta risoluzione, selezionare i colori di primo piano e di sfondo, tracciare cerchi, ellissi e triangoli. Le istruzioni per il trattamento degli sprite comprendono aiuti per creare e progettare sprite, comandi per farli apparire e sparire e procedure per posizionarli



Fase due

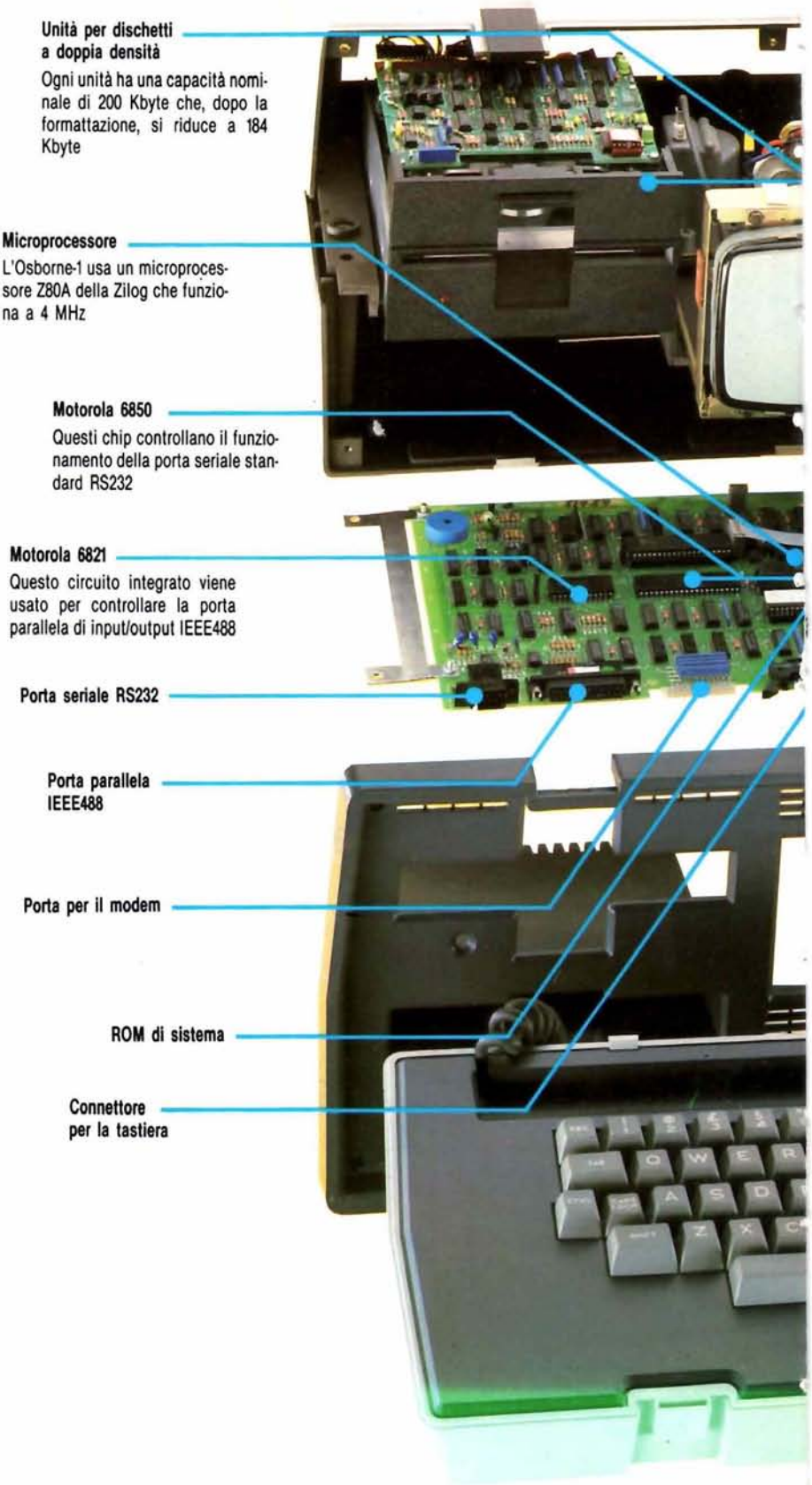
Queste linee di programma vanno aggiunte al listato riportato a pag. 359. Questa sezione usa due sprite espansi a più colori per creare una figura femminile ed un terzo sprite per visualizzare il carrello. I puntatori dello sprite vengono guidati in modo da far cambiare ripetutamente forma alla donna. Si ha quindi l'impressione che la figura danzi mentre attraversa il video. Per usare il programma "supermercato", come una subroutine per questo programma, è necessario modificare la linea 3270 così: 3270 RETURN

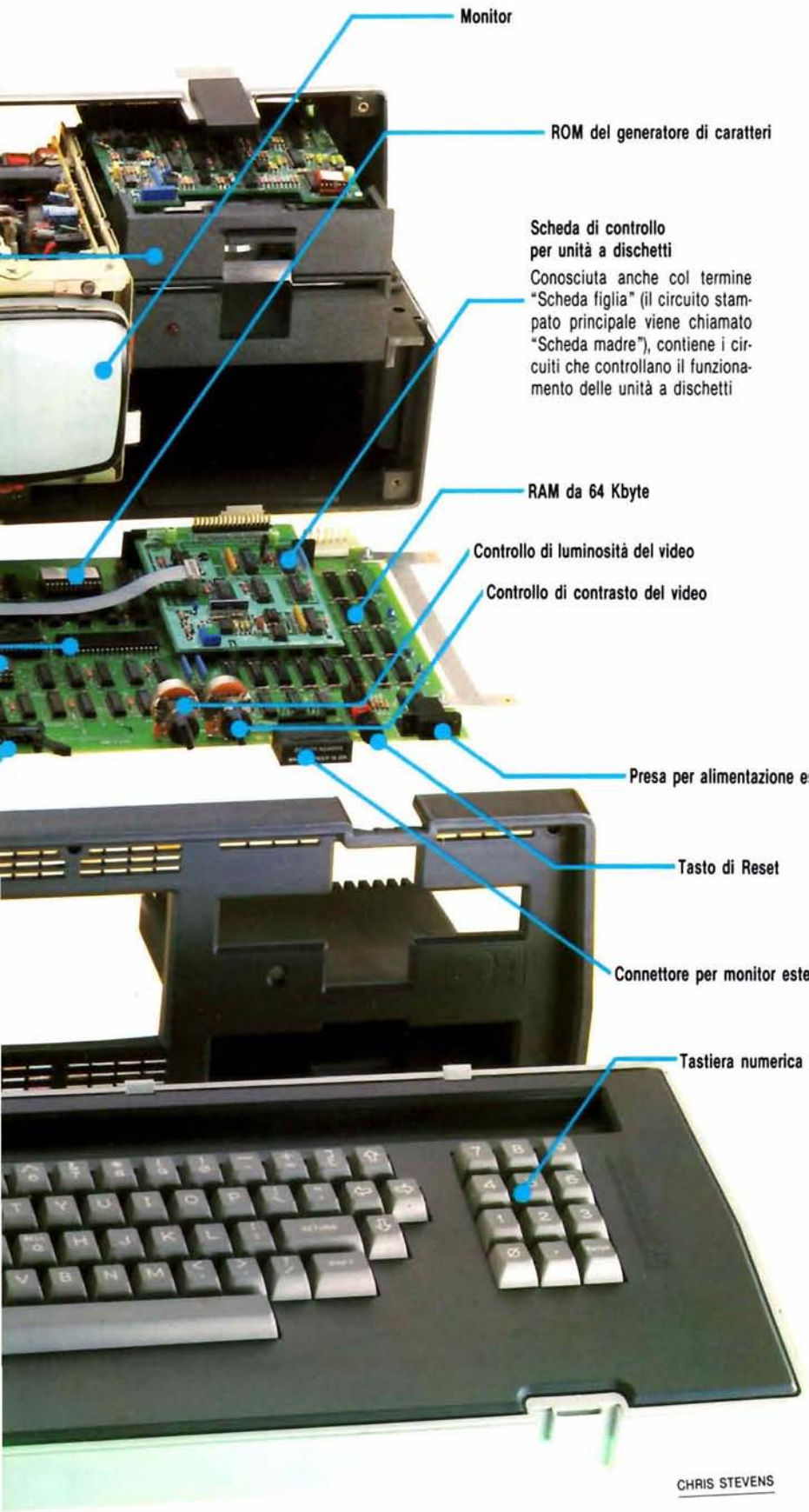
```
90 REM ** SPRITE PER IL 64 **
100 PRINT " "
10 V=53248
120 REM ---- LETTURA DATI DEGLI SPRITE ---
130 FORI=12288TO12350:READA:POKEI,A:NEXT
140 FORI=12352TO12414:READA:POKEI,A:NEXT
150 FORI=832TO894:READA:POKEI,A:NEXT
160 FORI=896TO958:READA:POKEI,A:NEXT
170 FORI=12416TO12478:READA:POKEI,A:NEXT
180 REM ---- ESPANSIONE DEGLI SPRITE ---
190 POKEV+23,7:POKEV+29,7
200 REM ---- COLORAZIONE DEGLI SPRITE ---
210 POVEV+39,10:POKEV+40,10
220 POKEV+41,1
230 REM --- COLORAZIONI MULTIPLE ---
240 POKEV+28,3 : POKEV+37,7 : POKEV+38,9
300 REM --- PUNTATORI ALLA MEMORIA ---
310 POKE2040,192 : POKE2041,193 : POKE2042,194
320 REM --- ASSE Y ---
330 YO=150 : Y1=YO+42 : Y2=YO+34
340 POKEV+1,YO : POKEV+3,Y1 : POKEV+5,Y2
400 REM --- ACCENSIONE SPRITE ---
410 POKEV+21,7
500 GOSUB3000 : REM (DA OMETTERE SE MANCA SUBROUTINE)
1000 XO=20
1010 POKE2040,13 : POKE2041,14
1020 POKEV,XO : POKEV+2,XO : POKEV+4,XO+48
1030 FORI=1TO500 : NEXT
1040 POKE2040,192 : POKE2041,193
1050 XO=XO+5
1060 POKEV,XO : POKEV+2,XO : POKEV+4,XO+48
1070 FORI=1TO500 : NEXT
1080 XO=XO+5
1090 IFXO)200THEN1110
1100 GOTO1010
1110 FORJ=1TO10
1120 POKE2040,13:POKE2041,14
1130 FORI=1TO50:NEXT
1140 POKE2040,192 : POKE2041,193
1150 FORI=1TO50:NEXT
1160 NEXT
1170 GOTO1170
9000 REM --- DATI PARTE SUPERIORE PERSONAGGIO ---
9010 DATA0,0,0,21,0,0,21,0,0,22,0,0,86,0
9020 DATA0,86,0,0,86,0,0,40,0,0,252,0
9030 DATA15,255,0,255,255,0,255,255,0
9040 DATA195,255,0,195,255,0,195,243,254
9050 DATA207,243,254
9060 DATA143,240,0,143,252,0,15,252,0
9070 DATA15,252,0,15,252,0
9100 REM --- DATI PARTE INFERIORE PERSONAGGIO ---
9110 DATA15,252,0,15,252,0,15,252,0
9120 DATA15,252,0,5,84,0,5,84,0,5,84,0
9130 DATA5,84,0,10,40,0,234,40,0,234,40,0
9140 DATA234,40,0,192,40,0,192,40,0,0,40,0
9150 DATA0,40,0,0,63,0,0,63,0,0,0,0,0,0,0
9160 DATA0,0,0
9200 REM --- DATI PARTE SUP. N.2 ---
9210 DATA0,0,0,0,20,32,32,85,32,32,105,48,48,105,48
9220 DATA48,105,48,48,105,48,48,40,48,48,252,48
9230 DATA63,255,240,63,255,240,63,255,0
9240 DATA3,255,0,3,255,0,3,240,0
9250 DATA15,240,0
9260 DATA15,240,0,15,252,0,15,252,0
9270 DATA15,252,0,15,252,0
9300 REM --- DATI PARTE INF. N.2 ---
9310 DATA15,252,0,15,252,0,15,252,0
9320 DATA15,252,0,5,84,0,5,84,0,5,84,0
9330 DATA5,84,0,10,40,0,58,168,0,58,168,0
9340 DATA58,0,0,58,0,0,10,0,0,10,0,0
9350 DATA10,0,0,15,192,0,15,192,0,0,0,0,0,0,0
9360 DATA0,0,0
9400 REM --- DATI DEL CARRELLO ---
9410 DATA192,0,0,224,0,0,118,0
9420 DATA0,55,192,0,32,60,0,53
9430 DATA87,240,32,0,15,53,85,85
9440 DATA32,0,3,53,85,85,0,0,3
9450 DATA21,85,85,31,255,255,24,0
9460 DATA0,12,0,0,12,0,0,31,255
9470 DATA240,31,255,255,1,0,2,7
9480 DATA0,14,7,0,14
```

L'Osborne-1

Anatomia di un microcomputer a diffusione europea: tecnologie a confronto

L'Osborne-1 è una macchina particolarmente interessante, in quanto è stato il primo computer portatile completamente autosufficiente. Dotato di due unità a dischetti e di un piccolo monitor, l'Osborne-1 mette a disposizione un capace sistema di elaborazione, ovunque si vada.
L'unica parte mancante, per renderlo un vero e proprio computer portatile, è un'alimentazione interna: effettivamente, una batteria, sufficiente al funzionamento autonomo, avrebbe reso eccessivo il peso dell'apparecchio che già raggiunge i 10 chili e mezzo. Il pannello anteriore, comunque, oltre ad altri connettori per interfacce, possiede una presa per la corrente continua. Il computer usa una corrente a 12 volt per le unità a dischetti ed una a 5 volt per i circuiti logici.
L'Osborne-1 è difficilmente collocabile nella categoria degli home computer, anche per quanto concerne il prezzo. Tuttavia, occorre tener presente che l'apparecchio viene venduto già corredato di un "pacchetto" di software di altissimo valore, che include il CBASIC della Microsoft (un compilatore BASIC che rende assai veloce il funzionamento dei programmi); il Supercalc, da moltissimi considerato il miglior programma di lavagna elettronica della prima generazione; il Wordstar ed il Mailmerge, il package di word processing più venduto nel mondo (può essere usato anche su altri computer). Ma la caratteristica migliore, in assoluto, è forse la presenza del sistema operativo CP/M (sigla che sta per Control Program/Monitor) della Digital Research, che permette di utilizzare, su tutte le macchine che ne fanno uso, una vasta gamma di pacchetti software.
Nell'Osborne-1, come nell'Apple II (vedere a pag. 349) ed altri microcomputer, il sistema operativo, memorizzato su dischetto, viene letto in memoria all'accensione dell'apparecchio. Il sistema CP/M, oltre a controllare il funzionamento interno del computer, dispone di un certo numero di utili comandi gestionali, coi quali è facile, ad esempio, la creazione di archivi (o interi dischetti) di copia (chiamati "back-up"), l'inizializzazione di dischetti vergini, la catalogazione del contenuto di un dischetto e così via. Il CP/M è uno dei sistemi operativi più diffusi nel settore dei microcomputer (vedere pag. a fianco). Uno dei principali motivi è dovuto al fatto che il software sviluppato sotto CP/M può essere facilmente "trasportato" su apparecchi di marche diverse (purché, ovviamente, dotate del medesimo





CHRIS STEVENS

OSBORNE-1

DIMENSIONI

mm. 510 x 325 x 225

PESO

10,5 chilogrammi

CPU

Z80A

CLOCK

4 MHz

MEMORIA

64 Kbyte di RAM e 4 Kbyte di ROM

VIDEO

24 linee di 52 caratteri visualizzabili. Dimensioni effettive dell'area video: 32 righe x 128 caratteri

INTERFACCE

Porta seriale RS232, parallela IEEE, porta per modem

LINGUAGGI FORNITI

BASIC, Assembler Z80

ALTRI LINGUAGGI DISPONIBILI

Tutti quelli disponibili sotto CP/M

ACCESSORI COMPRESI

CP/M, Wordstar, CBASIC, MBASIC, Mailmerge, Supercalc e relativi manuali, oltre a quello operativo

TASTIERA

96 tasti, disposizione tradizionale

DOCUMENTAZIONE

Adam Osborne, il creatore di questo apparecchio, ha venduto la casa editrice di cui era proprietario alla McGraw-Hill. Non sorprende perciò, data la precedente professione, l'altissimo livello nella qualità dei manuali, sempre molto chiari e sintetici.

IL CP/M DELLA DIGITAL RESEARCH

Dalla metà degli anni '60, data di nascita dei computer della seconda generazione, le grosse unità di elaborazione ed i minicomputer hanno beneficiato dell'esistenza di sistemi operativi indipendenti dalle macchine, ma è solo da una dozzina d'anni che simili sistemi di controllo sono disponibili anche per i microcomputer. Il CP/M della Digital Research è stato il primo di questi sistemi. Progettato per il microprocessore 8080 della Intel e per quelli della serie Zilog Z80, possiede una serie di utility e di programmi gestionali ed è in grado di gestire le procedure di interruzione e di riattivazione dei programmi sul computer. Un altro notevole vantaggio risiede nella possibilità di definire la struttura e la disposizione dei file che poi il CP/M gestisce. Usando un programma d'interscambio come il BSTAM, che riduce qualsiasi tipo di file alla sua forma base, è possibile trasferire con grande facilità, da una macchina all'altra, i programmi scritti sotto CP/M. È per questa ragione che gli utenti del CP/M hanno accesso ad una quantità virtualmente illimitata di software

sistema operativo). Questa particolare caratteristica offre, ai produttori di software, un mercato potenziale di vastissime dimensioni, incentivando, al tempo stesso, la creazione di programmi di qualità sempre migliore.
Per un breve periodo, nel prezzo d'acquisto dell'Osborne era incluso anche il "dBase II" della Asthon-Tate, considerato il più potente programma gestionale di database per microcomputer.
Sfortunatamente per l'Osborne, l'attenzione della maggior parte dei produttori americani si è concentrata sul Personal Computer IBM, un elaboratore a 16 bit basato sul microprocessore 8088 della Intel. Il chip 8088, inizialmente impiegato soltanto come soluzione temporanea, vanta infatti un indirizzamento a 16 bit, ma trasferisce solo 8 bit di dati alla volta. Tuttavia, esso è divenuto, di fatto, uno standard industriale, a seguito della scelta, da parte dell'IBM, di adottarlo in occasione del suo primo ingresso nel mercato dei microcomputer. Il personal computer IBM usa un sistema operativo appositamente progettato, chiamato PC-DOS. La Digital Research, per cercare di contrastare il successo della macchina americana, ha lanciato sul mercato due nuove versioni del sistema operativo CP/M: il CP/M Concorrente (che permette una reale multiprogrammazione multiutente) ed il CP/M86, progettato appositamente per il chip a 16 bit 8088 della Intel.
Tutti questi sviluppi hanno causato una graduale scomparsa dell'Osborne-1 dal mercato dei computer e, nel 1983, la Osborne Computer Corporation è andata in volontaria liquidazione. L'Osborne-1, con i suoi 64 Kbyte di memoria, 60 dei quali a disposizione dell'utente, è un elaboratore discretamente potente, dotato di porte RS232 e IEEE, di una porta per modem e della possibilità di funzionare a batteria.



Via col... computer

Oltre all'ottimo rapporto peso/prestazioni, l'Osborne-1 ha il grande vantaggio di essere un "portatile". Certo 10 chili e mezzo non sono pochi, ma questa macchina è completamente autosufficiente e ben bilanciata, quindi portarsela dietro non crea grossissimi problemi

IAN McKINNELL

Lo schermo dell'Osborne è di piccole dimensioni (3 pollici), ma il sistema di visualizzazione, simile a quello adottato anche dall'HX-20 della Epson (vedere a pag. 169), ricorre ad un "video virtuale", tre volte più grande di uno schermo di 24 righe per 52 colonne. L'uso del tasto CONTROL (una necessità standard del CP/M) e dei tasti per il movimento del cursore, permette di spostare il display sulla effettiva memoria video; ciò consente di eliminare molti svantaggi causati dalle piccole dimensioni dello schermo. La leggibilità è sufficientemente buona, considerando che i caratteri hanno un'altezza di solo 2 mm., e nella maggioranza delle applicazioni una simile miniaturizzazione non comporta grossi inconvenienti. Tuttavia, l'Osborne-1 possiede anche un connettore per un monitor esterno, nel qual caso la visualizzazione avviene secondo i normali standard. Comunque, per offrire agli utenti la possibilità di visualizzare l'intera memoria video (4 Kbyte) in un unico display di 32 righe per 128 colonne, esiste un apposito programma. L'utente può così scegliere fra tre diverse dimensioni video: 52, 96 o 128 caratteri per riga. Anche in quest'ultimo caso, nonostante la notevole densità, i caratteri risultano ben definiti e leggibili.
La tastiera dell'Osborne-1, alloggiata nel coperchio inferiore dell'apparecchio, è dotata di 69 tasti. Questi sono disposti come su di una normale macchina da scrivere, salvo l'aggiunta dei tasti Control ed Escape e di una tastierina numerica a 12 tasti, collocata sulla parte destra. Usando un programma chiamato SETUP, l'utente può assegnare ai tasti numerici nuove funzioni (fino ad un massimo di 96 caratteri per ciascun tasto) e quindi abilitarli premendoli assieme al tasto Control. Questa caratteristica è molto utile per parole, frasi o stringhe di comando d'impiego particolarmente frequente. I parametri generati dal programma SETUP, invece di essere immagazzinati in memoria, vengono scritti direttamente sul dischetto: le funzioni dei vari tasti possono essere, perciò, prestabilite in modo indipendente per ogni pacchetto software. Il computer selezionerà automaticamente le funzioni necessarie ogni volta che il sistema operativo viene caricato in memoria.
Oltre ai 96 caratteri maiuscoli e minuscoli standard, sono disponibili anche 32 caratteri grafici predefiniti, utilizzabili, però, soltanto dai programmi applicativi.
Poiché l'Osborne-1 impiega chip di supporto della serie 6800 (e non i loro equivalenti della famiglia 8080), la procedura di interrogazione ciclica della tastiera è leggermente differente. Una parte della memoria è riservata alla decodifica delle battute dei tasti e una ROM di sistema effettua controlli continui per verificare se un tasto è stato battuto. La tastiera non possiede circuiti logici propri per la decodifica ed è proprio questa caratteristica che permette di riprogrammare facilmente i tasti di funzione. Inoltre, siccome le nuove funzioni vengono registrate nella RAM, è possibile richiamarle e cambiarle anche durante l'esecuzione di un programma.

Un nuovo sort

Un metodo molto efficiente per riordinare dati

A pagina 286 abbiamo esaminato due metodi per riordinare una serie di dati: il Bubble sort e l'Insertion sort. Se, da un lato, il riordinamento "Bubble" è più semplice da realizzare, quello "ad inserzione" è, dall'altro, di più rapida esecuzione. In ambedue i casi, si può trarre un'utile indicazione: lo spostamento di dati, su distanze brevi, è il modo di procedere che impiega maggior tempo. È più conveniente spostare dati una sola volta su distanze lunghe che non più volte su distanze brevi.

```
7999 REM ***********************************************
8000 REM ************** SHELL SORT **************
8001 REM ***********************************************
8025 PRINT "PARTENZA DELLO SHELL SORT.."
8050 LET LK = LT
8100 FOR Z = 0 TO I STEP 0
8150 LET LK = INT(LK/II)
8200 FOR LB = I TO LK
8250 LET LL = LB + LK
8300 FOR P = LL TO LT STEP LK
8350 LET D = R(P)
8400 FOR Q = P TO LL STEP —LK
8450 LET R(Q) = R(Q—LK)
8500 IF D < = R(Q) THEN LET R(Q)=D:LET Q=LL
8550 NEXT Q
8600 IF D>R(LB) THEN LET R(LB)=D
8650 NEXT P
8700 NEXT LB
8750 IF LK = I THEN LET Z = I
8800 NEXT Z
8850 PRINT "FINE DELLO SHELL SORT !!"
8900 RETURN
```

Per utilizzare questa routine all'interno del programma di prova a pag. 287, cambiare le seguenti linee:

```
350 LET I=1:LET 0=0:LET II=2:LET TH=3
:
900 ON SR GOSUB 6000,7000,8000
```

Un terzo metodo, di gran lunga più efficiente, è lo "Shell sort". In esso, inizialmente, si riduce il disordine del gruppo di dati da riordinare (in modo da collocarli il più vicino possibile alla loro posizione corretta). Successivamente, si operano degli spostamenti, su distanze relativamente lunghe, per ottenere il riordino completo.
Riallacciandosi all'esempio usato per i due precedenti metodi, il funzionamento del riordinamento Shell si può così riassumere:

1) Prendiamo un gruppo di carte da un mazzo. Per semplicità, supponiamo che esse appartengano tutte allo stesso seme e che debbano esser sistemate in ordine decrescente da sinistra a destra (ossia dal Re all'Asso). Mescoliamo le carte e disponiamole in fila sul tavolo. Adesso, contiamo le carte (nel nostro caso sono 13) e dividiamo il loro numero per 2, ignorando eventuali resti della divisione. Il valore ottenuto, che chiamiamo "Link", lo annotiamo su un foglio a parte.
2) Poniamo una moneta da 50 lire sotto la carta più a sinistra (= Posizione 1) ed una moneta da 100 lire sotto la carta in posizione Link (= Posizione 6, inizialmente). Consideriamo ciascuna delle carte comprese tra la Posizione 1 e la Posizione 6, come la carta più a sinistra in una serie di "catene" di carte. Il numero di catene, pertanto, è pari al valore Link (inizialmente: 6). Le carte che appartengono ad una stessa catena si individuano ripetutamente sommando il valore di Link alla posizione della prima carta della catena, fino ad esaurire le posizioni. Nel nostro esempio, la prima catena è composta dalle carte di Posizione 1, 7 e 13; la seconda da quelle di Posizione 2 ed 8; la terza da quelle di Posizione 3 e 9 e così via. L'ultima catena è quindi formata dalle carte in Posizione 6 (valore corrente del Link) e 12.
3) Adesso, avendo marcato i limiti con le monete da 50 e da 100 lire, estraiamo dal gruppo di carte da riordinare quelle che fanno parte della prima catena. Operiamo un riordinamento su di esse, usando una delle procedure descritte a pagina 286 (Bubble o Insertion sort).
4) La catena, riordinata, viene ricollocata negli spazi rimasti vuoti. Ripetiamo nuovamente questa procedura per la catena successiva e così via finché tutte le catene, le cui carte all'estremità sinistra sono comprese tra le 50 e le 100 lire, saranno tutte riordinate.
5) Terminata la fase precedente, dividiamo il valore di Link per due (ignorando ogni resto). Se il risultante valore del Link è minore di 1, il gruppo di carte preso in esame è completamente riordinato. Altrimenti, si ripeterà la procedura a partire dal passaggio 2, utilizzando, però, il nuovo valore del Link.

LE FASI DEL RIORDINO

Posizione										Valore Link	Commenti
1	2	3	4	5	6	7	8	9			
2	8	9	3	J	5	K	6	7		(9/2) = > 4	passo iniziale
★	+	@	$	★	+	@	$	★			formazione catene
T			7				2				Sort della catena 1
	8			5							Sort della catena 2
		K			9						Sort della catena 3
			6			3					Sort della catena 4
J	8	K	6	7	5	9	3	2			passo intermedio
J	8	K	6	7	5	9	3	2		(4/2) = > 2	fine del passo
★	+	★	+	★	+	★	+	★			formazione catene
K		J		9		7		2			Sort della catena 1
	8		6		5		3				Sort della catena 2
K	8	J	6	9	5	7	3	2			fine del passo
K	8	J	6	9	5	7	3	2		(2/2) = > 1	passo iniziale
★	★	★	★	★	★	★	★	★			formazione catena 1
K	J	9	8	7	6	5	3	2			fine del passo

Lo Shell Sort

L'esempio di un riordinamento Shell per una mano ridotta, qui mostrato, evidenzia l'originalità del metodo: il gruppo di carte viene suddiviso in una serie di catene (con spaziature basate sul corrente valore del Link). Queste catene vengono riordinate separatamente, usando in questo caso il metodo "ad inserzione", prima che un passaggio venga completato.
Per confrontare i vari tipi di sort, il programma riportato qui a fianco deve essere aggiunto al programma di prova presentato a pagina 287. Una sensibile riduzione, nella durata dell'intero processo di riordino, si apprezza quando il numero degli elementi supera le 40 unità

LEGENDA	
★	Elemento della catena 1
+	Elemento della catena 2
@	Elemento della catena 3
$	Elemento della catena 4
K	Re
J	Fante

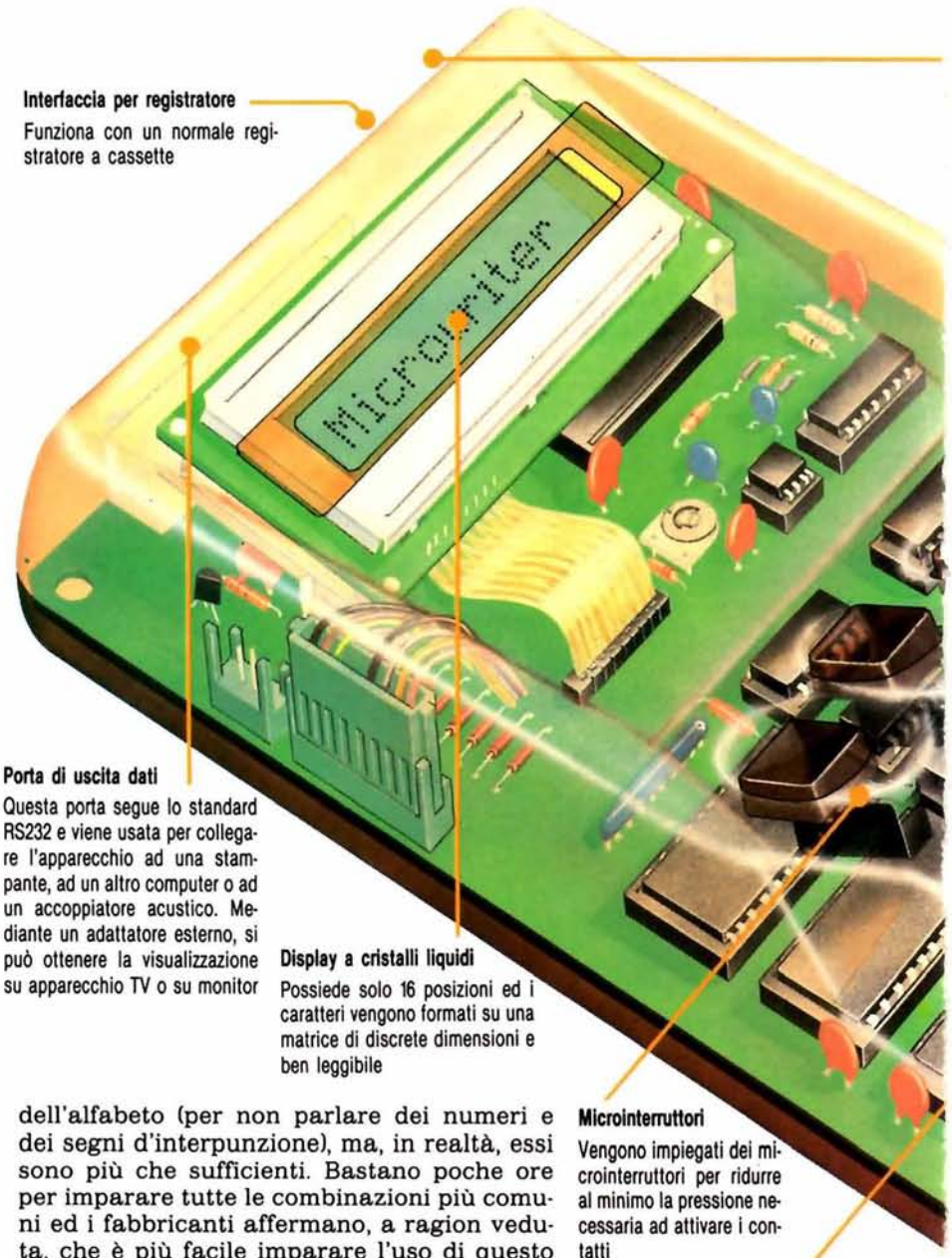
Il computer stenografo

Il Microwriter: un word processor sempre a portata di mano

I sistemi per l'elaborazione di testi, noti come "word processor", sono utilissimi in ufficio: permettono di eliminare monotone ripetizioni nella produzione di lettere e di incartamenti, servono per redigere, con ordine e precisione, serie di documenti (o copie di essi) oppure per elaborare il contenuto di un indirizzario. In effetti, i word processor si integrano talmente bene nell'attività lavorativa, che, anche quando si deve prendere un semplice appunto, le mani si dirigono automaticamente alla tastiera del computer, invece che alla normale carta e penna. Ma come fare quando occorre prendere appunti e si è lontani da casa o dall'ufficio? C'è un mercato in forte sviluppo per i computer portatili, come il Modello 100 della Tandy, l'Olivetti M10 e l'Epson HX-20. Questi hanno il vantaggio di comportarsi come word processor portatili o come terminali a distanza per sistemi di maggiori dimensioni, ma è un po' azzardato paragonarli, come maneggevolezza, ad un blocco notes o ad un dittafono. Un sistema di word processor tanto piccolo da esser portato tranquillamente in tasca, alimentato a batterie, utilizzabile con una sola mano e collegabile ad una stampante o anche ad un altro computer, è il "Microwriter", in commercio già da qualche anno.

Ideato da un americano, Cy Enfield, la caratteristica che lo distingue maggiormente è l'impiego di una speciale tastiera, munita di soli 6 tasti. Il concetto di base nacque dal desiderio di creare un apparecchio, adatto ad un gioco basato su parole, da poter tenere in mano. Una tastiera tradizionale, anche se in miniatura, sarebbe risultata troppo grande e costosa. L'ovvia soluzione era di creare una tastiera speciale, dotata di pochi tasti, ma con un numero di combinazioni sufficienti a codificare tutti i caratteri alfanumerici. Il passo risolutivo fu l'invenzione di un codice simbolico, caratteristico del Microwriter.

A prima vista sembra impossibile poter rappresentare, con sei soli tasti, tutte le lettere dell'alfabeto (per non parlare dei numeri e dei segni d'interpunzione), ma, in realtà, essi sono più che sufficienti. Bastano poche ore per imparare tutte le combinazioni più comuni ed i fabbricanti affermano, a ragion veduta, che è più facile imparare l'uso di questo apparecchio, che non quello di una tastiera QWERTY. La combinazione di tasti necessaria per ogni lettera si basa sulla forma fisica della stessa lettera e questo codice, per i meno esperti in dattilografia, si dimostra molto facile da imparare. Inoltre, poiché basta una sola mano per la battitura, il Microwriter consente l'uso di un word processor anche a persone andicappate, oppure impossibilitate ad usare le tastiere normali. Il microprocessore e la memoria sono realizzati con la tecnologia CMOS (Complementary Metal Oxide Silicon) e ciò consente un consu-



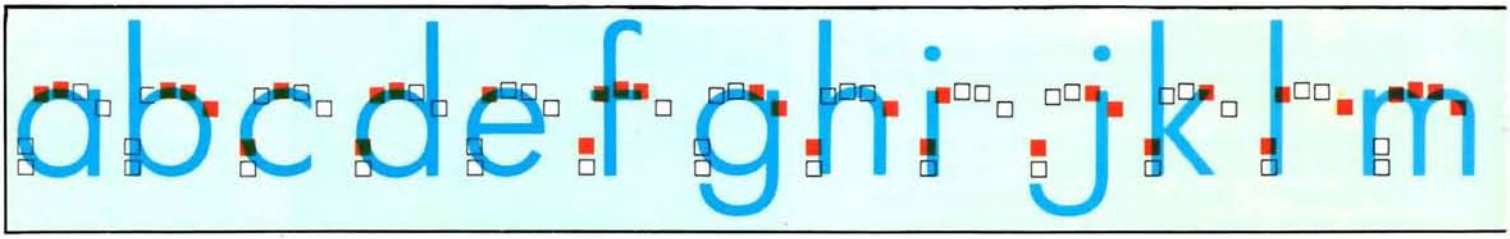
Interfaccia per registratore
Funziona con un normale registratore a cassette

Porta di uscita dati
Questa porta segue lo standard RS232 e viene usata per collegare l'apparecchio ad una stampante, ad un altro computer o ad un accoppiatore acustico. Mediante un adattatore esterno, si può ottenere la visualizzazione su apparecchio TV o su monitor

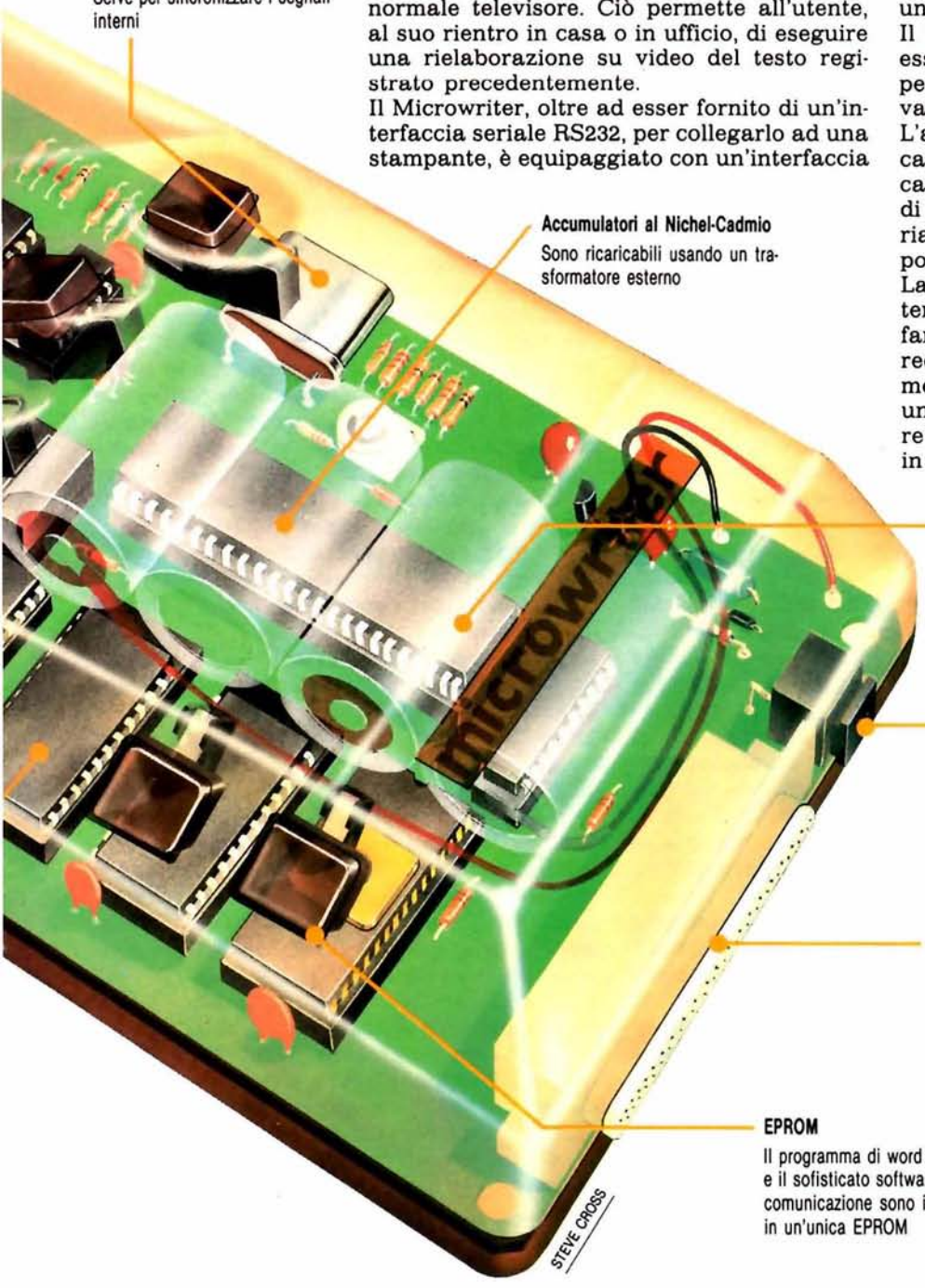
Display a cristalli liquidi
Possiede solo 16 posizioni ed i caratteri vengono formati su una matrice di discrete dimensioni e ben leggibile

Microinterruttori
Vengono impiegati dei microinterruttori per ridurre al minimo la pressione necessaria ad attivare i contatti

RAM
L'apparecchio possiede 8 Kbyte di RAM standard, ma è possibile inserire nello stesso zoccolo chip di maggiori dimensioni per accrescerne la capacità

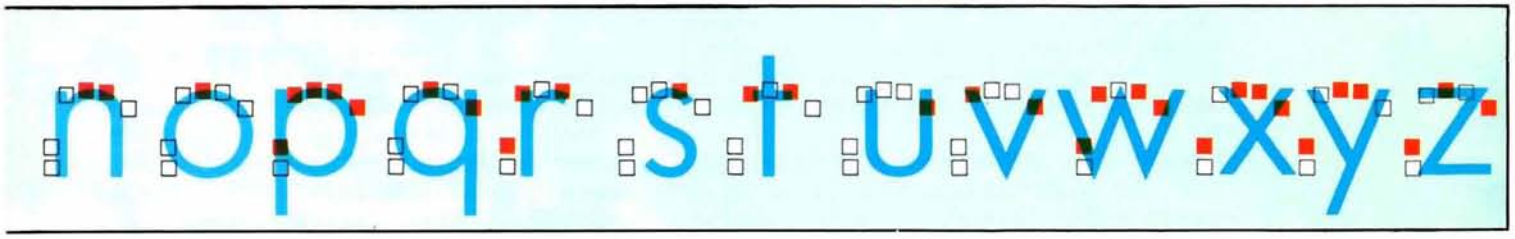


mo d'energia molto contenuto. Gli accumulatori ricaricabili al Nichel-Cadmio, incorporati, garantiscono un'autonomia di circa 30 ore. I caratteri vengono visualizzati su di un display LCD (a cristalli liquidi) che può contenere fino a 14 caratteri: questi "scorrono" verso destra via via che il testo viene battuto. Comunque, è possibile collegare, mediante una speciale interfaccia, l'apparecchio ad un normale televisore. Ciò permette all'utente, al suo rientro in casa o in ufficio, di eseguire una rielaborazione su video del testo registrato precedentemente.
Il Microwriter, oltre ad esser fornito di un'interfaccia seriale RS232, per collegarlo ad una stampante, è equipaggiato con un'interfaccia per registratore a cassetta, per memorizzare permanentemente i testi immagazzinati in memoria. L'interfaccia seriale fa sì che il Microwriter possa venir usato anche come terminale di un normale computer, cosicché i documenti battuti quando si è lontani da casa o dall'ufficio possono esser facilmente trasferiti su di un sistema di elaborazione di dimensioni e capacità maggiori, ai fini di un'ulteriore elaborazione.
Il testo immagazzinato nel Microwriter può essere suddiviso in documenti separati: ciò permette di inserire e trattare separatamente vari brani di testo non correlati tra loro. L'apparecchio è dotato anche di semplici capacità redazionali: è possibile aggiungere, cancellare o spostare porzioni di testo, anche di notevoli dimensioni, sfruttando la memoria dell'interfaccia per registratore come deposito temporaneo.
La speranza di chi ha progettato il Microwriter era che la sua tastiera potesse entrare a far parte del corredo standard di altri apparecchi elettronici ma, nonostante le sue ottime qualità, il Microwriter ha suscitato solo un limitato interesse e resta ancora da vedere se i produttori di computer utilizzeranno in futuro questo singolare prodotto.



CINQUE BASTANO

La documentazione del Microwriter comprende brani ed illustrazioni utili per apprendere automaticamente e ricordare le varie combinazioni di tasti, necessarie per codificare l'alfabeto. Il sesto tasto viene usato, congiuntamente agli altri cinque, per produrre la punteggiatura ed i comandi di redazione



LIZ DIXON

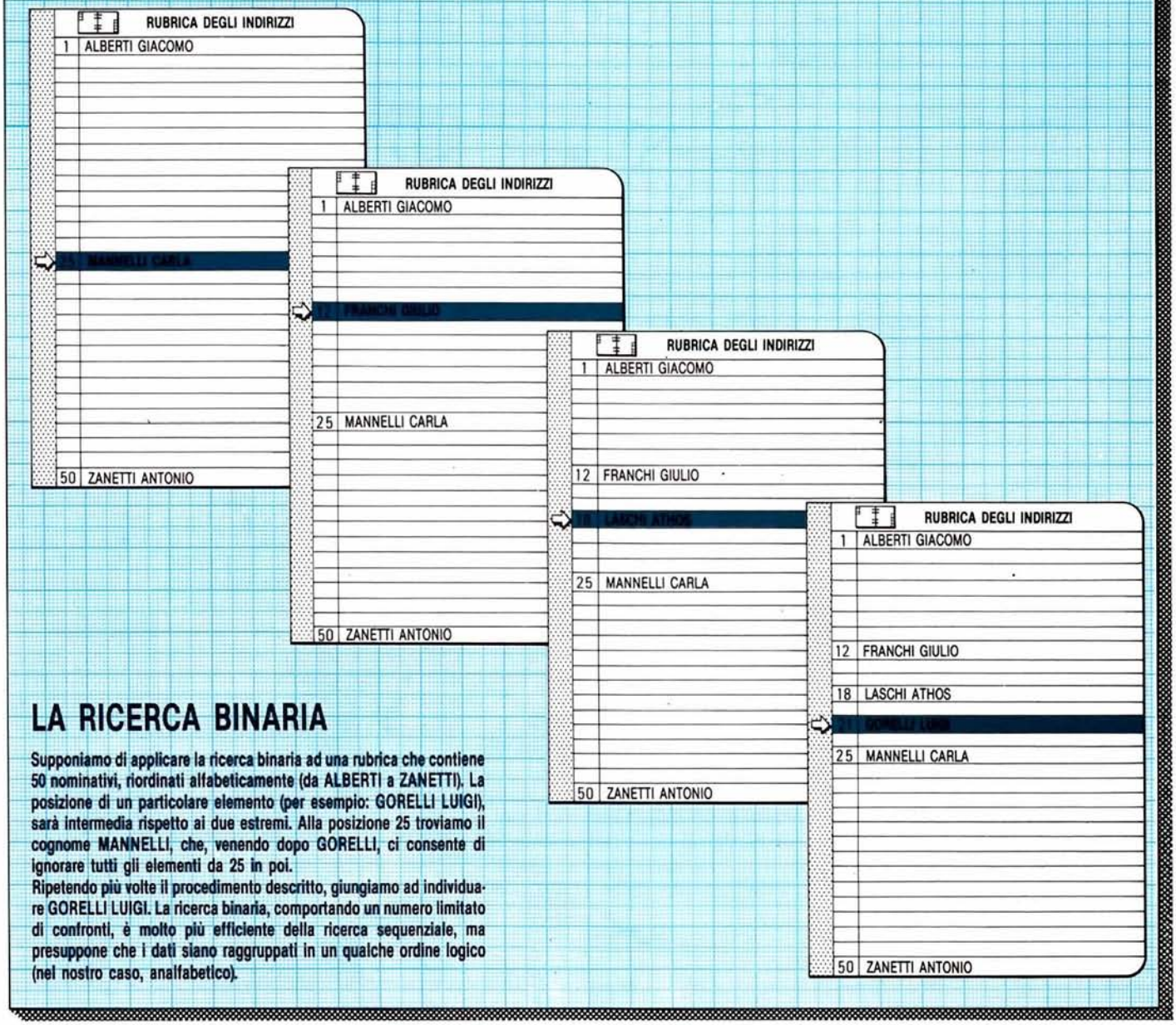
Chi cerca...

La ricerca binaria velocizza la ricerca dei dati, purché questi siano già ordinati

Le tre funzioni principali della nostra rubrica computerizzata (inserimento, memorizzazione su disco o nastro e lettura del file al primo lancio del programma) sono già state sviluppate.
Occorre adesso dedicarsi a quella parte di programma che consente la ricerca e la visualizzazione dei record memorizzati.
Forse, l'attività che verrà praticata più frequentemente sarà la ricerca di un record usando il nome di una persona: ecco perché questa è la prima opzione presentata dal menu. La ricerca è una delle funzioni più importanti nella maggior parte dei programmi, ma in particolare nei cosiddetti database, nei quali le informazioni devono essere estratte da archivi probabilmente molto vasti.
Il metodo della ricerca "lineare" (in cui il file viene esaminato sequenzialmente iniziando dal primo elemento in poi) è senza dubbio lentissimo, pur essendo l'unico applicabile qualora i dati non siano disposti secondo un qualche ordine logico. Il tempo medio di ricerca, in una matrice di N elementi, è direttamente proporzionale ad N/2. Ne consegue che se N è piccolo, il tempo medio di ricerca può essere ancora accettabile, ma se N è molto grande, la ricerca richiede tempi assolutamente troppo lunghi.
Se però i dati sono disposti secondo un certo ordine logico (per esempio, quello alfabetico), possiamo applicare un metodo di ricerca molto più efficiente: quello "binario". Per illustrarne il funzionamento, supponiamo di voler trovare la spiegazione del termine "leptocefalo" in un dizionario. Mai sogneremmo d'iniziare la ricerca dalla prima pagina e, non avendo trovato la parola, passare alla seconda pagina, e via dicendo. Al contrario, apriamo il dizionario verso la metà e controlliamo quale parola vi troviamo. Se questa dovesse essere, ad esempio, "metatarso", sappiamo di dover continuare la ricerca nella prima parte del dizionario e ciò che è contenuto nella seconda metà non ci interessa affatto. Ripetiamo il processo, indirizzandoci verso una pagina intermedia tra l'inizio del dizionario e la pagina scelta precedentemente.
Questa volta incontriamo il termine "dolabra", che, ai fini della nostra ricerca, ha un valore troppo "basso". A questo punto, la scelta di una nuova pagina è limitata tra quella che comprende "dolabra" e quella che contiene "metatarso". Il procedimento viene ripetuto, finché non troviamo la pagina che accoglie il termine cercato.
Nell'esempio appena esposto, il termine "leptocefalo" costituisce la "chiave di ricerca". Ogni volta che esaminiamo un record di un file, alla ricerca di un particolare dato, confrontiamo la "chiave di ricerca" con la corrispondente chiave del record. Alla chiave del record sarà associata una serie di informazioni aggiuntive. Nel caso del nostro esempio, le "informazioni aggiuntive" associate a leptocefalo saranno la spiegazione del termine (ossia: minuscolo pesce marino, ecc.).
L'analogia con la ricerca di un database è molto appropriata, purché i record del file risultino ordinati logicamente, come le parole del dizionario.
La parte di programma relativo alla ricerca, nella nostra rubrica computerizzata, è leggermente più complicata, come vedremo tra breve. La prima azione della routine (chiamiamola per il momento CERCREC) sarà quella di chiedere il nome da cercare, che costituirà la nostra "chiave di ricerca". Supponiamo che nel file esista il nominativo Luigi Gorelli. Il relativo record conterrà un campo composto da cognome e nome (nel formato standard): GORELLI LUIGI. La routine visualizzerà una richiesta del tipo: "NOME DA CERCARE?", alla quale potremmo rispondere con Luigi Gorelli, oppure L. Gorelli, o, ancora, Gigi Gorelli. Per non complicare le cose, supponiamo di rispondere con Luigi Gorelli. La routine provvederà immediatamente a convertire il nominativo nel formato standard: GORELLI LUIGI. Successivamente, confronterà questa chiave di ricerca con il campo CAMMOD$ dei vari record. Utilizzando una ricerca lineare, i confronti inizierebbero dal primo record, per terminare soltanto a ritrovamento avvenuto (o alla fine del file se il nominativo non è mai stato inserito). Ma, poiché abbiamo previsto una routine per il riordino alfabetico dei nominativi, possiamo applicare una ricerca binaria. Basta controllare (verificando il valore della variabile RMOD) se gli elementi sono ordinati o no. Se lo sono, il programma sa che il primo elemento da confrontare è CAMMOD$(1), mentre l'ultimo è CAMMOD$(DIMENSIONE −1). Per la ricerca binaria, ci occorrono tre variabili: FDO,

per indicare il limite inferiore di ricerca (CAMMOD$(1), inizialmente) CMA, per indicare il limite superiore (CAMMOD$ (DIMENSIONE—1), inizialmente) e MED, per indicare un valore corrispondente ad un elemento intermedio tra i precedenti.
Possiamo quindi porre, inizialmente, FDO = 1 e CMA = DIMENSIONE —1 (si ricordi che DIMENSIONE contiene il numero totale dei record inseriti, aumentato di uno). Se, per esempio, abbiamo 21 elementi nella rubrica, DIMENSIONE vale 22, FDO vale 1 e CMA vale 21. Il valore di MED viene ricavato, in BASIC, con l'espressione: MED = INT((FDO + CMA)/2). Pertanto, con i suddetti valori, MED vale 11. Per eseguire la ricerca binaria, estendiamo la ricerca, inizialmente, sull'intera matrice e determiniamo il valore di MED all'interno di un ciclo, che termini qualora venga trovato l'elemento cercato o la ricerca dia esito negativo. Calcolato MED, possiamo verificare se l'elemento corrispondente è identico alla chiave di ricerca, nel qual caso la ricerca è conclusa. Altrimenti, se il contenuto di CAMMOD$(MED) è troppo "basso", sappiamo che CAMMOD$(MED) può esser considerato il nuovo limite inferiore della parte di matrice nella quale continuare la ricerca. Possiamo dunque porre FDO = MED o, meglio ancora, FDO = MED + 1, poiché sappiamo già che l'elemento corrispondente a MED non è quello cercato. Analogamente, se CAMMOD$(MED) è "maggiore" rispetto alla chiave, a CMA si può assegnare il valore di MED — 1.
Quale passo intermedio, nello sviluppo della routine definitiva, possiamo redigerne una versione "di prova", nella quale, immessa

La ricerca di un nominativo



LA RICERCA BINARIA

Supponiamo di applicare la ricerca binaria ad una rubrica che contiene 50 nominativi, riordinati alfabeticamente (da ALBERTI a ZANETTI). La posizione di un particolare elemento (per esempio: GORELLI LUIGI), sarà intermedia rispetto ai due estremi. Alla posizione 25 troviamo il cognome MANNELLI, che, venendo dopo GORELLI, ci consente di ignorare tutti gli elementi da 25 in poi.
Ripetendo più volte il procedimento descritto, giungiamo ad individuare GORELLI LUIGI. La ricerca binaria, comportando un numero limitato di confronti, è molto più efficiente della ricerca sequenziale, ma presuppone che i dati siano raggruppati in un qualche ordine logico (nel nostro caso, analfabetico).

KEVIN JONES

una chiave di ricerca, venga visualizzato il messaggio "RECORD NON TROVATO", oppure "IL RECORD È IL N. XXX", dove XXX rappresenta il valore di MED. Iniziando la numerazione della routine da 13000, possiamo "accodarla" al programma principale presentato a pag. 399. Occorrerà, in questo caso, modificare la linea 4040 in: IF SCEL = 1 THEN GOSUB 13000. La linea 13240 contiene l'istruzione STOP, affinché il programma sospenda l'esecuzione non appena visualizzato uno dei due messaggi appena descritti.
È possibile riavviare il programma, senza perdere alcun dato, semplicemente impartendo da tastiera il comando CONT. Omettendo la STOP, il programma eseguirebbe immediatamente la successiva RETURN, non lasciando sufficiente tempo per leggere il messaggio visualizzato.
Nella linea 13100 alla variabile FDO viene assegnato il valore 1 (pari alla posizione dell'elemento "più basso" nella matrice CAMMOD$). Il valore di CMA, invece, è assegnato nella linea 13110 ed equivale a DIMENSIONE – 1 (la posizione dell'ultimo elemento nella matrice CAMMOD$). Alla linea 13120 ha inizio un ciclo, che termina soltanto quando viene trovato l'elemento cercato oppure la ricerca ha dato esito negativo.
Nella linea 13130, calcolando la media aritmetica tra i valori di FDO e CMA, si trova la posizione dell'elemento che sta al centro fra questi due estremi. Si noti l'impiego della funzione INT, che arrotonda il risultato, evitando di ottenere valori frazionari (ad esempio: 1,5). Può darsi che il contenuto di CAMMOD$(MED) corrisponda proprio alla chiave di ricerca, nel qual caso l'obiettivo è stato raggiunto.
Ma se così non fosse, la variabile L viene posta a 0, garantendoci che il ciclo non abbia termine. Essendo fallito il confronto (linea 13140), i casi sono due: o il valore di CAMMOD$(MED) è inferiore a quello di KEY$ (la chiave di ricerca), oppure è superiore. Nel primo caso, alla variabile FDO viene assegnato un valore pari a MED + 1 (linea 13150), nel secondo caso, invece, è la variabile CMA che assume un valore pari a MED – 1. La ragione per la quale non viene utilizzato direttamente il valore di MED è ovvio: sarebbe superfluo ripetere nuovamente un confronto sul contenuto di CAMMOD$(MED).
Se tutti i confronti falliscono (l'elemento cercato non esiste), ad un certo punto il valore di FDO sarà superiore a quello di CMA, ed il ciclo termina (13170), causando la visualizzazione del messaggio RECORD NON TROVATO (linea 13200).
Il frammento di programma è presentato a scopo dimostrativo, col solo fine di poter verificare il funzionamento della routine di ricerca.
Nella versione definitiva, dovremo prevedere la visualizzazione dell'intero record, qualora la ricerca abbia buon esito. Ciò non costituisce una grande difficoltà: conoscendo il numero del record, possiamo facilmente visualizzare le informazioni in esso contenute (CAMNOM$, CAMVIA$, ecc.), facendo seguire un messaggio del tipo "PREMERE IL TASTO SPAZIO PER CONTINUARE" o "DIGITARE 'P' PER STAMPARE IL RECORD".
Meno facile, invece, è la parte che concerne l'input che serve a FNDREC. Nel frammento di programma presentato, diamo per scontato che il formato della chiave di ricerca corrisponda a quello standard, ad esempio: GORELLI LUIGI. Imporre che la chiave di ricerca abbia questo formato è fortemente limitativo ed anche poco pratico. Inoltre, basta una minima imprecisione della chiave per far fallire la ricerca. Potremmo pensare di risolvere parte dei problemi ricorrendo alla subroutine MODNOM, sebbene la possibilità di effettuare una ricerca "approssimata", utilissima in una simile applicazione, debba essere, comunque, trattata separatamente. Se, però, esaminiamo attentamente il funzionamento di MODNOM, ci accorgeremo immediatamente che la subroutine manca di generalità (un errore abbastanza frequente nella programmazione): è stata scritta, infatti, principalmente in funzione di un'altra sezione di programma, AGGREC. In primo luogo si dà per scontato che il nominativo da convertire in formato standard sia contenuto in CAMNOM$(DIMENSIONE); in secondo luogo che il risultato della conversione vada conservato in CAMMOD$(DIMENSIONE). Abbiamo dunque tre possibilità: riscrivere totalmente il modulo MODNOM (il che comporterebbe apportare modifiche anche ad altre parti del programma), scrivere un nuovo modulo, pressoché identico a MODNOM (con evidente spreco di memoria e di tempo) oppure abbandonarsi a qualche biasimevole forma di cattiva programmazione e utilizzare alla meno peggio il modulo esistente. Quest'ultima soluzione è la peggiore, perché il programma risultante sarebbe talmente caotico, che troveremmo enormi difficoltà, in un secondo momento, a comprenderne noi stessi il funzionamento.
Morale della favola: occorre lavorare "sulla carta", prima di precipitarsi a scrivere le linee di programma. Solo così si otterranno buoni programmi.
Tanto per illustrare un cattivo esempio di programmazione, consideriamo la linea 13020, nella quale viene immessa la chiave di ricerca, e vediamo come modificare MODNOM, affinché serva ai nostri scopi:

```
13020 INPUT "CHIAVE DI RICERCA"; CAMNOM$(DIMENSIONE)
13030 GOSUB 10200 : REM SUBROUTINE MODNOM
13040 LET KEY$ = CAMMOD$(DIMENSIONE)
13050 .....
```

Fortunatamente, il valore di DIMENSIONE è sempre superiore al numero dell'ultimo record valido. In altre parole, non esiste alcun elemento nella posizione DIMENSIONE e ricorrere a questa soluzione "di ripiego" non causa alcun

danno né alla struttura, né al contenuto delle informazioni. Senza alcune linee REM, nelle quali spiegare a fondo la modifica apportata, ci troveremmo in seguito di fronte a grosse difficoltà.
Ma torniamo al problema della ricerca "approssimata". Supponiamo di immettere, come nominativo per la rubrica, Gigi Gorelli. Successivamente, tentiamo di cercare il record usando Luigi Gorelli. Internamente, i due nominativi verrebbero automaticamente convertiti nei formati GORELLI GIGI e GORELLI LUIGI. Una ricerca darebbe quindi esito negativo, nonostante la presenza dell'elemento cercato.
Una traccia per risolvere questo problema, che i più volenterosi potranno sviluppare ed inserire nel programma definitivo, è la seguente:

```
INIZIO (ricerca di un nominativo)
   SE trovata esatta corrispondenza
      ALLORA visualizza l'intero record
      ALTRIMENTI esegui ricerca approssimata
         SE esiste qualche approssimazione
            ALLORA visualizza il record "approssimativo"
            ALTRIMENTI avvisa "RECORD NON TROVATO"
         FINE SE
   FINE SE
FINE
```

Il modulo di ricerca approssimata potrebbe avere questa struttura:

```
INIZIO (ricerca approssimata)
   Ricerca di identità nel cognome
   SE trovata identità nel cognome
      ALLORA cerca il nome più somigliante
      VISUALIZZA i record trovati
      ALTRIMENTI cerca somiglianza nel cognome
         SE somiglianza trovata
         ALLORA visualizza record trovati
         FINE SE
   FINE SE
FINE
```

La procedura per trovare il nome od il cognome "più somigliante" può basarsi, ad esempio, sul numero di caratteri comuni tra la stringa cercata e quelle presenti nella rubrica, oppure su di una identità parziale tra l'una e l'altra.
Il criterio è totalmente soggettivo ed esiste più di una soluzione.
Esiste, però, un effetto collaterale, nel frammento di programma qui presentato, da non trascurare. Supponiamo che si presenti la seguente particolare sequenza di eventi. Nella rubrica sono già stati inseriti dieci nominativi. Lanciamo il programma e, mediante il modulo AGGREC, aggiungiamo un nominativo. Subito dopo, cerchiamo di localizzare un record (mediante la subroutine FNDREC). Ebbene, quando usciamo definitivamente dal programma (subroutine EXPROG), il record da noi aggiunto non viene affatto memorizzato nella rubrica! Questa è l'immediata conseguenza di una certa operazione compiuta nella subroutine FNDREC. Perché accada un simile inconveniente e come evitare simili perdite d'informazioni verrà spiegato nella prossima lezione.
Ed ecco un'ultima esercitazione:

cosa accadrebbe se nella rubrica vi fossero esattamente 50 nominativi e venisse usata la subroutine FNDREC (che a sua volta chiama MODNOM)?
Il valore di DIMENSIONE sarebbe 51 e allora?

Basic & Basic

SPECTRUM

Per lo Spectrum sono necessarie le seguenti modifiche:

```
13000 REM VERSIONE PROVA DI FNDREC
13010 IF RMOD = 1 THEN GOSUB 11200
13020 INPUT "CHIAVE DI RICERCA";S$
13100 LET FDO = 1
13110 LET CMA = DIMENSIONE — 1
13120 FOR L = 1 TO 1
13130 LET MED = INT((FDO + CMA)/2)
13140 IF M$(MED) <> S$ THEN LET L = 0
13150 IF M$(MED) < S$ THEN LET FDO =
      MED + 1
13160 IF M$(MED) > S$ THEN LET CMA =
      MED — 1
13170 IF FDO > CMA THEN LET L = 1
13180 NEXT L
:
13200 IF FDO > CMA THEN PRINT "RECORD
      NON TROVATO"
13210 IF FDO < = CMA THEN PRINT "IL
      RECORD È IL N.";MED
:
13240 STOP
13250 RETURN
```

Si noti, ancora una volta, che nello Spectrum i nomi delle variabili alfanumeriche sono limitati ad una sola lettera

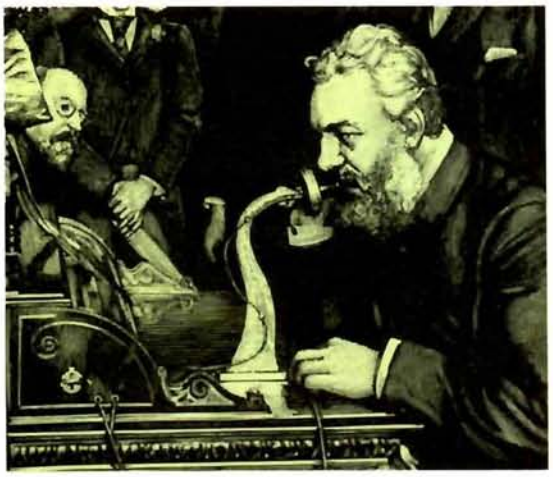
```
13000 REM VERSIONE DI PROVA DI == FNDREC ==
13010 IF RMOD = 1 THEN GOSUB 11200
13020 INPUT "CHIAVE DI RICERCA ";KEY$
13030 REM
13040 REM
13050 REM
13060 REM
13070 REM
13080 REM
13090 REM
13100 LET FDO = 1
13110 LET CMA = DIMENSIONE - 1
13120 FOR L = 1 TO 1
13130 LET MED = INT((FDO + CMA)/2)
13140 IF CAMMOD$(MED) <> KEY$ THEN L = 0
13150 IF CAMMOD$(MED) < KEY$ THEN FDO = MED + 1
13160 IF CAMMOD$(MED) > KEY$ THEN CMA = MED - 1
13170 IF FDO > CMA THEN L = 1
13180 NEXT L
13190 REM
13200 IF FDO > CMA THEN PRINT "RECORD NON TROVATO"
13210 IF FDO < = CMA THEN PRINT "IL RECORD E' IL N.";MED
13220 REM
13230 REM
13240 STOP
13250 RETURN
```

La Bell Laboratories

Un istituto che ha fortemente contribuito allo sviluppo dei computer

Cent'anni fa la regina Vittoria fu molto divertita da una nuova invenzione, il telefono, che le permetteva di parlare con i suoi ministri, a Londra, mentre si trovava nell'Isola di Wight. Da allora, il telefono ha compiuto enormi progressi e molta della tecnologia adottata in questo campo è servita nella fabbricazione dei computer. Quando le ricerche, in questo settore, erano ancora agli inizi, l'American Telephone and Telegraph Company decise di creare un'organizzazione che sperimentasse nuovi metodi per migliorare i sistemi telefonici allora esistenti. Nel 1925 furono inaugurati a Murray Hill, nel New Jersey, i "Laboratori Bell", meglio noti, oltre oceano, come "Ma Bell".

Pur facendo parte di un'industria, i cui fini sono il profitto, questi laboratori sono un'istituzione piuttosto inconsueta, poiché si occupano esclusivamente di ricerca, senza scopi immediati di commercializzazione. Gli scienziati che vi lavorano non sono quindi oppressi dal problema di produrre invenzioni da immettere rapidamente sul mercato (situazione tipica, invece, nei comuni laboratori di ricerca delle industrie). La Bell, infatti, considera la ricerca un investimento a lungo termine. Gli scienziati hanno modo di approfondire con calma gli aspetti della ricerca che giudicano più stimolanti: la filosofia industriale della Bell si basa sulla teoria che la libera ricerca, prima o poi, si dimostra altamente remunerativa. Questi laboratori si sono già guadagnati due premi Nobel ed hanno fatto scoperte in diversi campi della ricerca scientifica. Esamineremo adesso alcuni aspetti particolarmente rilevanti per lo sviluppo dei computer.



I primi telefoni di Bell

I laboratori Bell sono così chiamati dal nome di Alexander Graham Bell (1847-1922), che viene universalmente considerato uno degli inventori del telefono (1876). Pare che le prime parole trasmesse su cavi elettrici siano state rivolte da Bell ad un suo assistente che si trovava in un'altra stanza: "Venga qui, signor Watson, ho bisogno di lei!"

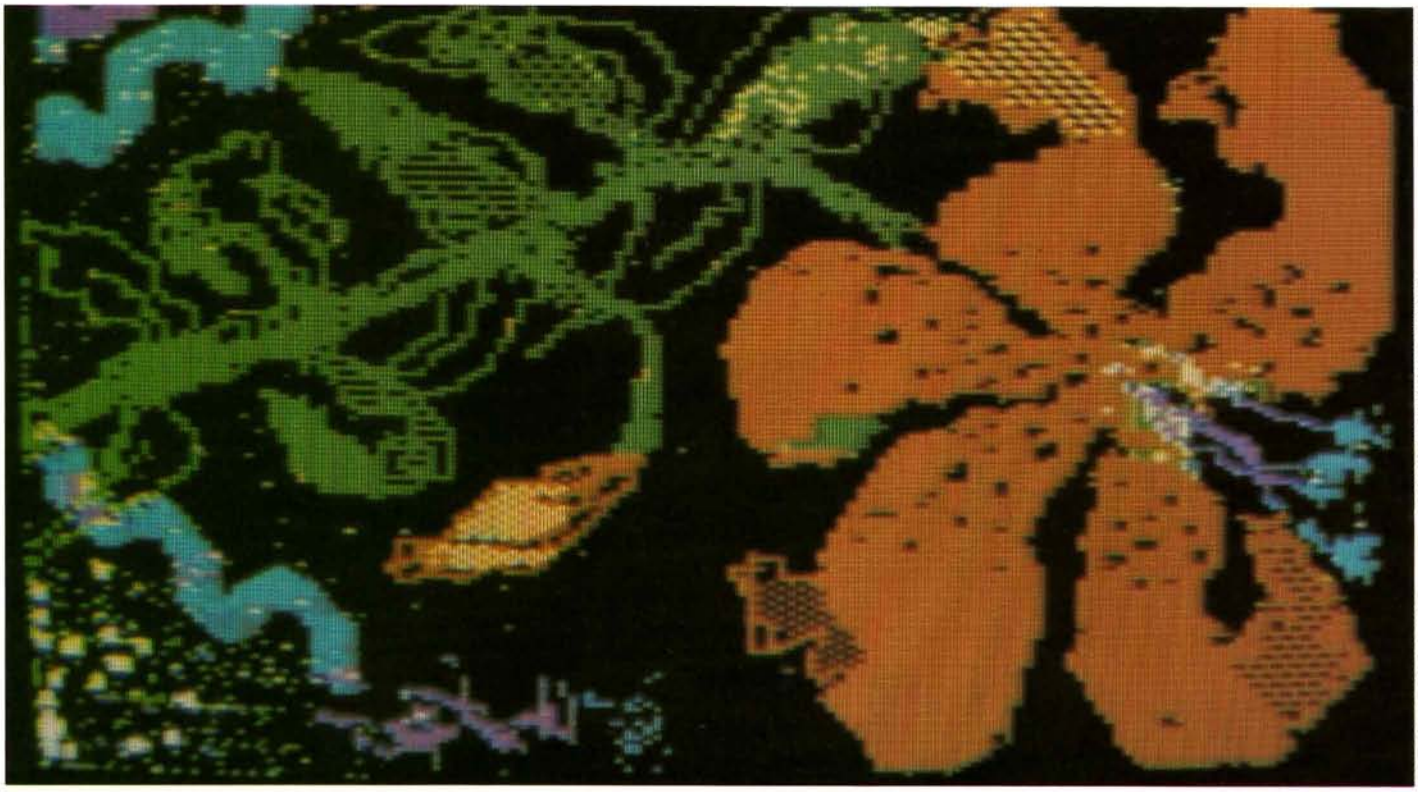
Nel 1930 i sistemi telefonici stavano diventando sempre più sofisticati ed automatici. I messaggi venivano inviati sulle linee telefoniche in forma analogica, ma i collegamenti erano realizzati usando informazioni espresse in un codice di chiamata digitale. Il numero selezionato, inizialmente in forma analogica, veniva convertito in una sequenza di impulsi digitali. Questi venivano temporaneamente immagazzinati in una memoria costituita da interruttori a relè, finché il collegamento non veniva completato da una batteria di commutatori a barre incrociate. Questi contavano gli impulsi nel codice di chiamata e li convertivano in coordinate su una centralina elettromeccanica. Erano quindi già presenti tutti gli ingredienti di un computer: mancava solo un inventore adatto! George Stibitz era un matematico impiegato alla Bell che notò la somiglianza tra il procedimento di "conteggio" e quello di "somma" degli impulsi. Lavorando nella cucina della propria casa, tra vecchi commutatori a barre incrociate e relè elettromeccanici, egli fabbricò i primi circuiti di computer a relè.

In seguito Stibitz lavorò assieme ad un ingegnere esperto in interruttori, Samuel B. Williams, che per 25 anni aveva fabbricato circuiti di commutazione. I due crearono un Calcolatore per Numeri Complessi (i numeri complessi implicano il concetto dei numeri cosiddetti "immaginari", ossia derivati dalla radice quadrata di numeri negativi, che sono necessari alla soluzione delle equazioni polinominali). Il loro lavoro ebbe inizio nel 1937 e l'apparecchio prodotto utilizzava 450 relè e 10 commutatori a barre incrociate, operava in notazione binaria ed era capace di dividere due numeri di 8 cifre in 30 secondi. Il Calcolatore per Numeri Complessi divenne operativo l'8 gennaio 1940 e nel settembre dello stesso anno venne tenuta una dimostrazione per la Società Matematica Americana al Dartmouth College (dove in seguito venne sviluppato il linguaggio BASIC). Il calcolatore era caratterizzato dalla possibilità di collegare più terminali, anche distanti dall'apparecchio, costituiti da tastiere tipo macchina da scrivere e collegati per mezzo di cavi telefonici al meccanismo di calcolo che si trovava a New York. Il grande pubblico rimase particolarmente colpito dal modo di procedere della macchina, che per certi aspetti somigliava a quello umano. Dopo l'impostazione di un quesito, il calcolatore sembrava fermarsi a riflettere per qualche secondo prima di dare la risposta.

Alla Bell sono state inventate anche altre componenti hardware per computer: ad esempio, le testine a cuscinetto d'aria per la lettura dei nastri megnetici e gli amplificatori a reazione negativa. Ma l'invenzione più famosa è stata senza dubbio il transistor, progettato nel 1947 da Bardeen, Brattain e Shockley (vedere a pag. 47). È stato proprio il transistor a rendere possibile lo sviluppo della seconda generazione di computer.

Il Computer Aided Design

La realizzazione di immagini grafiche al computer richiede complessi calcoli e molta memoria



DIGITISER BY CONNY JUDE SCREEN SHOT BY IAN McKINNELL per concessione della SOFT

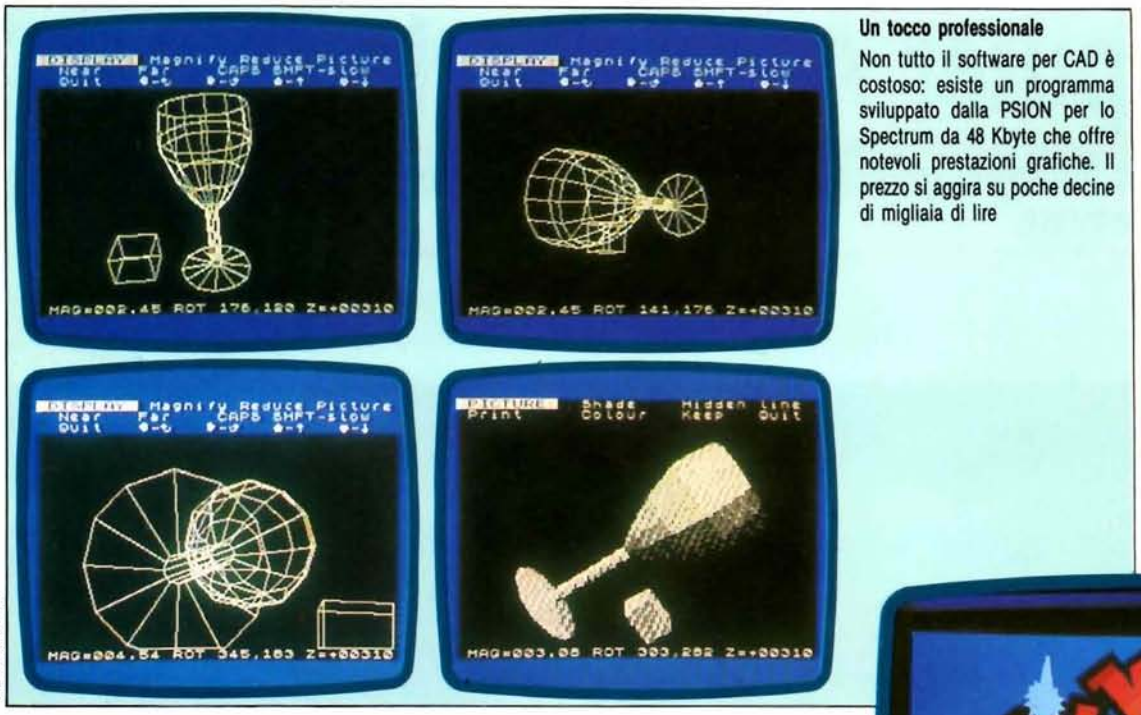
L'idea di applicare i computer nella produzione industriale delle immagini è nata agli inizi degli anni '60, probabilmente al Massachusetts Institute of Technology. L'effettiva disponibilità, però, di apparecchiature tali da permettere il disegno computerizzato vero e proprio si è avuta solo dieci anni più tardi. I "ferri del mestiere" sono penne ottiche, digitiser, plotter e monitor ad altissima risoluzione. Grazie a questi strumenti, il disegnatore può creare immagini di grandissimo effetto, come abbiamo già accennato a pagina 181, parlando della produzione d'immagini nel cinema d'animazione. In sostanza, si tratta di applicare, alle immagini, concetti di elaborazione simili a quelli usati per le parole nel word processing.

Abbiamo già fatto notare precedentemente che la qualità di un'immagine dipende da due fattori principali: la risoluzione grafica dell'apparecchio di visualizzazione (il numero di pixel rappresentabili sullo schermo del monitor) e le dimensioni della memoria del computer. Esistono monitor dotati di una risoluzione di 1000×1200 pixel, pilotati da computer capaci di elaborare oltre un milione di elementi d'immagine in meno di un 24mo di secondo. L'analogia tra il disegno assistito da computer (più brevemente CAD, da Computer Aided Design) e l'elaborazione dei testi è piuttosto appropriata: invece di spostare parole, frasi o interi paragrafi all'interno di un testo, apportandovi tutte le modifiche necessarie, qui si tratta di elaborare informazioni visive. Cambia il contenuto, ma i principi sono identici.

Il principale problema di programmazione concerne il miglior metodo da adottare per conservare le immagini in maniera da poterle modificare e manipolare efficacemente. Lavorando su un modello fisico di un oggetto, esistono due metodi d'intervento: quello aggiuntivo e quello sottrattivo. Il criterio aggiuntivo è quello impiegato, ad esempio, nella creazione di forme con l'argilla: via via si aggiungono nuovi pezzi di materia, fino a raggiungere la forma desiderata. Il criterio sottrattivo, invece, è paragonabile al lavoro di uno scultore che lavori un pezzo di marmo: in questo caso l'artista, per ottenere il risultato voluto, rimuove parti di materia sempre più piccole. Nel computer, il blocco di marmo

La "mela" in fiore

Per quanto sofisticato sia il software, l'elemento di maggiore importanza, in un sistema CAD, rimane sempre il disegnatore umano. L'illuminazione mostra un disegno prodotto col Versawriter, un programma che gira sull'Apple II, purché dotato di due unità a dischi ed un digitiser. La creazione di un'immagine può richiedere molto tempo ed è paragonabile all'opera di un artista



Un tocco professionale

Non tutto il software per CAD è costoso: esiste un programma sviluppato dalla PSION per lo Spectrum da 48 Kbyte che offre notevoli prestazioni grafiche. Il prezzo si aggira su poche decine di migliaia di lire

IAN McKINNELL

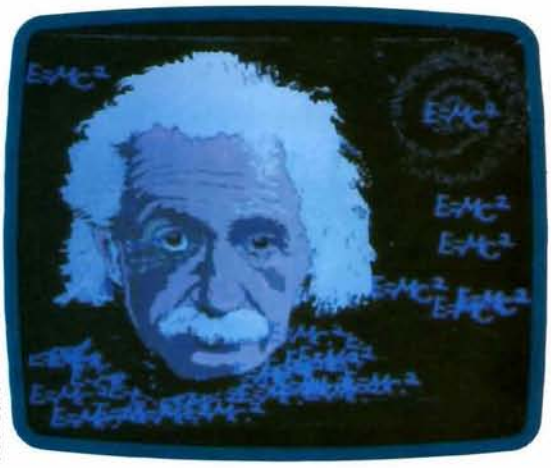


potrebbe corrispondere ad una matrice tridimensionale di elementi numerici. Di conseguenza, la memoria e la velocità di calcolo del computer divengono i fattori predominanti. Se la matrice è sufficientemente ampia da consentire la memorizzazione di un intero byte per ogni pixel da visualizzare, allora la quantità di informazioni, relative a ciascun elemento, offre possibilità d'intervento pressoché illimitate. Ma, purtroppo, la gestione di matrici di notevoli dimensioni crea non pochi problemi, e si è costretti, in genere, a ricorrere ad un compromesso: invece di riservare un intero byte per ogni elemento dell'immagine, ci si limita ad impegnare un solo bit, che possa indicarci la presenza o l'assenza di un particolare punto luminoso nell'immagine da rappresentare.

I programmi per il CAD hanno molti processi in comune con quelli per la generazione di immagini computerizzate (il cosiddetto Computer Generated Image Software): l'arrotondamento automatico delle curve, la gestione delle linee nascoste, l'ombreggiatura, la colorazione di blocchi e via dicendo. Spesso, non si tratta che di trovare ripetutamente le soluzioni di un'equazione per una serie di valori prestabiliti. Per esempio, possiamo specificare la posizione di tre punti della curva (quello di partenza, quello di arrivo ed uno intermedio) ed il computer ci fornisce una delle possibili soluzioni, tracciando una curva che si adatti ai tre punti. Inoltre, è anche possibile seguire il percorso inverso: tracciare una linea e lasciare al computer il compito di trovare l'equazione che serve a rappresentare, matematicamente, la curva.

Viva la grafica!

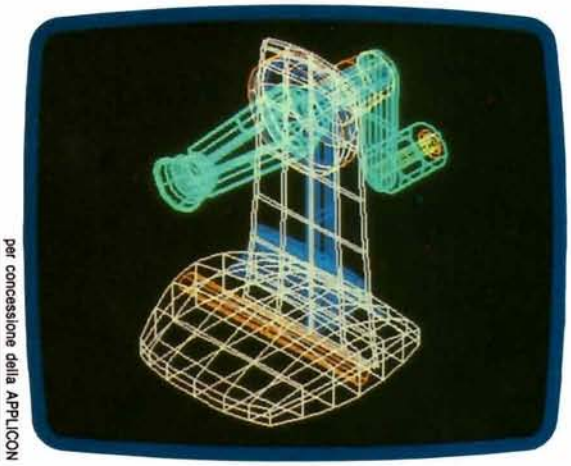
Il sistema Pluto, della Io Research, aggiunge alla maggior parte dei microcomputer una gestione grafica veramente d'effetto. Il sistema di base, che si può successivamente ampliare, dispone di otto colori e una risoluzione di 670 x 576 pixel



per concessione della RESEARCH



La vera forza di un sistema CAD, risiede proprio nella facoltà di utilizzare forme preesistenti (conservate nella memoria del computer) per comporre immagini e strutture totalmente nuove, evitando di dover ridisegnare ogni singola componente per ogni nuovo lavoro.
Una tipica applicazione di questo genere è l'impiego di computer nel progetto delle future generazioni di elaboratori.
Infatti, il progetto dei circuiti stampati di un apparecchio elettronico è un lavoro estremamente complesso, al quale occorre sempre applicare criteri di ottimizzazione per scegliere la migliore disposizione dei componenti e dei collegamenti fra di essi.
Manualmente, è necessario procedere molto empiricamente, mentre, grazie al CAD, il lavoro diventa più rapido e preciso. I vari componenti sono memorizzati sotto forma di immagini predefinite, che vengono richiamate con semplici comandi: risulta facile, pertanto, produrre sul monitor degli schemi di circuiti (anche molto complessi) e verificarne la correttezza, prima di impostare la produzione dei relativi circuiti stampati. In certi casi è addirittura possibile simulare il comportamento elettronico dei vari componenti, per verificarne il funzionamento teorico.
Il progetto di circuiti integrati, ed ancor più quello dei famigerati chip, ove la densità dei componenti è altissima, impone un'ulteriore necessità: quella di poter ingrandire a volontà alcuni dettagli dell'immagine completa, consentendo di apportare modifiche sull'ingrandimento del disegno e di ritornare alla scala originale.



per concessione della APPLICON

Questa facoltà d'ingrandimento e riduzione è una delle caratteristiche più importanti di un sistema CAD, senza la quale non saremmo, oggi, in grado di progettare le apparecchiature più sofisticate.
Oltre alla capacità di variare la scala del disegno, esiste un'altra interessantissima possibilità: quella di abbinare ad un disegno di base più immagini indipendenti. Per esempio, nel progetto di un'automobile si possono separare il circuito elettrico, quello dei freni, il gruppo di trasmissione, e via dicendo, dalla rappresentazione del telaio o della carrozzeria. Ogni particolare aspetto, quindi, viene sviluppato separatamente dai tecnici competenti, pur avendo la possibilità di riunire il complesso delle immagini in un'unica rappresentazione tridimensionale.



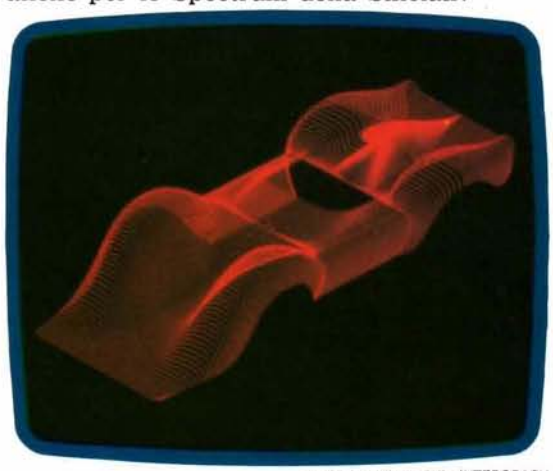
per concessione della SIGGRAPH

Immaginiamo che...

Lo sfondo di quest'immagine, tratta dal film "Road to Point Reyes", prodotto dalla Lucasfilms, è stata ottenuta mediante l'impiego di una nuova tecnica CAD, detta "frastagliamento". Le colline ed i monti, visibili sullo sfondo, sono stati costruiti partendo da semplici forme geometriche, applicando un processo di "frastagliamento" casuale. Il principio di questa tecnica è più facilmente comprensibile osservando come viene costruita l'immagine di un fiocco di neve (vedi illustrazione sottostante)



La grande innovazione consiste, oltre che nello sviluppo di progetti secondo le modalità appena descritte, anche nella capacità di conservare, assieme alle immagini ed ai diagrammi, anche specifiche informazioni relative a ciascun componente del prodotto finale. Si tratta, quindi, di qualcosa di molto più esteso e complesso di un semplice archivio d'immagini: si può addirittura interrogare il sistema, affinché ci fornisca informazioni sul genere e la quantità di componenti da ordinare per uno specifico progetto, oppure un piano di lavoro dettagliato sulla sequenza temporale degli interventi necessari per portare a termine la costruzione di un particolare prodotto.
Allo stato attuale, non è troppo azzardato prevedere che ben presto verranno utilizzati dei sistemi di gigantesche dimensioni in grado di controllare totalmente, anche grazie all'impiego di sofisticati robot, l'intera produzione manifatturiera, lasciando all'uomo il solo compito di sbizzarrire la propria creatività.
La maggior parte delle applicazioni qui descritte riguarda elaboratori di notevoli dimensioni e capacità, ma esiste una discreta gamma di programmi CAD, ovviamente in versione ridotta, per microcomputer dotati del sistema operativo CP/M e, addirittura, anche per lo Spectrum della Sinclair.



per concessione della INTERGRAPH

Una fitta rete

La prima fase, nella creazione di un'immagine, consiste nel creare un reticolo tridimensionale, partendo dalle coordinate di alcuni punti che compongono l'oggetto da rappresentare. Successivamente, si può chiedere al computer di smussare la curvatura delle linee, di rimuovere le "linee nascoste" e di colorare le varie superfici, aggiungendo, eventualmente, anche un'opportuna ombreggiatura

La macchina di Turing

Un apparecchio che appartiene alla teoria dell'informatica

Fino ad ora, la predominanza degli argomenti presentati nel corso de "Il mio computer" hanno avuto carattere essenzialmente pratico. Questa volta, invece, affrontiamo una materia che appartiene alla teoria dell'informatica. Si sa che, spesso, è dalla teoria che scaturiscono le idee per applicazioni d'ordine pratico: è questo il caso della macchina di Turing.

Sviluppata da Alan Turing (vedere a pag. 200), questa macchina puramente teorica è nata come sussidio allo studio degli algoritmi e della possibilità di risolvere alcuni particolari problemi di elaborazione. In altre parole, si tratta di un "banco di prova" teorico per verificare se un problema è risolvibile mediante computer. Turing decise che questo apparato doveva possedere tre caratteristiche principali: un sistema di memorizzazione esterno dei dati immessi e di quelli elaborati, un dispositivo di lettura e di scrittura di tali dati ed un meccanismo di controllo per stabilire le azioni da intraprendere.

L'unità esterna di memorizzazione della macchina di Turing è definita come un "nastro" (paragonabile ad un nastro magnetico, se questo facilita la comprensione) di lunghezza infinita (capace di immagazzinare tutti i dati relativi ad una particolare elaborazione, senza alcun limite di quantità). Il nastro viene suddiviso, per comodità, in una serie di quadrati, ognuno dei quali può essere vuoto oppure contenere un simbolo. Il dispositivo di lettura/scrittura si può immaginare come una testina di registrazione, capace di trasferire simboli da e nei quadrati del nastro, comandata da un'unità di controllo, che determina quali simboli trasferire ed in quale direzione muovere il nastro.

Infine, il meccanismo di controllo deve contenere il programma da eseguire. In questa ottica, la macchina di Turing si può considerare progettata "ad hoc" per una sola applicazione, in quanto non è previsto alcun sistema di lettura o di conservazione del programma. Come si è ripetuto più volte, la macchina di Turing è squisitamente teorica, sebbene ne esistano alcune realizzazioni a puro scopo didattico.

Il programma di controllo della macchina di Turing è costituito da frasi composte di 5 istruzioni, chiamate anche "quintette". Quale quintetta venga eseguita in un particolare momento dipende contemporaneamente da due fattori: dal simbolo presente nell'unità di lettura/scrittura e dallo "stato" o "condizione" della macchina. La qualità di questo stato o condizione è decisamente arbitraria: possiamo definire che la macchina inizia nella condizione S_A e che, raggiunta la speciale condizione H (HALT), essa si ferma, avendo terminato il calcolo. Tra questi due eventi, lo stato della macchina può cambiare numerose volte, in base alle istruzioni delle quintette. Lo stato della macchina serve, ovviamente, a riflettere, via via, i risultati dell'elaborazione in corso, ed a decidere quale quintetta eseguire di conseguenza.

I cinque elementi di una quintetta sono:

1) L'attuale stato della macchina
2) Il simbolo contenuto nel quadrato attualmente in corso di lettura
3) Il simbolo da scrivere nel medesimo quadrato del nastro (uguale a quello del punto 2 se non avviene alcuna modifica dei dati)
4) Lo stato della macchina a seguito delle precedenti operazioni (punti 2 e 3)
5) La direzione di scorrimento del nastro (avanti o indietro).

La quintetta (S_A; 5; 3; S_B; R), ad esempio, viene eseguita ogni qualvolta la macchina si trova nello stato S_A e la "testina" legge un 5. A questo punto, il 5 viene sostituito da un 3, la macchina assume lo stato S_B, ed il nastro viene fatto scorrere di una posizione a destra (R). Progettare una macchina di Turing per un particolare problema, significa definire il formato dei dati immessi mediante il sistema di lettura, quello dei dati prodotti dall'elaborazione (che verranno riscritti sul nastro quando, a lavoro concluso, la macchina giungerà allo stato H) ed il numero di quintette necessarie all'algoritmo.

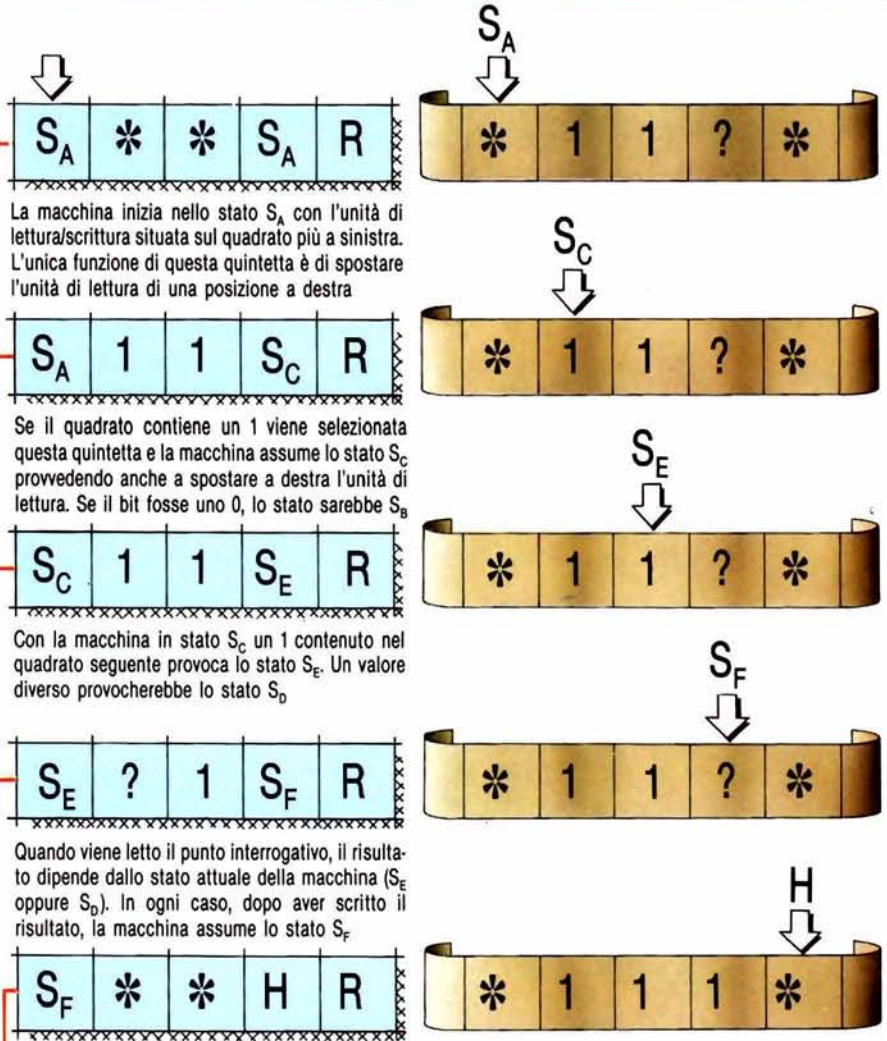
Nell'illustrazione a fianco, abbiamo rappresentato una macchina di Turing per l'applicazione della funzione AND. Collochiamo due bit (che rappresentano i dati in ingresso) in altrettanti quadrati, adiacenti, del nastro. Nel terzo quadrato, abbiamo posto un punto interrogativo che verrà, successivamente, sostituito dal risultato dell'elaborazione (un uno o uno zero). Per completezza, abbiamo posto due asterischi (uno precede i dati, l'altro li segue) per indicare quando la macchina debba iniziare l'elaborazione e quando il lavoro è stato portato a termine.

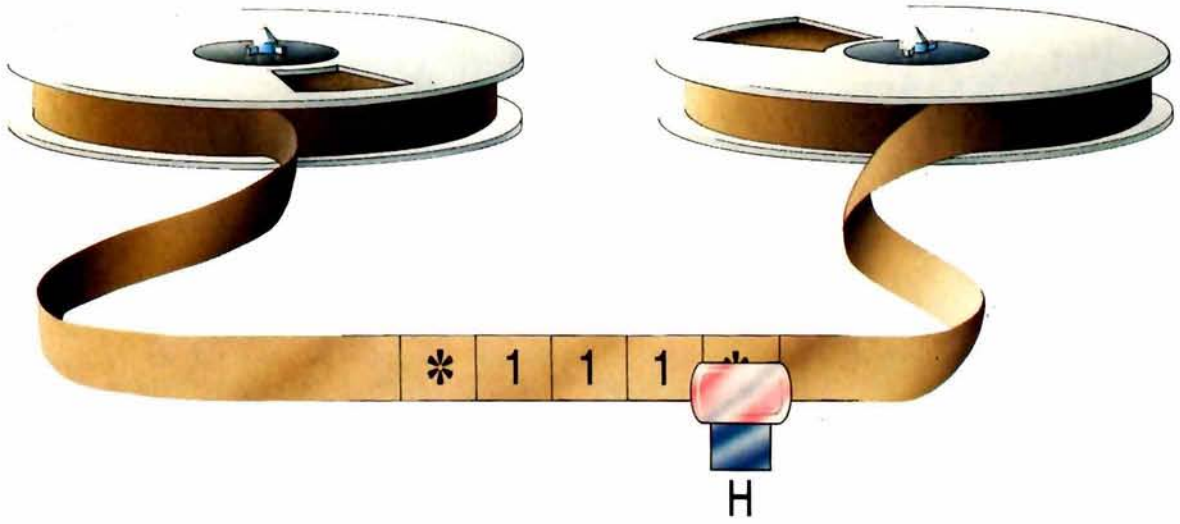
Per definire la funzione AND sono occorse dieci quintette, benchè, come si può notare dall'esempio riportato (1 AND 1 = 1), soltanto 5 di esse vengano effettivamente attivate nel corso di una sola elaborazione. Se proviamo con un altro esempio (0 AND 1) vedremo che le quintette coinvolte non sono le stesse.

LA MACCHINA DI TURING

L'esempio illustra la realizzazione di una macchina di Turing per operare un confronto logico AND. In effetti, essa simula teoricamente il funzionamento di una porta logica AND. I dati in ingresso, situati in due quadrati adiacenti del nastro, sono seguiti da un punto interrogativo, che verrà sostituito con il risultato dell'operazione. Due asterischi, posti ai due estremi dell'ipotetico nastro, servono per "avviare" e per "arrestare" la macchina. Occorrono 10 quintette per descrivere tutte le varie possibilità che si possono presentare, benché per ogni singola elaborazione non ne servano più di cinque

S_A	*	*	S_A	R
S_A	0	0	S_B	R
S_A	1	1	S_C	R
S_B	0	0	S_D	R
S_B	1	1	S_D	R
S_C	0	0	S_D	R
S_C	1	1	S_E	R
S_D	?	0	S_F	R
S_E	?	1	S_F	R
S_F	*	*	H	R





KEVIN JONES

Idee sonore

Continua la spiegazione del sofisticato comando ENVELOPE per il Micro BBC

Proseguiamo l'esame del comando ENVELOPE, usato congiuntamente a SOUND (vedere a pag. 358), con la spiegazione dell'"inviluppo di volume". Nella seguente linea di programma, i parametri da N a NS3 si riferiscono all'inviluppo di tono esaminato a pagina 408:

ENVELOPE N,T,PS1,PS2,PS3,NS1,NS2,NS3,AR,DR,SR,RR, FAL,FDL

I restanti parametri concernono l'inviluppo di volume e stabiliscono sia il volume di picco, che la variazione di volume durante l'emissione di un singolo suono definito dal relativo comando SOUND.
AR individua la percentuale di "Attacco" di un suono e benché possa variare tra −127 e 127, si ottengono risultati apprezzabili soltanto con valori positivi (da 1 a 127). Questo valore stabilisce gli incrementi di volume, per unità di tempo, da applicare al suono fino a raggiungere il livello di volume finale (fissato dal valore di FAL), che costituisce l'inizio della fase discendente. Analogamente, DR individua la percentuale di "Decadimento" ed i valori impiegati sono generalmente negativi. Il volume diminuisce gradualmente fino a raggiungere il livello indicato dal valore FDL (= livello di decadimento finale). Attualmente, l'hardware consente livelli di volume compresi soltanto tra 0 e 16, nonostante il software accetti un arco molto più ampio.
I parametri SR (percentuale di Persistenza) ed RR (percentuale di Rilascio) si riferiscono ambedue al graduale cambiamento di volume, ed assumono valori negativi (da −127 a 0). La Persistenza continua finché non si completa la durata stabilita dal comando SOUND. Ciò significa che se il tempo totale dell'Attacco e del Decadimento è maggiore o uguale alla Durata, non vi sarà alcuna Persistenza, anche se programmata. Il Rilascio, invece, inizia quando termina la Durata. Il volume cala gradualmente a zero, secondo la percentuale fissata da RR, salvo che non venga emesso un nuovo suono sullo stesso oscillatore, mediante un nuovo comando SOUND &.

Onde di luce

La grafica dell'Atari è tra le più interessanti

Gli home computer Atari 400 ed 800 sono ben noti per l'ampia gamma di giochi, forniti su cartucce. Comunque, anche il BASIC impiegato su questi apparecchi è ricco di ottimi comandi per la gestione della grafica, che, in ambedue i modelli, possiede ben nove livelli: tre per la visualizzazione di testi (con caratteri di diversa ampiezza) e sei per la grafica vera e propria. La massima risoluzione è di 320 × 192 punti.
La gamma di colori si estende su 16 diverse tonalità, ma il massimo numero di colori selezionabili contemporaneamente è di 5. Il set dei caratteri, oltre a comprendere lo standard ASCII (maiuscole e minuscole), è arricchito da 37 simboli speciali definiti dalla stessa Atari, grazie ai quali, mediante l'istruzione PRINT, si possono comodamente costruire tabelle semigrafiche. Il movimento del cursore si può comandare anche in ambiente BASIC, specificando la riga e la colonna desiderata all'interno dell'istruzione PRINT.
Una caratteristica molto attraente degli Atari è la possibilità di usare una grafica di tipo "a sprite", denominata "Giocatore/Missile" (o più brevemente PM, dall'inglese "Player/Missile"), che consente un'alta velocità d'azione nei giochi. Stranamente, però, non esistono appositi comandi o istruzioni BASIC per la gestione della grafica PM e tutto il lavoro di programmazione deve basarsi sull'impiego di PEEK e POKE. Parleremo della grafica PM più avanti nel corso.

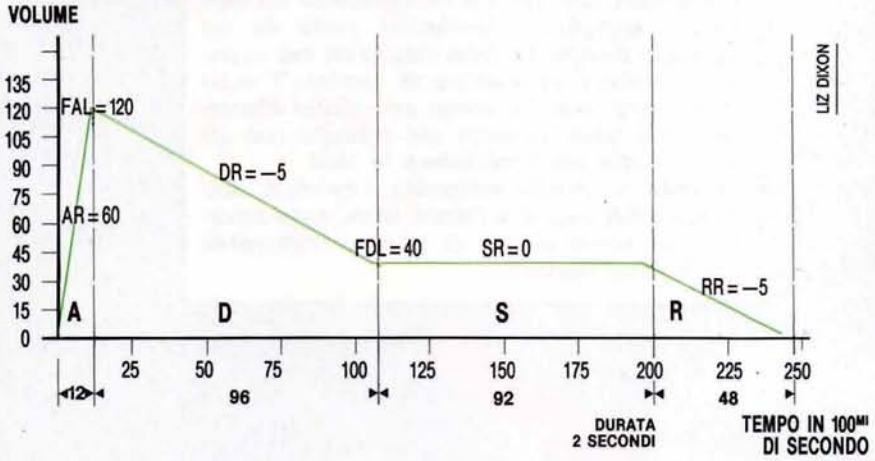
I Modi di visualizzazione

I Modi 0, 1 e 2 sono dedicati, come già accennato, alla visualizzazione di testi: all'accensione, l'apparecchio si predispone automaticamente in Modo 0 (lo schermo suddiviso in 24 linee di 40 caratteri ciascuna). In questo caso il set di caratteri ha il formato standard ASCII su matrice 8 × 8. Nel Modo 1, i caratteri riprodotti hanno larghezza doppia, ma altezza identica a quelli del Modo 0, mentre nel Modo 2, sia l'altezza che la larghezza sono doppi rispetto al Modo 0.
Ad eccezione del Modo 0, in tutti gli altri Modi lo schermo è suddiviso in due parti, ed un paio di righe, nella parte inferiore, vengono usate per l'immissione di comandi o per eventuali messaggi d'errore. Nei Modi 1 e 2, per la visualizzazione nella parte principale dello schermo occorre specificare un canale di output, all'interno dell'istruzione PRINT. Ad

Riferendosi al diagramma qui presentato, i valori necessari per un inviluppo che simuli un pianoforte sono:

T = 6 AR = 60 SR = 0 FAL = 120
DR = –5 RR = –5 FDL = 40

INVILUPPO DI VOLUME



durata del suono = 40 (due secondi).

Ovvero:

```
ENVELOPE 1,6,0,0,0,0,0,0,60,–5,0,–5,120,40
```

Il seguente programma utilizza tutti i comandi di BASIC offerti dal BBC, per generare un noto brano musicale, usando l'inviluppo per pianoforte, oltre ad un effetto speciale sull'accordo finale:

```
10 REM = = = = = = = = = = COSMIC = = = = = = = = = =
20 ENVELOPE 1,6,0,0,0,0,0,0,60,–5,0,–5,120,40
30 ENVELOPE 2,6,1,–1,1,1,2,1,60,–5,0,–5,120,40
40 FOR I = 1 TO 4 : READ N
50 SOUND 1,1,N,20:REM = = = SUONA: LA,SI,SOL,SOL = = =
60 SOUND &1001,0,0,5:NEXT I
70 SOUND &201,2,77,40:REM = = = = = FINALE = = = = =
80 SOUND &202,2,89,40:REM = = = RE MAGGIORE = = =
90 SOUND &203,2,109,40
100 DATA 137,145,129,85:REM = = = LA,SI,SOL,SOL = = =
```

esempio, PRINT#6. I Modi da 3 ad 8, grafici, consentono di tracciare linee sullo schermo, in differenti gradi di risoluzione.
Questa tabella riporta le combinazioni a disposizione:

MODO	TIPO	RIGHE	COLONNE	COLORI
0	testo	24	40	2
1	testo	20	20	5
2	testo	10	20	5
3	grafica	20	40	4
4	grafica	40	80	2
5	grafica	40	80	4
6	grafica	80	160	2
7	grafica	80	160	4
8	grafica	160	320	1

La scelta dipende dalla quantità di memoria disponibile. Per esempio, il Modo 5 (con 4 colori) impegna il doppio di memoria che non il Modo 4 (con soli 2 colori).

I comandi BASIC

I comandi BASIC per la grafica funzionano, più o meno con lo stesso formato, nei diversi modi di visualizzazione.
Esistono 5 registri di colore, anche se non tutti utilizzabili nei vari Modi. Il comando SETCOLOR a,b,c permette la selezione del colore, dove: a è il numero del registro di colore (da 0 a 4), b è il numero del colore da usare e c è il livello d'intensità, tra gli 8 disponibili, da usare (si adoperano numeri pari tra 0 e 14).
Il comando COLOR n funziona in due maniere differenti, a seconda se si usa il modo grafico o meno.

Nei Modi 0 o 1 o 2, il valore di n è compreso tra 0 e 255, che nella sua forma binaria assume il seguente significato: i primi sei bit si riferiscono al codice ASCII del carattere visualizzato, i restanti due al colore impiegato. Nei modi grafici, invece, il valore di n è compreso tra 0 e 3, per selezionare un particolare registro di colore durante il tracciamento di linee.
Sullo schermo dell'Atari, l'origine delle coordinate è in alto a sinistra. Il comando PLOT x,y permette di "accendere" o "spegnere" il punto di coordinate x,y.
Analogamente, il comando

```
POSITION x,y
```

consente di posizionare il cursore nel punto di coordinate x,y, mentre

```
DRAWTO x,y
```

offre la possibilità di tracciare una linea retta (o un'approssimazione, a seconda del Modo) tra l'ultima posizione del cursore e quella individuata dai valori x ed y.
Infine, la linea:

```
X10 18,#6,0,0,"S:"
```

sfrutta il comando Atari di input/output X10, per colorare l'interno di una figura disegnata sullo schermo.
I risultati di quest'ultimo comando sono ottimi, ma il suo impiego è particolarmente complesso: occorre posizionare il cursore nel punto più in basso a sinistra della superficie chiusa da colorare e quindi far eseguire la colorazione. Il colore viene scelto, mediante una POKE alla locazione 765, depositando il valore 1, 2 o 3, con significato identico a quanto detto per il comando COLOR.

Caratteri in risalto

La grafica dell'Atari è interessante, ma non particolarmente facile all'uso. La mancanza di comandi predefiniti, per le forme geometriche più comuni, si fa sentire. Tuttavia, c'è il vantaggio di disporre di più Modi di visualizzazione (anche per i testi). Il seguente programma illustra come adoperare il set di caratteri espansi assieme al comando POSITION

```
10 REM =CARATTERI ESPANSI=
20 GRAPHICS 2 + 16
30 SETCOLOR 0,3,6
40 FOR X=19TO8 STEP–1
50 POSITION X,1
60 FOR J=1TO100:NEXT J
70 PRINT#6;"IL "
80 NEXT X
90 FOR X=19TO6 STEP–1
100 POSITION X,3
110 FOR J=1TO100:NEXT J
120 PRINT#6;"MIO "
130 NEXT X
140 FOR X=13TO7 STEP–1
150 POSITION X,9
160 FOR J=1TO100:NEXT J
170 PRINT#6;"COMPUTER
    "
180 NEXT X
190 SETCOLOR 0,5,5
200 FOR Y=9TO5 STEP–1
210 POSITION 7,Y
220 PRINT#6;"È BELLO! "
230 NEXT Y
240 GOTO 240
```

Nota: nella selezione di un nuovo Modo, si può eliminare la suddivisione dello schermo in due zone aggiungendo 16 al numero del Modo

Il Computerese

I misteriosi termini di un gergo sempre più diffuso

In ogni settore delle attività umane esistono parole o frasi particolari, dal significato totalmente incomprensibile ai "non addetti". Il gergo del mondo dei computer è ricchissimo di termini, quasi esclusivamente inglesi, tanto che lo stesso gergo ha un nomignolo: Buzzwords. Tentiamo di spiegare i termini più ricorrenti.

BUZZWORDS

Il termine pare tragga origine da un giochetto inventato negli anni '60 da un dipendente del settore pubblicitario della Honeywell: *Il generatore di Buzzword*. In questo programma il computer associa casualmente tra loro alcuni aggettivi e sostantivi, scelti da un apposito elenco. Le frasi risultanti (ad esempio: "modulo interattivo di sistema") sono molti verosimili. Lo scopo era quello d'intercalare le frasi ottenute nei dialoghi con gli altri... tanto per confondere le idee.
Si veda, a questo proposito, l'esempio illustrato nella pagina a fianco, dove viene riportato un breve elenco di termini utilizzabili per le frasi casuali.

BOOT

Sembra che il termine derivi da "Bootstrap" (laccio da scarpone) e dal detto anglosassone "Tirarsi su per i lacci degli scarponi". Il programma di bootstrap è quello lanciato automaticamente dal computer all'accensione e "fare il boot al sistema" è un modo molto "in" per dire semplicemente "accendere il computer"!
In alcuni sistemi, il programma di "boot ", memorizzato nelle ROM, serve per leggere automaticamente il sistema operativo.

BIT

L'origine di questo termine è ampiamente giustificata: proviene da Binary digit (ossia cifra binaria). In inglese, il termine è particolarmente felice, poiché "bit", nel linguaggio corrente, significa anche "pezzettino piccolo piccolo", qual è, appunto, un bit.
Nella nostra lingua, se non altro, la pronuncia di questo termine è inequivocabile, a differenza di molti altri entrati prepotentemente a far parte del linguaggio comune.
Bit viene talvolta usato come prefisso, come nel caso di "Bit-slicing" (traducibile, più o meno, con "processo per affettare i bit"). Il Bit-slicing è un procedimento grazie al quale sofisticati microprocessori riescono ad elaborare informazioni composte da ben 32 bit.
Un altro esempio: "Bit-decay" (letteralmente, "carie dei bit"), che si riferisce ad un curioso fenomeno, quasi una legge: i programmi inutilizzati per lungo tempo accumulano nuovi "bug" (errori).

TURNKEY

Il senso di questo termine può essere reso, in italiano, con l'espressione "chiavi in mano". Infatti, esso indica un'apparecchiatura preparata appositamente affinché il cliente possa immediatamente adoperarla. Spesso la preparazione, effettuata da ditte specializzate, anche esterne alle organizzazioni di vendita, consiste nell'istallazione di software che faciliti le operazioni d'impiego: il cliente non ha che da "girare la chiave".

BASIC

L'azzeccatissima sigla BASIC proviene da: Beginner's All-purpose Simbolic Instruction Code (Codice d'istruzioni simboliche di applicazione generica per principianti).

HARDWARE SOFTWARE

Oltre a questi due termini (l'uno riferito al complesso dei macchinari, l'altro a tutto ciò che concerne i programmi) sono stati coniati altri due nuovi sostantivi: FIRMWARE (per indicare programmi memorizzati su ROM) e LIVEWARE per indicare tutti coloro che hanno la fortuna (o la sfortuna?) di lavorare coi computer.

BAUD

Il termine è usato per indicare la velocità di trasmissione di dati, in omaggio ad Emile Baudot, l'inventore del primo codice per telegrafo, soppiantato poi da quello di Morse.

BYTE

Non si ha una traccia precisa dell'origine di BYTE. Inizialmente, fino alla comparsa dei primi microprocessori, il termine non indicava un preciso numero di bit, ma semplicemente il gruppo di bit necessario per rappresentare un carattere su di un particolare computer (a volte, quindi, un byte era lungo 24 bit, altre addirittura 64!). Questo termine è stato preso come spunto per un numero incredibile di nomi di prodotti commerciali ed anche di ditte.
Due termini associati a byte sono: NYBBLE (metà byte) e GULP per indicare un piccolo gruppo di byte.

GARBAGE

Letteralmente "spazzatura". È stata addirittura coniata la sigla GIGO (Garbage In, Garbage Out), per ricordare che, se ad un programma vengono dati in pasto "rifiuti" (ossia dati scorretti), i risultati non possono che essere altri "rifiuti".

GARBAGE COLLECTION (Raccolta di rifiuti) è il nome dato ad un'importante funzione interna a molti linguaggi (ad esempio nella gran parte dei BASIC): il recupero di aree di memoria riutilizzabili durante l'esecuzione di un programma. L'esecuzione del programma applicativo viene momentaneamente sospesa (nei casi peggiori anche per diversi minuti) permettendo al Garbage Collector di duplicare, in una nuova zona di memoria, il contenuto "più recente" delle stringhe. La parte di memoria contenente le vecchie stringhe può così essere riutilizzata.
Questo processo è particolarmente utile nel software in cui viene fatto largo uso di variabili alfanumeriche, che altrimenti riempirebbero in breve tempo tutta la memoria disponibile.
In alcuni BASIC, tuttavia, il funzionamento del "Garbage Collector" è talmente rapido da non interferire apparentemente con il programma in corso d'esecuzione.

LOGIC BOMBS
TROJAN HORSES

Le espressioni "Bombe logiche" o i "Cavalli di Troia", in gergo, identificano due metodi a dir poco fraudolenti di cui fanno uso persone poco oneste. Il primo consiste in programmi o porzioni di essi, mimetizzati e "dormienti" all'interno di un sistema complesso, ma pronti ad "esplodere" al verificarsi di determinate condizioni (o al mancato verificarsi di esse, ad esempio: un aumento di stipendio del programmatore!), bloccando l'intero sistema. Il secondo, invece, pare si riferisca al "camuffamento" di un programma, che serve in realtà ad accedere a programmi o dati per i quali la persona non possiede autorizzazione (per esempio il programmatore esterno alla ditta presso la quale è istallato il software da lui creato).

HANDSHAKE

Il termine, letteralmente significa "stretta di mano", e nei computer, "handshake" si riferisce ad un particolare segnale, che viene inviato all'elaboratore quando uno scambio di dati si è completato con successo.

TIME BOMB

Le cosiddette "Bombe ad orologeria" sono comunemente adoperate dalle case produttrici di software, per proteggersi contro il fenomeno della "pirateria". Nei programmi viene inserito un meccanismo di protezione che, trascorso un certo tempo (o ad una precisa data), distrugge tutti i dati e, talvolta, lo stesso programma. Ovviamente, qualora l'istallazione venga effettuata da personale autorizzato, la "bomba" viene opportunamente disinnescata, consentendo di usare i programmi in tutta tranquillità. Spesso, purtroppo, simili meccanismi vengono adoperati anche da programmatori poco scrupolosi, per assicurarsi la "fedeltà" dei clienti (ovviamente, la presenza della "bomba" viene taciuta al momento dell'acquisto!).

0. integrato	0. database	0. rete
1. interattivo	1. top-down	1. capacità
2. bufferizzato	2. diagnostico	2. sistema
3. digitalizzato	3. indirizzamento	3. algoritmo
4. stocastico	4. lineare	4. processore
5. periferico	5. grafico	5. processo
6. euristico	6. alfanumerico	6. vettore
7. relazionale	7. immaginario	7. gerarchia
8. personalizzato	8. schematico	8. generatore
9. programmabile	9. centralizzato	9. interfaccia

Il "Buzzword generator"
In passato, c'è chi si è divertito alle spalle degli inesperti, generando frasi senza alcun senso, al computer. Le parole venivano scelte da un elenco di termini reali o realistici come quelli qui riportati. Il gioco si chiamava, appunto, "Buzzword generator".

Il Commodore PET 4032

Anatomia di un microcomputer a diffusione europea: tecnologie a confronto

Per certi versi, il Commodore PET è l'apparecchio che ha dato inizio al "boom" dei microcomputer. Da quando venne lanciato, nel 1977, il PET ha rappresentato per lungo tempo una pietra miliare cui ha fatto riferimento una gran parte della produzione successiva. Alcune caratteristiche del PET, anche a distanza di anni, sono ancora significative: per esempio, il contenitore metallico (spesso oggi si ricorre alla plastica, più economica) e l'alimentatore incorporato.
Quando fu messo in vendita il PET, esistevano già molti microcomputer, ma si trattava, in gran parte, di apparecchi sperimentali, montati su circuiti stampati e certo non adatti ad applicazioni commerciali ed il PET può ben essere considerato il primo microcomputer "pronto all'uso". Nella prima versione del PET erano incorporati un registratore a cassette (il cui motore era controllato dal computer), un monitor ed un BASIC residente in ROM. Per mettere l'apparecchio in funzione bastava collegarlo alla corrente, girare un interruttore e, quasi istantaneamente, compariva la scritta:

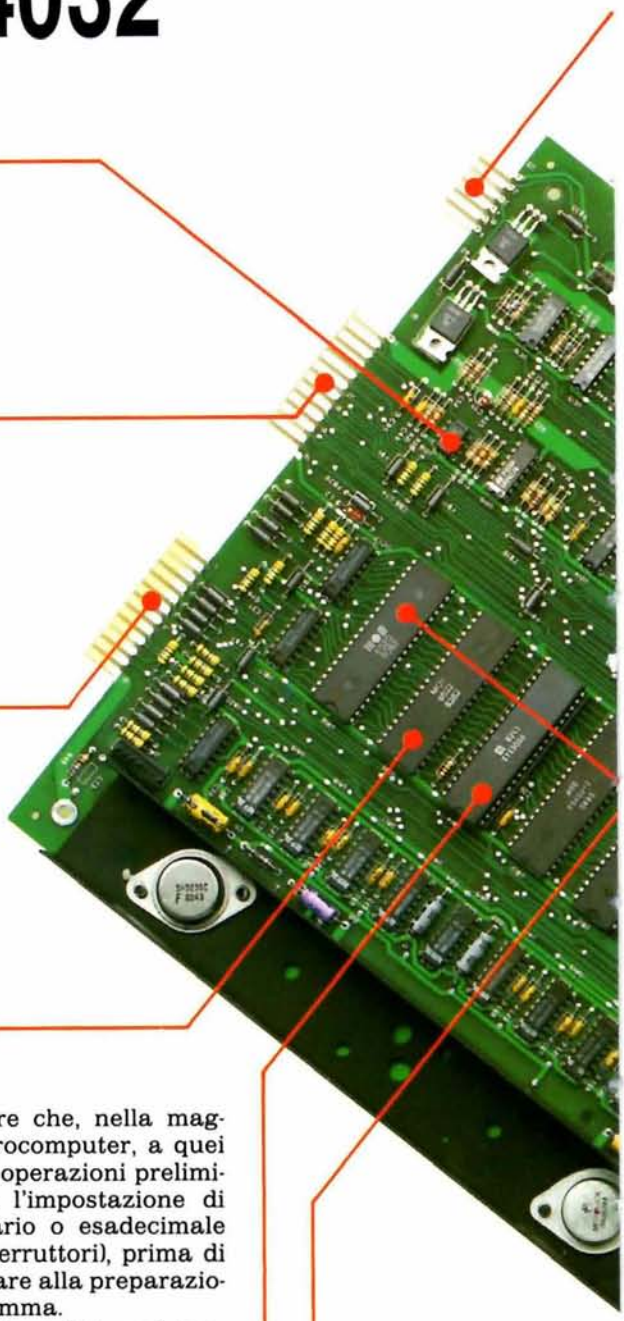
```
COMMODORE BASIC VER. 1.0
7167 BYTES FREE
READY
```

A questo punto, l'utente poteva iniziare ad immettere istruzioni BASIC o richiamare un programma precedentemente conservato su cassetta. Per comprendere il valore di questa innovazione, basti pensare che, nella maggior parte degli altri microcomputer, a quei tempi, occorrevano molte operazioni preliminari (compresa, talvolta, l'impostazione di comandi in formato binario o esadecimale mediante una serie di interruttori), prima di potersi (finalmente!) dedicare alla preparazione o all'uso di un programma.
Il BASIC del Commodore ha subìto, nel tempo, numerose revisioni, tanto da essere, oggi, quasi un dialetto a sé stante.
Un'altra peculiarità del PET è stata, fin dagli inizi, la disponibilità di simboli grafici predefiniti, oltre ai normali caratteri ASCII, sebbene i codici di questi ultimi non corrispondessero precisamente allo standard. Grazie ai numerosi simboli grafici, benché la risoluzione non fosse molto alta, era possibile creare immagini, per allora, di grandissimo effetto. Inoltre, l'impiego della grafica "a blocchi" permetteva di riprodurre su stampante le immagini generate sullo schermo, senza particolari difficoltà di programmazione. Ovviamente, in questo caso, non tutte le stampanti erano adatte ed era necessario acquistare quelle prodotte dalla Commodore.



La tastiera ed il monitor del PET
I primi PET non erano dotati di tastiere eccezionalmente comode, ma in seguito vennero sostituiti per assomigliare maggiormente a quelle di una normale macchina da scrivere. Il monitor è sempre stato incorporato nell'apparecchio: i più recenti modelli hanno uno schermo di 12" (30 cm.) con caratteri verdi su sfondo nero e la possibilità di scegliere tra una visualizzazione a 40 o 80 colonne

CHRIS STEVENS



Chip temporizzatore
All'accensione, questo circuito impiega qualche frazione di secondo per stabilizzarsi, dopo di che provvede ad avviare automaticamente il BASIC

Porta d'ingresso/uscita
Quest'interfaccia possiede numerose linee, utili per collegare tutta una gamma di periferiche (monitor, stampanti o apparecchi di misura)

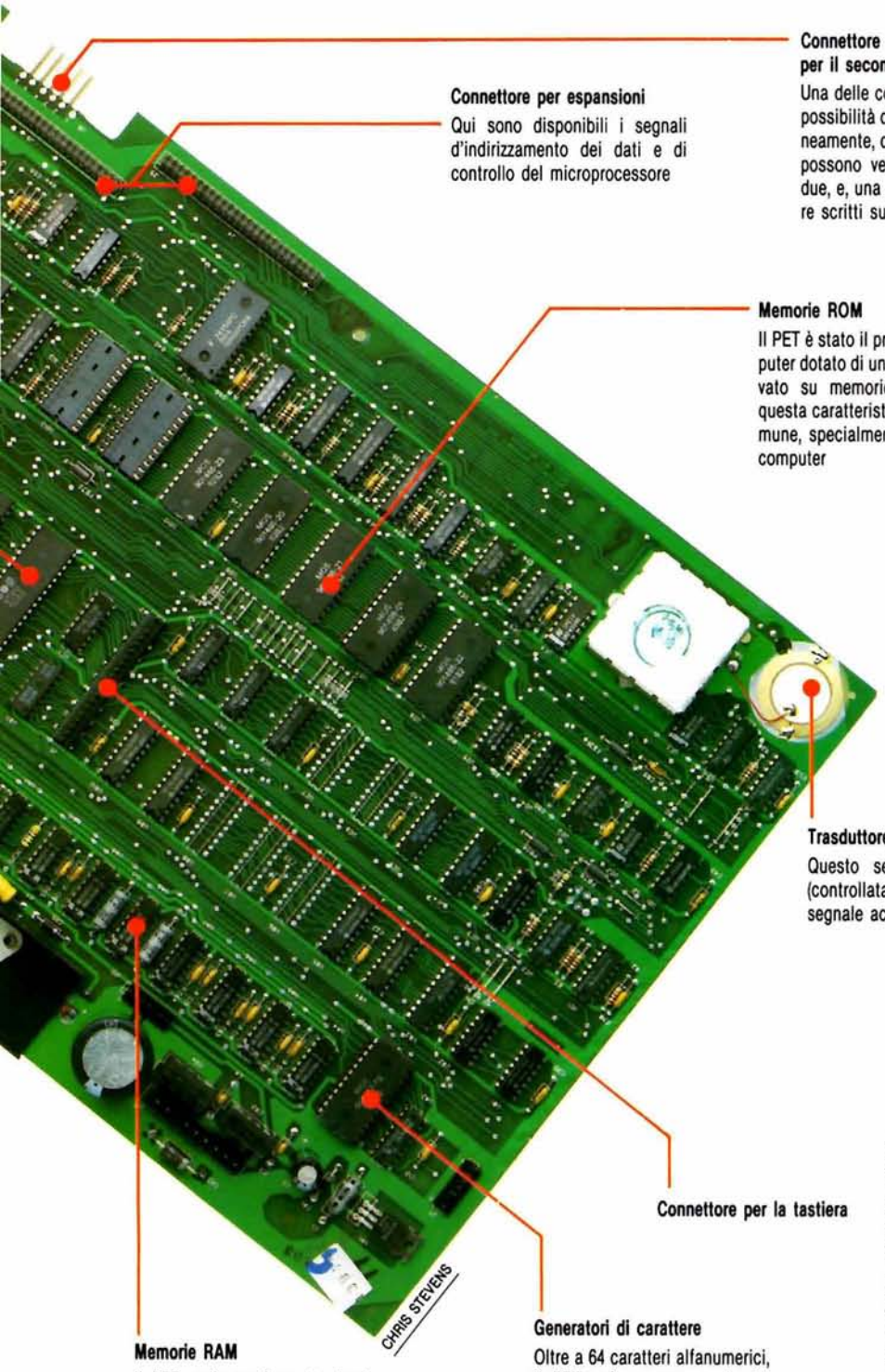
Porta IEEE 488
Il PET fu, per lungo tempo, l'unico microcomputer dotato di un'interfaccia rispondente allo standard IEEE 488, impiegato nella gran parte degli apparecchi scientifici

Chip 6522
Questo circuito, un versatile adattatore d'interfaccia, è simile al 6520, ma possiede, in più, un convertitore di segnali da seriali a paralleli e di due temporizzatori programmabili

Chip 6520
Questi sono dei circuiti adattatori d'interfaccia per periferiche, impiegati per controllare il registratore e la tastiera

CPU 6502
Il PET venne progettato da Chuck Peddle, padre del chip 6502. Non sorprende, quindi, il fatto che sia stato impiegato questo microprocessore ad 8 bit, ancora oggi considerato di alta qualità

Connettore per registratore
Si può collegare soltanto il registratore speciale prodotto dalla Commodore. Nei primi tempi, i comuni registratori non erano di qualità adeguata



Connettore per espansioni
Qui sono disponibili i segnali d'indirizzamento dei dati e di controllo del microprocessore

Connettore per il secondo registratore
Una delle comodità del PET è la possibilità di adoperare, simultaneamente, due registratori: i dati possono venir letti da uno dei due, e, una volta elaborati, essere scritti sull'altro

Memorie ROM
Il PET è stato il primo microcomputer dotato di un BASIC conservato su memorie ROM. Oggi, questa caratteristica è molto comune, specialmente negli home computer

Trasduttore piezoelettrico
Questo serve per l'emissione (controllata da software) di un segnale acustico

Connettore per la tastiera

Memorie RAM
Il PET, nella versione standard, viene venduto con una RAM da 8 Kbyte o da 32 Kbyte. Comunque, è possibile ampliare la capacità della RAM fino a 96 Kbyte

Generatori di carattere
Oltre a 64 caratteri alfanumerici, il PET può generare anche 64 simboli grafici. Alternativamente, si può scegliere la visualizzazione di caratteri sia maiuscoli che minuscoli

COMMODORE PET

DIMENSIONI
mm. 480 x 440 x 300

PESO
—

CPU
6502

CLOCK
1 MHz

MEMORIA
32 Kbyte di RAM (espandibile) a 96 Kbyte
20 Kbyte di ROM

VIDEO
25 linee di 40 caratteri. Monitor a fosfori verdi, da 12 pollici, incorporato. 256 caratteri visualizzabili. Grafica in bassa risoluzione (50 x 80)

INTERFACCE
Due interfacce per registratori a cassette. Due porte parallele (di cui una IEEE 488)

LINGUAGGI FORNITI
BASIC; Programma di controllo per linguaggio macchina

ALTRI LINGUAGGI DISPONIBILI
PASCAL, LISP, COMAL

ACCESSORI COMPRESI
Manuale operativo

TASTIERA
64 tasti, tipo macchina per scrivere. Tastierino numerico indipendente

DOCUMENTAZIONE
Trattandosi di uno dei primi apparecchi sul mercato, la documentazione non si può dire di qualità eccellente, specialmente se confrontata con il livello di quella fornita attualmente come standard

È stata sviluppata una notevole quantità di buon software per il PET, ma, a causa delle particolarità appena descritte, è difficile sia utilizzare programmi scritti per altri computer, che trasferire quelli del PET su altri apparecchi. Pertanto, questa macchina si è gradualmente trovata in una posizione d'isolamento, scarsamente toccata dai notevoli mutamenti avvenuti nel settore. La Commodore, nel frattempo, si è dedicata alla produzione di apparecchi più orientati al settore dell'home computer (il Vic-20 ed il C-64), cercando anche di allinearsi a caratteristiche più standard.

Elementi... sovversivi

Cercare e correggere gli errori in un programma richiede tempo: eliminiamo il problema alla radice

Migliorando le capacità di programmazione, il tempo necessario a rilevare e correggere gli inevitabili errori (operazione nota col termine "debugging") diminuisce gradualmente, poiché gli errori di sintassi e di logica si fanno via via meno frequenti. Qui di seguito suggeriamo qualche espediente per evitare gli errori più banali o per facilitare la procedura di "debug".

Il primo suggerimento (ed il più ovvio!) è di chiarirsi bene le idee sugli obiettivi di un programma, prima ancora di sedersi al computer. Spesso è utile discutere del funzionamento e dell'impostazione da dare al programma con altri: è incredibile quanto questo sforzo riesca a chiarire le idee. È necessario, poi, suddividere il problema in una serie di sezioni logiche (ad esempio, le caratteristiche dell'input, quelle dell'output, gli algoritmi da applicare, la struttura dei dati, le varie fasi dell'elaborazione, ecc.) ed affrontare ciascuna sezione individualmente. All'occorrenza, ogni sezione può esser scomposta in moduli ancora più semplici. L'impiego di uno pseudo-linguaggio, anche se non indispensabile, è certo utilissimo per fissare su carta le varie fasi dello sviluppo. Inoltre, è sempre buona norma costruirsi una documentazione quanto più completa possibile, sin dai primi stadi dello sviluppo: ciò eviterà d'incorrere in banali errori o dimenticanze, fonte di fastidiosissimi grattacapi durante le operazioni di "debug". In sostanza, si tratta di tenersi alla lontana dal computer il più a lungo possibile, fintanto che il problema sia stato sviscerato e risolto in ogni sua componente. Questi concetti sono alla base del metodo di programmazione definito "topdown" (dall'alto in basso).

Un vantaggio collaterale di una simile impostazione è la produzione di una raccolta di sottoprogrammi o moduli sufficientemente generici da essere riutilizzati per altri programmi, con conseguente risparmio notevole di tempo. Se, ad esempio, per un nuovo programma occorre il riordino alfabetico di una serie di nominativi, è addirittura sciocco riscrivere ogni volta da capo questa sezione, se è già stata inclusa in un programma precedente.

Una buona abitudine è quella di adoperare, per le variabili, nomi facilmente riconoscibili: NETTO = LORDO – TASSE, ad esempio, è molto chiaro; NT=LD–TX è già ambiguo, mentre N=L–T è totalmente oscuro, ai fini della comprensione del programma. Ahimè, non tutti i BASIC consentono, però, l'impiego di nomi sufficientemente lunghi. Comunque sia, conviene sempre mantenere un elenco (meglio se in ordine alfabetico) delle variabili usate, assieme ad una concisa spiegazione per ognuna di esse.

A CACCIA DI "BUG"

```
100 GOTO 200:X$="QUESTO E' TUTTO!"
120 I=12:K=1984
140 FOR K = 1 TO LT
160 PRINT "STRUTTURA DEL PROGRAMMA DI ";N$ : NEXT
180 RESTORE
190 FOR L=1 TO I
200 INPUT "COME TI CHIAMI":N$
220 INPUT "QUANTI ANNI HAI";LT
240 GOSUB 100
260 PRINT SE HAI ";LT;" ANNI"
280 PRINT SEI NATO NEL";K-LT
300 YR$=K-LT
320 LY = INT(YR)/4*4
340 IF LY=YR THEN IF INT(LY/100)*100=LY THEN GOTO 370
380 PRINT YR "ERA BISESTILE":GOTO 420
390 PRINT YR; "NON ERA BISESTILE"
400 NEXT
420 PRINT X$
440 STEP
```

- La seconda parte di questa linea non verrà mai eseguita, poiché la GOTO dirotta sempre l'esecuzione alla linea 200
- Queste due linee non sono nel corretto ordine. La linea 100 dovrebbe contenere GOTO 190
- Il valore di K dovrebbe essere una costante (linea 120), ma qui le viene assegnato un altro valore
- Poiché sono stati omessi i doppi apici, a chiusura della stringa da visualizzare, l'istruzione NEXT non verrà mai eseguita
- Fischi per fiaschi, ossia RESTORE al posto di RETURN
- Syntax Error: al posto di un punto e virgola, ci sono i due punti
- La linea specificata dovrebbe forse essere 140, ma non 100
- Mancano i doppi apici iniziali
- Il risultato di K–LT sarà certamente errato, per quanto detto sulla linea 140 (K non contiene più una costante)
- Syntax Error: non si possono mescolare variabili di tipo diverso. La linea dovrebbe 'eggersi: YR = K–LT
- La parentesi di chiusura è fuori posto. In questo caso la forma corretta è: INT(YR/4)*4
- Non esiste una linea 370!
- Al posto di ' dovrebbe esserci ;". Inoltre, la GOTO 420 provoca un salto indesiderato fuori dal ciclo
- Certi BASIC esigono che venga specificata la variabile del ciclo (in questo caso: L)
- Nel programma, ad X$ non viene assegnato alcun valore: questa linea non ha quindi alcuno scopo pratico
- Syntax Error: STEP al posto di STOP

L'elenco può servire per evitare pericolose ripetizioni, ma anche per ottimizzare l'impiego delle variabili (ad esempio, i contatori all'interno dei cicli FOR...NEXT).
Spesso nei programmi si adoperano dei valori costanti: ai fini della chiarezza è meglio assegnare questi valori a delle variabili e raggruppare tutte queste assegnazioni all'inizio del programma, anziché spargerle un po' dovunque. Ciò permette anche una più facile modifica di tali valori, qualora occorra.
Tuttavia, anche seguendo gli accorgimenti fin qui descritti, si possono commettere errori: è necessario, quindi, adottare un criterio anche nella ricerca e nella correzione di essi.
Gli errori più frequenti (spesso chiamati in gergo "bachi" o, in inglese, "bug") sono quelli di sintassi ed è in genere lo stesso BASIC ad informarci della loro presenza. Capitano, però, dei casi nei quali l'errore si cela in modo subdolo. Si consideri, ad esempio:

```
10 PRINT "I PEGGIORI ERRORI PROVENGONO"
20 PRINT "DA UNA MANCATA ANALISI PRELIMINARE"
```

Usando un monitor a 40 colonne, nel digitare la linea 10, composta esattamente da 40 caratteri, il cursore si porta automaticamente all'inizio della riga successiva, ed è facile dimenticarsi di premere il tasto RETURN. Come risultato, la linea 20 sarà accodata alla 10, producendo un errore in fase d'esecuzione difficile da rilevare.
Esistono, poi, dei messaggi d'errore piuttosto fuorvianti, come nel seguente caso:

```
10 FOR K = 1 TO 7 : READ N(K) : NEXT K
25 DATA 34,6.78,34,124.6,2O4,30,50
```

Solertemente, il BASIC ci avvertirà di un errore nella linea 10, mentre, in realtà, l'errore è nella digitazione di uno dei dati nella linea 25 (una O al posto di uno 0). Il BASIC, naturalmente, ha ragione nell'informarci che l'anomalia è stata riscontrata durante la lettura di dati (READ N(K), nel nostro caso), ma lascia a noi il compito d'individuare la vera causa!
La difficoltà maggiore, comunque, risiede nell'individuare gli errori che non appartengono a quelli di sintassi e che, quindi, spesso non causano né la visualizzazione di un messaggio, né l'interruzione del programma, ma soltanto risultati errati. In questi casi occorre essere scrupolosamente metodici. Se il programma è ben strutturato, può essere facile "isolare" il modulo responsabile del difettoso funzionamento, magari ricorrendo al comando TRACE, che visualizza il numero della linea di programma in corso d'esecuzione. Se il BASIC è sprovvisto di TRACE, allora è necessario inserire delle linee aggiuntive nel programma (ad esempio: PRINT "LINEA 150" all'inizio della subroutine che inizia, appunto, alla linea 150). Inoltre, si può inserire qualche STOP, in posizioni strategiche, per sospendere l'esecuzione e consentire l'esame delle variabili. Ciò si ottiene o mediante l'impiego del comando PRINT, oppure scrivendo un'apposita sezione di programma, ad esempio:

```
11000 REM = = = = = ESAME DELLE VARIABILI = = = = =
11100 PRINT "PUNTEGGIO, DIMENSIONE, FLAG"
11200 PRINT SC,SZ,F1,F2
11300 PRINT "MATRICE DELLA SCACCHIERA"
11400 FOR K=1 TO 10:PRINT B$(K):NEXT K
```

Di conseguenza, quando il programma incontra una STOP, basta impartire il comando GOTO 11000 per esaminare le variabili che interessano. All'occorrenza, si può cambiare il valore di una delle variabili (digitando, ad esempio, SZ=17, seguito dalla pressione del tasto RETURN) e poi riprendere l'esecuzione mediante il comando CONT.
Si presti attenzione ad apportare una sola modifica alla volta, altrimenti si rischia di non saper più distinguere quale rimedio abbia effettivamente "curato" l'errore.
I cicli, in special modo quelli nidificati, e le diramazioni (IF...THEN GOTO) costituiscono un terreno fertilissimo per i "bachi"! Si consideri la seguente situazione:

```
460 IF SM < 0 AND SC < > −1 THEN IF SC > 0 OR SM = SC−
    F9 THEN LT = 500
470 FOR C1 = 1 TO LT:FOR C2 = LT TO C1 STEP −1
480 SC = SM + SC*C2
490 NEXT C2:SM = 0:NEXT C1
```

Neanche il programmatore che ha scritto queste righe di programma sarebbe capace di dipanare questa matassa, in caso d'errore! Un'altra "zona pericolosa" è costituita da linee contenenti più istruzioni, la prima delle quali è una frase IF...THEN:

```
655 IF A$ = "" THEN GOTO 980 : A$=B$
660 PRINT A$
```

La parte A$ = B$ (linea 655) non verrà mai eseguita: infatti, se la condizione A$ = "" è vera, l'esecuzione passa alla linea 980, mentre, in caso contrario, il resto della linea viene ignorato e l'esecuzione "scende" alla linea 660.
Come abbiamo detto in principio, l'esperienza è la migliore maestra per evitare errori, specialmente se aiutata da un criterio di lavoro ordinato e metodico.



TONY LODGE

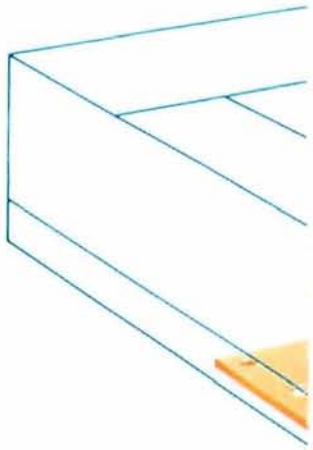
IL PRIMO BUG

Ai programmatori alle prime armi, i "bug" o "bachi" sembrano proprio dotati di vita propria: pare che si muovano, nascondendosi, da un punto all'altro del programma.
Comunque, il termine "bug" deriva proprio da un cattivo funzionamento dovuto ad un animaletto vivente. Nel 1945, Grace Hopper (vedere a pag. 440) stava tentando di individuare e correggere un errore di un programma, sviluppato sul computer MARK II presso la Harvard University. Si scoprì che l'errore era dovuto ad una falena, che, attratta forse dai bagliori delle numerose valvole, era riuscita a bloccare uno dei delicati meccanismi

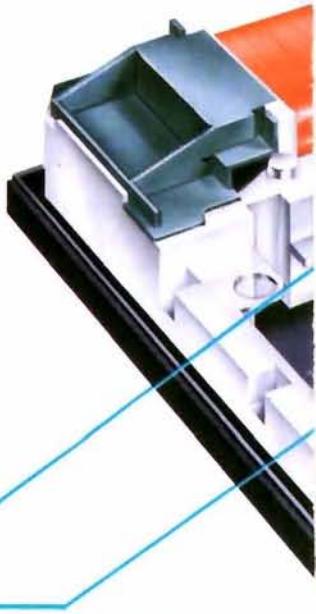
I dischi a laser

La tecnologia dei dischi ottici (a laser) apre nuovi orizzonti per l'informatica

Invariabilmente, nelle conversazioni incentrate sui personal e sugli home computer, l'argomento più ricorrente è la quantità di memoria RAM. Certo, questo fattore è molto importante, ma anche le caratteristiche delle memorie di massa sono altrettanto importanti. Infatti, anche i possessori di home computer, nel breve giro di qualche mese, accumulano una notevole quantità di programmi e dati, occupando numerosi nastri o dischetti. Molti programmi, una volta redatti nella forma definitiva (ed opportunamente verificati) non necessitano di alcun ulteriore intervento, e sarebbe molto più comodo, oltreché sicuro, conservarli su dei supporti "a sola lettura", simili a delle ROM o a delle cartucce, ma di capacità molto maggiore.
Tali supporti esistono, e sono i dischi ottici a laser. Attualmente, però, questi dispositivi sono costosi e vengono, per lo più, impiegati al posto dei videoregistratori per conservare un'incredibile quantità d'immagini. In sostanza, si tratta dell'equivalente visivo dei dischi laser musicali, che vanno sostituendo i tradizionali dischi con lettura mediante puntina.
La principale differenza tra i due sistemi, oltre al diverso aspetto esteriore dei dischi, è il principio di funzionamento. Mentre i videodischi conservano segnali in forma analogica, nei "compact disk" i segnali sono in forma digitale (ossia, una sequenza di 1 e 0), ed i suoni originali vengono "ricomposti", impiegando uno speciale convertitore digitale/analogico. Per le registrazioni video non conviene utilizzare dei "floppy disk" (simili a quelli dei computer), poiché nell'ambito domestico sono molte le fonti di campi magnetici o elettrici che li potrebbero danneggiare. In ogni caso, la capacità di un disco ottico può facilmente superare le svariate migliaia di megabyte, molto più di quanto si possa ottenere anche con i dischi Winchester.
In commercio esistono già numerosi modelli di sistemi a dischi ottici ed uno tra i migliori è quello della Philips. I dischi, in materiale plastico, hanno un diametro di 14 pollici (35 cm.) e le informazioni sono, letteralmente, inglobate nella plastica, sotto forma di microscopiche incisioni su di una lamina metallica, protetta dallo strato plastico. Così come accade nei dischetti da computer, sul disco ottico esiste una parte riservata all'indice delle immagini che consente all'apparecchio di posizionarsi rapidamente su di una particolare immagine richiesta dall'utente. Dopo essersi posizionata correttamente, la testina di lettura, mediante un raggio laser, riflesso dalla superficie metallica incisa, "legge", attraverso una cellula fotosensibile, le informazioni. Una singola immagine video è codificata lungo una spirale e, poiché ad ogni giro corrisponde un'immagine, il numero totale delle immagini memorizzabili è 54.000 per ciascuna facciata.
Esistono due principali applicazioni dei dischi laser nel campo dell'informatica. Il primo, già largamente diffuso, è chiamato "video interattivo". A differenza di un programma televisivo tradizionale, sul quale lo spettatore non ha alcun potere d'intervenire sulla sequenza delle immagini, nel "video interattivo" si può selezionare a piacimento l'ordine di presentazione delle varie immagini. Un video disco, ad esempio, potendo contenere sia immagini che pagine di testi, può venir impiegato come biblioteca di consultazione, sfruttando il proprio televisore di casa. L'apparecchio di lettura dei dischi può esser dotato di un microprocessore interno, oppure può essere collegato ad un microcomputer esterno. In ambedue i casi, con pochi comandi si possono effettuare velocissime ricerche per visualizzare le immagini desiderate. Questi sistemi vengono già impiegati con successo (in Italia dalla IBM e dalla FIAT) per corsi di aggiornamento professionale.
Il secondo campo di applicazioni consiste nell'impiego dei dischi ottici come futuro supporto per il software da computer. Si immagini la comodità di poter disporre di un supporto, praticamente indistruttibile, sul quale è memorizzato qualche centinaio (o anche qualche migliaio) di programmi. Sebbene gli sviluppi in questo settore siano ancora allo stadio sperimentale, è certo che saranno presto disponibili dei dischetti laser per computer. Due note case, la Sony e la Philips, hanno infatti preannunciato l'imminente produzione di unità a dischi laser per computer, ai quali è stata assegnata la sigla CDROM.

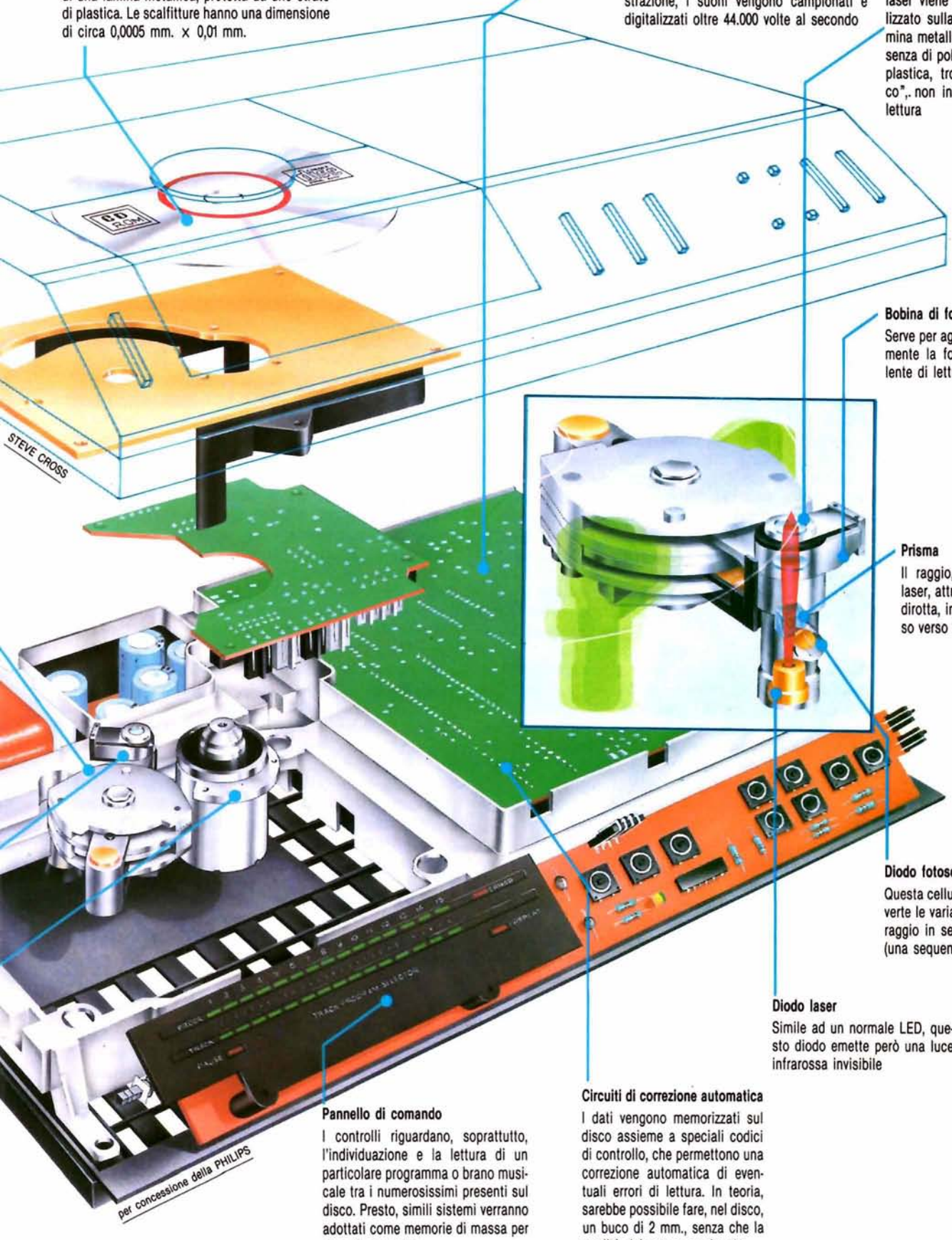


Motore lineare
Il servomeccanismo impiegato per lo spostamento del braccio di lettura consiste semplicemente in un elettromagnete collegato ad una molla di richiamo. Il principio è identico a quello usato nei vecchi strumenti di misura a lancette



Braccio di lettura
Il braccio, imperniato al suo centro, è perfettamente bilanciato. La lettura avviene ispezionando, con un raggio laser, i solchi del disco

Motore
La rotazione del disco è accuratamente controllata e mantenuta a livelli costanti da uno speciale circuito. A seconda della posizione del braccio di lettura, la velocità varia da 200 a 500 giri al minuto, ottenendo una densità costante delle informazioni



Disco

Le informazioni sono memorizzate, in forma digitale, mediante microscopiche scalfitture su di una lamina metallica, protetta da uno strato di plastica. Le scalfitture hanno una dimensione di circa 0,0005 mm. x 0,01 mm.

Elaborazione digitale

Nei due sistemi Sony e Philips i dati hanno un formato di 16 bit, consentendo ben 65.536 combinazioni diverse. In fase di registrazione, i suoni vengono campionati e digitalizzati oltre 44.000 volte al secondo

Lente

Grazie ad una lente, il fascio laser viene perfettamente focalizzato sulla superficie della lamina metallica. Pertanto, la presenza di polvere sulla superficie plastica, trovandosi "fuori fuoco",. non infastidisce affatto la lettura

Bobina di focalizzazione

Serve per aggiustare accuratamente la focalizzazione della lente di lettura

Prisma

Il raggio, emesso dal diodo laser, attraversa il prisma, che dirotta, invece, il raggio riflesso verso il diodo fotosensibile

Diodo fotosensibile

Questa cellula fotosensibile converte le variazioni d'intensità del raggio in segnali elettrici binari (una sequenza di 1 e 0)

Diodo laser

Simile ad un normale LED, questo diodo emette però una luce infrarossa invisibile

Circuiti di correzione automatica

I dati vengono memorizzati sul disco assieme a speciali codici di controllo, che permettono una correzione automatica di eventuali errori di lettura. In teoria, sarebbe possibile fare, nel disco, un buco di 2 mm., senza che la qualità del suono ne risenta

Pannello di comando

I controlli riguardano, soprattutto, l'individuazione e la lettura di un particolare programma o brano musicale tra i numerosissimi presenti sul disco. Presto, simili sistemi verranno adottati come memorie di massa per piccoli computer

Ritocchi finali

Completiamo la nostra rubrica collegando i moduli sviluppati

Abbiamo terminato l'ultima lezione, ponendo un quesito: come mai, lanciando il programma, aggiungendo un record e, successivamente, localizzando un record (subroutine FNDREC) ed uscendo dal programma (EXPROG), il record aggiunto non verrebbe memorizzato? Abbiamo usato la variabile RMOD per segnalare al programma quando occorra riordinare i record della rubrica, a seguito di una qualche modifica. La subroutine SRTREC, una volta eseguito il riordino alfabetico dei nominativi, assegna ad RMOD il valore 0, per indicare che i record sono in ordine. La routine necessaria per uscire da programma (EXPROG) controlla il valore di RMOD e, se questo è 0, conclude l'esecuzione, saltando la fase di memorizzazione dei record.
L'immissione di un nuovo record, giustamente, implica una modifica del valore di RMOD, portandolo ad 1 (per segnalare che è stata apportata una modifica ai dati), ma il conseguente riordino alfabetico azzera, come si è detto, la variabile RMOD, traendo così in inganno la routine EXPROG. Abbiamo dunque tentato di usare una sola "flag", quando, in realtà, ne occorrono due: una per segnalare eventuali modifiche apportate ai dati (RMOD) ed una, che chiameremo SRTD, per indicare se è stato eseguito il riordino oppure no.
Nella versione del programma, riportata a pagina 399, alla linea 1230 si può notare l'inizializzazione di una variabile, SVED, che non viene poi utilizzata nel programma. Essa era stata inclusa proprio prevedendo la possibile necessità di un interruttore "logico" durante le fasi relative alla memorizzazione dei dati. Ai fini di una maggiore chiarezza del listato, abbiamo sostituito la linea 1230 con:

```
1230 LET SRTD = 1
```

Adesso, in base alle due variabili RMOD e SRTD, dobbiamo costruirci una "tavola della verità", per verificare tutte le possibili situazioni nelle quali si potrebbe trovare il programma:

RMOD	SRTD	DATI
0	0	Non modificati, non riordinati
1	0	Modificati, non riordinati
0	1	Non modificati, ma riordinati
1	1	Modificati e riordinati

Notiamo immediatamente che la condizione RMOD = 0 e SRTD = 0 non deve presentarsi. Infatti, il programma provvede sempre a riordinare i dati prima del salvataggio. Quando il programma viene lanciato per la prima volta, RMOD viene azzerata (linea 1220) ed a SRTD viene assegnato il valore 1 (linea 1230). Qualsiasi operazione che implichi una modifica dei dati (mediante le routine AGGREC, DELREC o MODREC) provoca l'assegnamento, a RMOD, del valore 1 e questo valore rimane inalterato per tutto il resto delle operazioni. La variabile SRTD, inizialmente posta ad 1, è azzerata da tutte quelle operazioni che implicano una modifica nell'ordine dei dati (quale, ad esempio, la routine MODREC, per la modifica di un record). Ovviamente, ogni azione che basi il proprio funzionamento sull'ipotesi che i dati siano riordinati (FNDREC, ad esempio), deve controllare il valore di SRTD ed eseguire il sort qualora esso valga 0. Usando due flag, al posto della sola RMOD, potremo star certi che, terminando regolarmente il programma, non sorgeranno inconvenienti tali da causare la perdita di eventuali nuovi nominativi aggiunti.
La seconda variabile, inizializzata ma non utilizzata finora, è CORR. Questa viene impiegata per conservare il numero di record "corrente", dopo averlo individuato usando la subroutine di ricerca. Infatti, le ultime linee della subroutine FNDREC (la 3320 e la 3330), sono state modificate in modo da assegnare a CORR il valore 0, se la ricerca dà esito negativo, o il valore MED (la posizione del record cercato), in caso contrario. CORR non viene più modificata ulteriormente, e porta a spasso per tutto il resto del programma (fino ad una nuova ricerca) il numero del record "corrente", a disposizione di tutte le subroutine che ne possano aver bisogno.
Se il valore di CORR è 0, la linea 13340 devia il percorso dell'esecuzione verso la subroutine NOTREC, che visualizza, assieme ad un messaggio di ricerca "fallita", la chiave di ricerca, ovvero CAMNOM$(DIMENSIONE). La subroutine NOTREC ritorna al programma principale non appena viene premuto il tasto dello spazio, ma niente impedisce di modificarla, in modo da consentire un nuovo tentativo di ricerca:

```
PREMERE RETURN PER UN NUOVO TENTATIVO OPPURE
SPAZIO PER TORNARE AL PROGRAMMA PRINCIPALE
```

A prima vista, la soluzione più semplice potrebbe sembrare quella di richiamare la subroutine FNDREC nel caso venga premuto il tasto RETURN. Tuttavia, sebbene in BASIC non sia proibito inserire una chiamata ad una subroutine all'interno di sé stessa, un metodo simile è sconsigliato, perché potrebbe facilmente confondere lo stack del BASIC, causando tutta una serie di fastidiosi problemi. Una soluzione abbastanza semplice, anche se formalmente non molto elegante, potrebbe

essere quella di utilizzare una variabile come flag (chiamiamola NREC), azzerandola nella subroutine NOTREC, subito prima di tornare al programma principale. Una verifica di NREC (ad esempio: 95 IF NREC = 0 THEN 80) forzerebbe l'esecuzione a ripetere EXECUT. Per non complicare troppo il programma (e anche per evitare uno smodato uso di GOTO abbiamo preferito semplicemente tornare al programma principale, se la ricerca di un particolare record fallisce.
Si noti una piccola modifica, apportata alla linea 10490 (subroutine MODNOM). L'azzeramento della variabile S (LET S = 0). Omettendola, si rischia che, in particolari circostanze, la subroutine non funzioni correttamente.
Un'altra routine, aggiunta nella versione definitiva del programma, è la MODREC, che serve per trovare un particolare record (chiamando FNDREC dalla linea 14120) ed alla sua successiva modifica. Si noti, però, che invece di GOSUB 13000, abbiamo usato GOSUB 13030, per evitare una pulizia dello schermo. Se la ricerca del record ha buon esito, il programma visualizza il record ed appare:

MODIFICA AL NOMINATIVO?
PREMERE RETURN PER MODIFICARE IL NOMINATIVO
OPPURE SPAZIO PER IL CAMPO SUCCESSIVO

La routine che permette la scelta è compresa tra la linea 14190 e la 14280.
Le linee da 14190 a 14220 contengono un semplice ciclo, dal quale si esce soltanto premendo il tasto RETURN o uno spazio. Se A$ non corrisponde né a CHR$(13), ossia al codice ASCII corrispondente a RETURN, né al carattere spazio (il cui codice è 32, ma qui viene espresso più semplicemente come " "), allora la variabile I viene posta uguale a 0, ed il ciclo si ripete. Qualora si prema il tasto RETURN, le righe che seguono permettono l'inserimento di un nuovo valore in CAMMOD$(CORR), modificano opportunamente le variabili RMOD e SRTD, chiamano la subroutine MODNOM e provvedono a collocare il nominativo in forma standardizzata nella variabile CAMMOD$(CORR).
Il resto di MODNOM ha un funzionamento analogo a quello appena descritto, salvo per il fatto che, mentre viene modificata la variabile RMOD (per indicare l'avvenuta modifica), il valore di SRTD non viene cambiato, in quanto, non avendo modificato il nominativo, non è necessario riordinare alfabeticamente i dati.
Veniamo, infine, alla subroutine DELREC, per l'annullamento di un record. Essa non presenta particolari problemi: viene ripulito lo schermo (linea 15020) e presentato un messaggio per informare quale sia l'operazione in corso. Dopo aver chiamato FNDREC, per localizzare il record da eliminare, vengono offerte due possibilità: cancellare il record (pressione del tasto RETURN) oppure tornare al programma principale senza alcun annullamento (pressione di uno spazio). Viene anche visualizzato un messaggio d'avvertimento (linea 15160) per evitare errori. Per una maggiore salvaguardia dei dati, si potrebbe aggiungere qualche linea in più, in modo da chiedere conferma dell'annullamento (nel caso sia stato premuto un RETURN) e procedere soltanto se la risposta è un carattere S (per sì), ad esempio: IF INKEY$ = "S" THEN... Non occorre modificare il valore all'interno di DELREC, in quanto i dati sono già in ordine alfabetico e l'eliminazione di uno di essi non modifica affatto tale ordine. Tuttavia, essendo stata apportata una modifica, è necessario segnalare il fatto mediante RMOD, che viene azzerata alla linea 15340. Inoltre, si diminuisce di 1 il valore di DIMENSIONE (linea 15350), per tener conto dell'avvenuta eliminazione di un record. Per riempire il vuoto creatosi, tutti gli elementi vengono "scalati" di una posizione (linee da 15260 a 15320).
Si sarà osservato che in FNDREC è presente una chiamata, subordinata al fatto che il contenuto di A$ sia diverso da uno spazio, per l'output su stampante del record individuato dalla subroutine. Non disponendo di una stampante, si può disabilitare la chiamata, inserendo una REM all'inizio della linea 13540 e, volendo, eliminare completamente le linee da 13600 a 13690.
Un'ultima modifica, essenziale per chi intenda sfruttare il programma al massimo, la lasciamo come esercitazione: come gestire il caso in cui si riempia totalmente la rubrica.
Proseguendo nel corso di programmazione, verranno affrontati altri interessanti progetti pratici.

Basic & Basic

LPRINT

Questo comando non è disponibile sui seguenti apparecchi: Commodore 64, Vic-20, Micro BBC e Dragon 32.

Sul Micro BBC, dotato di stampante parallela, inserire:

```
13605 VDU 2
13680 VDU 3
```

che, rispettivamente, abilitano e disabilitano la stampante. Occorre anche sostituire tutte le LPRINT con delle PRINT, nelle linee da 13610 a 13670. Per ulteriori informazioni, consultare il manuale operativo.

Sui Commodore, inserire le linee:

```
13605 OPEN 4,4:CMD 4
13680 PRINT #4:CLOSE 4
```

che abilitano e disabilitano la stampante. Sostituire con delle PRINT tutte le LPRINT comprese tra la linea 13610 e 13670.

Sul Dragon 32 inserire le linee:

```
13605 OPEN "O",—2
13680 CLOSE —2
```

che abilitano e disabilitano, rispettivamente, la stampante. Si sostituisca ogni LPRINT con PRINT —2, (la virgola fa parte del comando)

SPECTRUM

Il programma completo per lo Spectrum verrà presentato nella prossima lezione

PROGRAMMA PER LA RUBRICA DEGLI INDIRIZZI

```
10 REM 'MAINPG'
20 REM == INITL ===
30 GOSUB 1000
40 REM == GREETS ==
50 GOSUB 3000
60 REM == SCELTA ==
70 GOSUB 3500
80 REM == EXECUT ==
90 GOSUB 4000
100 IF SCEL () 9 THEN 60
110 END
1000 REM SUBROUTINE == INITIL ==
1010 GOSUB 1100 : REM == CREMAT == (CREAZIONE MATRICI)
1020 GOSUB 1400 : REM == RDINFL == (LETTURA DEL FILE)
1030 GOSUB 1600 : REM == SETFLG == (ASSEGNAZIONE DELLE FLAG)
1040 REM
1050 REM
1060 REM
1070 REM
1080 REM
1090 RETURN
1100 REM SUBROUTINE == CREMAT == (CREAZIONE MATRICI)
1110 DIM CAMNOM$(50)
1120 DIM CAMMOD$(50)
1130 DIM CAMVIA$(50)
1140 DIM CAMLOC$(50)
1150 DIM CAMNAZ$(50)
1160 DIM CAMTEL$(50)
1170 DIM CAMIND$(50)
1180 REM
1190 REM
1200 REM
1210 LET DIMENSIONE = 0
1220 LET RMOD = 0
1230 LET SRTD = 1
1240 LET CORR = 0
1250 REM
1260 REM
1270 REM
1280 REM
1290 REM
1300 RETURN
1400 REM SUBROUTINE == RDINFL == (LETTURA DEL FILE)
1410 OPEN "I",#1,"RUBR.DAT"
1420 INPUT #1,TEST$
1430 IF TEST$ = "@PRIMO" THEN GOTO 1540:REM CHIUDE E RITORNA
1440 LET CAMNOM$(1) = TEST$
1450 INPUT #1,CAMMOD$(1),CAMVIA$(1),CAMLOC$(1),CAMNAZ$(1),
     CAMTEL$(1)
1460 INPUT #1,CAMIND$(1)
1470 LET DIMENSIONE = 2
1480 FOR L = 2 TO 50
1490 INPUT #1,CAMNOM$(L),CAMMOD$(L),CAMVIA$(L),CAMNAZ$(L)
1500 INPUT #1,CAMLOC$(L),CAMTEL$(L),CAMIND$(L)
1510 LET DIMENSIONE = DIMENSIONE + 1
1520 IF EOF(1) = -1 THEN L = 50
1530 NEXT L
1540 CLOSE #1
1550 RETURN
1600 REM SUBROUTINE == SETFLG == (ASSEGNAZIONE DELLE FLAG)
1610 REM LE FLAG VENGONO ASSEGNATE DOPO LA LETTURA DEL FILE
1620 REM
1630 REM
1640 IF TEST$ = "@PRIMO" THEN LET DIMENSIONE = 1
1650 REM
1660 REM
1670 REM
1680 REM
1690 RETURN
3000 REM SUBROUTINE == GREETS == (MESSAGGIO INTRODUTTIVO)
3010 PRINT CHR$(12) : REM PULIZIA DELLO SCHERMO
3020 PRINT
3030 PRINT
3040 PRINT
3050 PRINT
3060 PRINT TAB(22);"RUBRICA COMPUTERIZZATA"
3070 PRINT TAB(23);"PRESENTATA NEL CORSO"
3080 PRINT TAB(24);"'IL MIO COMPUTER'"
3090 PRINT
3100 PRINT TAB(15);"(PREMERE UNO SPAZIO PER CONTINUARE)"
3110 FOR L = 1 TO 1
3120 IF INKEY$ () " " THEN L = 0
3130 NEXT L
3140 PRINT CHR$(12)
3150 RETURN
3500 REM SUBROUTINE == SCELTA == (MENU DELLE OPZIONI)
3510 REM
3520 IF TEST$ = "@PRIMO" THEN GOSUB 3860: REM (FILE VUOTO)
3530 IF TEST$ = "@PRIMO" THEN RETURN
3540 REM == MENU ==
3550 PRINT CHR$(12)
3560 PRINT TAB(20);"MENU DELLE OPZIONI"
3570 PRINT
3580 PRINT
3590 PRINT
3600 PRINT TAB(8);"1. TROVARE UN RECORD (USANDO IL NOME)"
3610 PRINT TAB(8);"2. TROVARE UN RECORD (USANDO PARTE DEL NOME)"
3620 PRINT TAB(8);"3. TROVARE RECORD (USANDO LA LOCALITA')"
3630 PRINT TAB(8);"4. TROVARE UN RECORD (USANDO LE INIZIALI)"
3640 PRINT TAB(8);"5. ELENCO DI TUTTI I RECORD"
3650 PRINT TAB(8);"6. INSERIMENTO DI UN RECORD"
3660 PRINT TAB(8);"7. MODIFICA DI UN RECORD"
3670 PRINT TAB(8);"8. ANNULLAMENTO DI UN RECORD"
3680 PRINT TAB(8);"9. USCITA DAL PROGRAMMA"
3690 PRINT
3700 PRINT
3710 REM == INCHOI ==
3720 REM
3730 LET L = 0
3740 LET I = 0
3750 FOR L = 1 TO 1
3760 PRINT TAB(8);"QUALE (1 - 9) ?";
3770 FOR I = 1 TO 1
3780 LET A$ = INKEY$
3790 IF A$ = "" THEN I = 0
3800 NEXT I
3810 LET SCEL = VAL(A$)
3820 IF SCEL ( 1 THEN L = 0
3830 IF SCEL ) 9 THEN L = 0
3840 NEXT L
3850 RETURN
3860 REM SUBROUTINE == FIRSTM == (FILE VUOTO)
3870 LET SCEL = 6
3880 PRINT CHR$(12)
3890 PRINT
3900 PRINT TAB(8);"L'ARCHIVIO E' VUOTO ED OCCORRE"
3910 PRINT TAB(8);"INIZIARE INSERENDO IL PRIMO"
3920 PRINT TAB(8);"RECORD DEL FILE"
3930 PRINT
3940 PRINT TAB(5);"(PREMERE UNO SPAZIO PER CONTNUARE)"
3950 FOR B = 1 TO 1
3960 IF INKEY$ () " " THEN B = 0
3970 NEXT B
3980 PRINT CHR$(12) : REM PULIZIA DELLO SCHERMO
3990 RETURN
4000 REM SUBROUTINE == EXECUT ==
4010 REM
4020 REM
4030 REM
4040 IF SCEL = 1 THEN GOSUB 13000 : REM == FNDREC ==
4050 REM 2 E' == FNDNMS ==
4060 REM 3 E' == FNDTWN ==
4070 REM 4 E' == FNDINT ==
4080 IF SCEL = 5 THEN GOSUB 17000 : REM ELENCO DEI RECORD
4090 IF SCEL = 6 THEN GOSUB 10000 : REM == AGGREC ==
4100 IF SCEL = 7 THEN GOSUB 14000 : REM == MODREC ==
4110 IF SCEL = 8 THEN GOSUB 15000 : REM == DELREC ==
4120 IF SCEL = 9 THEN GOSUB 11000 : REM == EXPROG ==
4130 REM
4140 RETURN
10000 REM SUBROUTINE == AGGREC ==
10010 PRINT CHR$(12) : REM PULIZIA DELLO SCHERMO
10020 INPUT "NOMINATIVO ";CAMNOM$(DIMENSIONE)
10025 IF LEN(CAMNOM$(DIMENSIONE)) = 0 THEN 10020
10030 INPUT "INDIRIZZO ";CAMVIA$(DIMENSIONE)
10040 INPUT "LOCALITA' ";CAMLOC$(DIMENSIONE)
10050 INPUT "NAZIONE ";CAMNAZ$(DIMENSIONE)
10060 INPUT "NUMERO DI TELEFONO ";CAMTEL$(DIMENSIONE)
10070 LET RMOD = 1 : LET SRTD = 0 : REM DATI MODIFICATI MA
      NON RIORDINATI
10080 LET CAMIND$(DIMENSIONE) = STR$(DIMENSIONE)
10090 LET TEST$ = ""
10100 GOSUB 10200 : REM == MODNOM ==
10110 LET SCEL = 0
10120 LET DIMENSIONE = DIMENSIONE + 1
10130 REM
10140 REM
10150 RETURN
10200 REM SUBROUTINE == MODNOM ==
10210 REM CONVERTE IL CONTENUTO DI CAMNOM$ IN MAIUSCOLE
10220 REM RIMUOVE CARATTERI ESTRANEI E MEMORIZZA:
10230 REM COGNOME+SPAZIO+NOME IN CAMMOD$
10240 REM
10250 LET N$ = CAMNOM$(DIMENSIONE)
10260 FOR L = 1 TO LEN(N$)
10270 LET TEMP$ = MID$(N$,L,1)
10280 LET T = ASC(TEMP$)
10290 IF T )= 97 THEN T = T - 32
10300 LET TEMP$ = CHR$(T)
10310 LET P$ = P$ + TEMP$
10320 NEXT L
10330 LET N$ = P$
10340 REM LOCALIZZA L'ULTIMO SPAZIO
10350 FOR L = 1 TO LEN(N$)
10360 IF MID$(N$,L,1) = " " THEN S = L
10370 NEXT L
10380 REM RIMUOVE CARATTERI ESTRANEI
10390 REM E SALVA NOME IN CNAM$
10400 FOR L = 1 TO S - 1
10410 IF ASC(MID$(N$,L,1)) ) 64 THEN CNAM$ = CNAM$+MID$(N$,L,1)
10420 NEXT L
10430 REM RIMUOVE CARATTERI ESTRANEI
10440 REM E SALVA CGNOME IN SNAM$
10450 FOR L = S + 1 TO LEN(N$)
10460 IF ASC(MID$(N$,L,1)) ) 64 THEN SNAM$ = SNAM$ + MID$(N$,L,1)
10470 NEXT L
10480 LET CAMMOD$(DIMENSIONE) = SNAM$ + " " + CNAM$
10490 LET P$ = "":LET N$ = "":LET SNAM$ = "": LET CNAM$ = ""
10500 RETURN
11000 REM SUBROUTINE == EXPROG ==
11010 REM RIORDINA E MEMORIZZA IL FILE
11020 REM SE SONO STATE APPORTATE MODIFICHE
11030 REM AD ALMENO UN RECORD (RMOD = 1)
11040 REM O SE OCCORRE UN RIORDINO (SRTD=0)
11050 REM RMOD = 0 E SRTD = 0 ASSIEME NON SONO ACCETTABILI
11060 REM
11070 IF RMOD = 0 AND SRTD = 1 THEN RETURN
11080 IF RMOD = 1 AND SRTD = 0 THEN GOSUB 11200 : REM == SRTREC ==
11090 GOSUB 12000 : REM == SAVREC ==
11100 RETURN
11200 REM SUBROUTINE == SRTREC ==
11210 REM ESEGUE UN RIORDINO ALFABETICO
11220 REM SUL CAMPO CAMMOD$, ED AGGIORNA
11230 REM CAMIND$
11240 REM
11250 LET S = 0
11260 FOR L = 1 TO DIMENSIONE - 2
11270 IF CAMMOD$(L) ) CAMMOD$(L+1) THEN GOSUB 11350
11280 NEXT L
11290 IF S = 1 THEN 11250
11300 REM
11310 REM
11320 LET SRTD = 1 : REM MODIFICA 'FLAG' DI RIORDINAMENTO
11330 REM
```

```
11340 RETURN
11350 REM SUBROUTINE == SWPREC == (SCAMBIO RECORD)
11360 LET TEMNOM$ = CAMNOM$(L)
11370 LET TEMMOD$ = CAMMOD$(L)
11380 LET TEMVIA$ = CAMVIA$(L)
11390 LET TEMLOC$ = CAMLOC$(L)
11400 LET TEMNAZ$ = CAMNAZ$(L)
11410 LET TEMTEL$ = CAMTEL$(L)
11420 REM
11430 LET CAMNOM$(L) = CAMNOM$(L+1)
11440 LET CAMMOD$(L) = CAMMOD$(L+1)
11450 LET CAMVIA$(L) = CAMVIA$(L+1)
11460 LET CAMLOC$(L) = CAMLOC$(L+1)
11470 LET CAMNAZ$(L) = CAMNAZ$(L+1)
11480 LET CAMTEL$(L) = CAMTEL$(L+1)
11490 LET CAMIND$(L) = STR$(L)
11500 REM
11510 LET CAMNOM$(L+1) = TEMNOM$
11520 LET CAMMOD$(L+1) = TEMMOD$
11530 LET CAMVIA$(L+1) = TEMVIA$
11540 LET CAMLOC$(L+1) = TEMLOC$
11550 LET CAMNAZ$(L+1) = TEMNAZ$
11560 LET CAMTEL$(L+1) = TEMTEL$
11570 LET CAMIND$(L+1) = STR$(L+1)
11580 LET S = 1
11590 REM
11600 RETURN
12000 REM SUBROUTINE == SAVREC ==
12010 REM
12020 REM
12030 OPEN "O",#1,"RUBR.DAT"
12040 REM
12050 FOR L = 1 TO DIMENSIONE - 1
12060 PRINT #1,CAMNOM$(L);",";CAMMOD$(L);",";CAMVIA$(L);",";
      CAMLOC$(L)
12070 PRINT #1,CAMNAZ$(L);",";CAMTEL$(L);",";CAMIND$(L)
12080 NEXT L
12090 REM
12100 REM
12110 REM
12120 REM
12130 CLOSE #1
12140 REM
12150 RETURN
12500 REM SUBROUTINE == FLSIZE ==
12510 IF CAMNOM$(L) = "" THEN LET L = 50
12520 IF CAMNOM$(L) = "" THEN RETURN
12530 LET DIMENSIONE = DIMENSIONE + 1
12540 REM
12550 REM
12560 RETURN
13000 REM SUBROUTINE == FNDREC ==
13010 PRINT CHR$(12) : REM PULIZIA SCHERMO
13020 REM
13030 IF SRTD = 0 THEN GOSUB 11200 : REM == SRTREC ==
13040 PRINT
13050 PRINT
13060 PRINT TAB(18); "RICERCA DI UN RECORD"
13070 PRINT TAB(18); "IN BASE AL NOME"
13080 PRINT
13090 PRINT TAB(18); "IMMETTERE IL NOMINATIVO"
13100 PRINT TAB(18); "(NEL FORMATO: NOME, COGNOME)"
13110 PRINT
13120 PRINT
13130 REM
13140 INPUT "NOMINATIVO ";CAMNOM$(DIMENSIONE)
13150 GOSUB 10200: REM SUBROUTINE == MODNOM==
13160 LET KEY$ = CAMMOD$(DIMENSIONE)
13170 REM
13180 REM
13190 REM
13200 REM
13210 REM
13220 LET FDO = 1
13230 LET CMA = DIMENSIONE - 1
13240 FOR L = 1 TO 1
13250 LET MED = INT((FDO + CMA)/2)
13260 IF CAMMOD$(MED) <> KEY$ THEN L = 0
13270 IF CAMMOD$(MED) < KEY$ THEN FDO = MED + 1
13280 IF CAMMOD$(MED) > KEY$ THEN CMA = MED - 1
13290 IF FDO > CMA THEN L = 1
13300 NEXT L
13310 REM
13320 IF FDO > CMA THEN LET CORR = 0
13330 IF FDO <= CMA THEN LET CORR = MED
13340 IF CORR = 0 THEN GOSUB 13700 : REM ==NOTREC==
13350 IF CORR = 0 THEN RETURN
13360 REM
13370 REM
13380 PRINT CHR$(12)
13390 PRINT
13400 PRINT TAB(18);"RECORD TROVATO"
13410 PRINT
13420 PRINT "NOMINATIVO:",CAMNOM$(CORR)
13430 PRINT "INDIRIZZO:",CAMVIA$(CORR)
13440 PRINT "CITTA`:",CAMLOC$(CORR)
13450 PRINT "NAZIONE:",CAMNAZ$(CORR)
13460 PRINT "N. TEL.:",CAMTEL$(CORR)
13470 PRINT
13480 PRINT TAB(8);"PREMERE QUALSIASI TASTO COPIA SU STAMPANTE"
13490 PRINT TAB(8);"OPPURE UNO SPAZIO PER CONTINUARE"
13500 FOR I = 1 TO 1
13510 LET A$ = INKEY$
13520 IF A$ = "" THEN I = 0
13530 NEXT I
13540 IF A$ <> " " THEN GOSUB 13600 : REM ==LSTCUR==
13550 RETURN
13600 REM  SUBROUTINE ==LSTCUR== (COPIA SU STAMPANTE)
13610 LPRINT
13620 LPRINT "NOMINATIVO:",CAMNOM$(CORR)
13630 LPRINT "INDIRIZZO:",CAMVIA$(CORR)
13640 LPRINT "CITTA`:",CAMLOC$(CORR)
13650 LPRINT "NAZIONE:",CAMNAZ$(CORR)
13660 LPRINT "N. TEL.:",CAMTEL$(CORR)
13670 LPRINT
13680 LPRINT
13690 RETURN
13700 REM SUBROUTINE ==NOTREC== (RECORD NON TROVATO)
13710 PRINT CHR$(12) : REM PULIZIA DELLO SCHERMO
13720 PRINT TAB(18); "RECORD NON TROVATO"
13730 PRINT TAB(4); "SOTTO LA FORMA: ";CAMNOM$(DIMENSIONE)
13740 PRINT
13750 PRINT TAB(4);"(PREMERE IL TASTO SPAZIO PER CONTINUARE)"
13760 FOR I = 1 TO 1
13770 IF INKEY$ <> " " THEN I = 0
13780 NEXT I
13790 RETURN
14000 REM SUBROUTINE ==MODREC== (MODIFICA RECORD)
14010 REM
14020 PRINT CHR$(12) : REM PULIZIA DELLO SCHERMO
14030 PRINT
14040 PRINT
14050 PRINT
14060 PRINT
14070 PRINT TAB(8);"PER MODIFICARE UN RECORD OCCORRE PRIMA
      INDIVIDUARLO"
14080 REM
14090 REM
14100 REM
14110 GOSUB 13030 : REM SUBROUTINE ==FNDREC==
14120 IF CORR = 0 THEN RETURN : REM RECORD NON TROVATO
14130 PRINT
14140 PRINT "MODIFICA AL NOMINATIVO ";
14150 GOSUB 16000 : REM IMMISSIONE RISPOSTA
14160 IF A$ <> CHR$(13) THEN GOTO 14220
14170 INPUT "NUOVO NOMINATIVO ";CAMNOM$(CORR)
14180 LET RMOD = 1 : LET SRTD = 0
14190 CAMNOM$(DIMENSIONE) = CAMNOM$(CORR)
14200 GOSUB 10200 : REM SUBROUTINE ==MODNOM==
14210 LET CAMMOD$(CORR) = CAMMOD$(DIMENSIONE)
14220 PRINT
14230 PRINT "MODIFICA ALLA VIA ";
14240 GOSUB 16000 : REM IMMISSIONE RISPOSTA
14250 IF A$ <> CHR$(13) THEN GOTO 14280
14260 LET RMOD = 1
14270 INPUT "NUOVA VIA ";CAMVIA$(CORR)
14280 PRINT
14290 PRINT "MODIFICA ALLA CITTA` ";
14300 GOSUB 16000 : REM IMMISSIONE RISPOSTA
14310 IF A$ <> CHR$(13) THEN GOTO 14340
14320 LET RMOD = 1
14330 INPUT "NUOVA CITTA` ";CAMLOC$(CORR)
14340 PRINT
14350 PRINT "MODIFICA ALLA NAZIONE ";
14360 GOSUB 16000 : REM IMMISSIONE RISPOSTA
14370 IF A$ <> CHR$(13) THEN GOTO 14400
14380 LET RMOD = 1
14390 INPUT "NUOVA NAZIONE ";CAMNAZ$(CORR)
14400 PRINT
14410 PRINT "MODIFICA AL N. TELEFONICO ";
14420 GOSUB 16000 : REM IMMISSIONE RISPOSTA
14430 IF A$ = CHR$(13) THEN RMOD = 1
14440 IF A$ = CHR$(13) THEN INPUT "NUOVO TELEFONO ";CAMTEL$(CORR)
14450 REM
14460 REM
14470 RETURN
15000 REM SUBROUTINE ==DELREC== (ANNULLAMENTO RECORD)
15010 REM
15020 PRINT CHR$(12): REM PULIZIA DELLO SCHERMO
15030 PRINT
15040 PRINT
15050 PRINT
15060 PRINT
15070 PRINT TAB(18); "PER CANCELLARE UN RECORD OCCORRE"
15080 PRINT TAB(18); "PRIMA INDIVIDUARLO"
15090 REM
15100 REM
15110 REM
15120 GOSUB 13030 : REM SUBROUTINE ==FNDREC==
15130 IF CORR = 0 THEN RETURN : REM RECORD NON TROVATO
15140 PRINT
15150 PRINT TAB(3);"E` QUESTO IL RECORD DA CANCELLARE ?"
15160 PRINT TAB(5); "ATTENZIONE: VALE LA PRIMA RISPOSTA!"
15170 PRINT
15180 PRINT TAB(18);"PREMERE RETURN PER CANCELLARE"
15190 PRINT TAB(18);"OPPURE SPAZIO PER CONTINUARE"
15200 FOR I = 1 TO 1
15210 LET A$ = INKEY$
15220 IF A$ <> CHR$(13) AND A$ <> " " THEN I = 0
15230 NEXT I
15240 IF A$ = " " THEN RETURN
15250 FOR L = CORR TO DIMENSIONE - 2
15260 LET CAMNOM$(L) = CAMNOM$(L + 1)
15270 LET CAMMOD$(L) = CAMMOD$(L + 1)
15280 LET CAMVIA$(L) = CAMVIA$(L + 1)
15290 LET CAMLOC$(L) = CAMLOC$(L + 1)
15300 LET CAMNAZ$(L) = CAMNAZ$(L + 1)
15310 LET CAMTEL$(L) = CAMTEL$(L + 1)
15320 LET CAMIND$(L) = CAMIND$(L + 1)
15330 NEXT L
15340 LET RMOD = 1
15350 LET DIMENSIONE = DIMENSIONE - 1
15360 REM
15370 REM
15380 REM
15390 RETURN
16000 REM SUBROUTINE PER IMMISIONE RISPOSTA
16010 PRINT "(RETURN = MODIFICA   SPAZIO = NESSUNA MODIFICA)";
16020 FOR I = 1 TO 1
16030 LET A$ = INKEY$
16040 IF A$ <> CHR$(13) AND A$ <> " " THEN I = 0
16050 NEXT I
16060 PRINT
16070 RETURN
17000 REM SUBROUTINE ==LSTREC==
17010 FOR CORR = 1 TO DIMENSIONE - 1
17020 GOSUB 13380
17030 NEXT CORR
17040 CORR = 0
17050 RETURN
```

Grace Hopper

Questa donna ha lavorato a lungo per lo sviluppo di linguaggi ad alto livello

L'informatica viene spesso considerata come una materia quasi esclusivamente riservata agli uomini, ma sono sempre più frequenti le donne impegnate alacremente nello sviluppo dei computer e relative applicazioni. Un personaggio pionieristico in questo settore è stato Grace Hopper, che ha enormemente contribuito allo sviluppo del software: oltre ad aver creato il primo "compilatore", è stata una delle creatrici del linguaggio COBOL. Da molti è ricordata, scherzosamente, per esser stata la prima a battezzare gli errori dei programmi col termine "bug" (vedere a pag. 433).

Dopo un corso post-universitario a Yale



per concessione della SPERRY LTD

(USA), Grace Hopper tornò, come membro della facoltà di matematica, presso l'università dove aveva iniziato gli studi, a Vassar. Qui rimase fino all'età di 39 anni, quando venne chiamata a far parte del servizio di difesa militare, lavorando al "Naval Ordinance Computation Project". Nel 1945 fu inviata all'università di Harvard, affinché collaborasse, assieme al fisico Howard Aiken, alla costruzione di un computer.

Aiken aveva in passato cercato ed ottenuto un finanziamento dalla IBM per lo sviluppo di un simile apparecchio, presentando un prototipo meccanico del proprio progetto. Il computer venne costruito e battezzato col nome HARVARD MARK II.

A quell'epoca, i calcolatori venivano programmati mediante allacciamenti elettrici, ottenuti con un numero incredibile di cavi e, una mattina della calda estate del '45, Grace Hopper si trovò letteralmente imbrigliata nel bel mezzo di un urgentissimo calcolo balistico, necessario agli organi di guerra.

Un giorno, un improvviso quanto inspiegabile guasto del computer bloccò ogni attività. Grace Hopper, allora insignita del grado di capitano, riuscì a trovare l'inconveniente: un insetto, entrato da una delle finestre del laboratorio, si era cacciato tra i delicati meccanismi a relé del computer. Individuato quello che lei chiamò il "bug", esso venne rimosso e l'apparecchio tornò a funzionare regolarmente.

L'insetto è oggi conservato nel Museo Navale della Virginia, tra le pagine del manuale dell'Harvard Mark II. Accanto, una data: il 9 settembre 1945.

Durante lo stesso anno, veniva messo a punto da due ingegneri, John Mauchly e Presper Eckert, un altro computer, l'ENIAC (vedere a pag. 46). Terminata la guerra, i due ingegneri si dedicarono alla produzione commerciale di quel computer, e chiesero alla Hopper di unirsi al progetto. Il suo maggior contributo alla nascita dell'UNIVAC, fu la creazione del software per questo nuovo computer.

Fu proprio lavorando a questo progetto che a Grace Hopper venne l'idea di sviluppare delle "scorciatoie" nella programmazione, facendo sì che fosse lo stesso computer a generare parte dei sottoprogrammi di più frequente uso: nacque così il primo linguaggio dotato di un proprio compilatore, al quale venne assegnato il nome "A-O". Allo stupore ed all'incredulità iniziali, tra i professionisti dell'informatica del tempo, abituati a considerare il computer soltanto come strumento di puro calcolo, si sostituì ben presto l'entusiasmo.

Nel maggio del 1959, il capitano Grace Hopper fu invitato dal Pentagono a collaborare ad un ambizioso progetto: lo sviluppo di un linguaggio standard, adatto ad applicazioni commerciali.

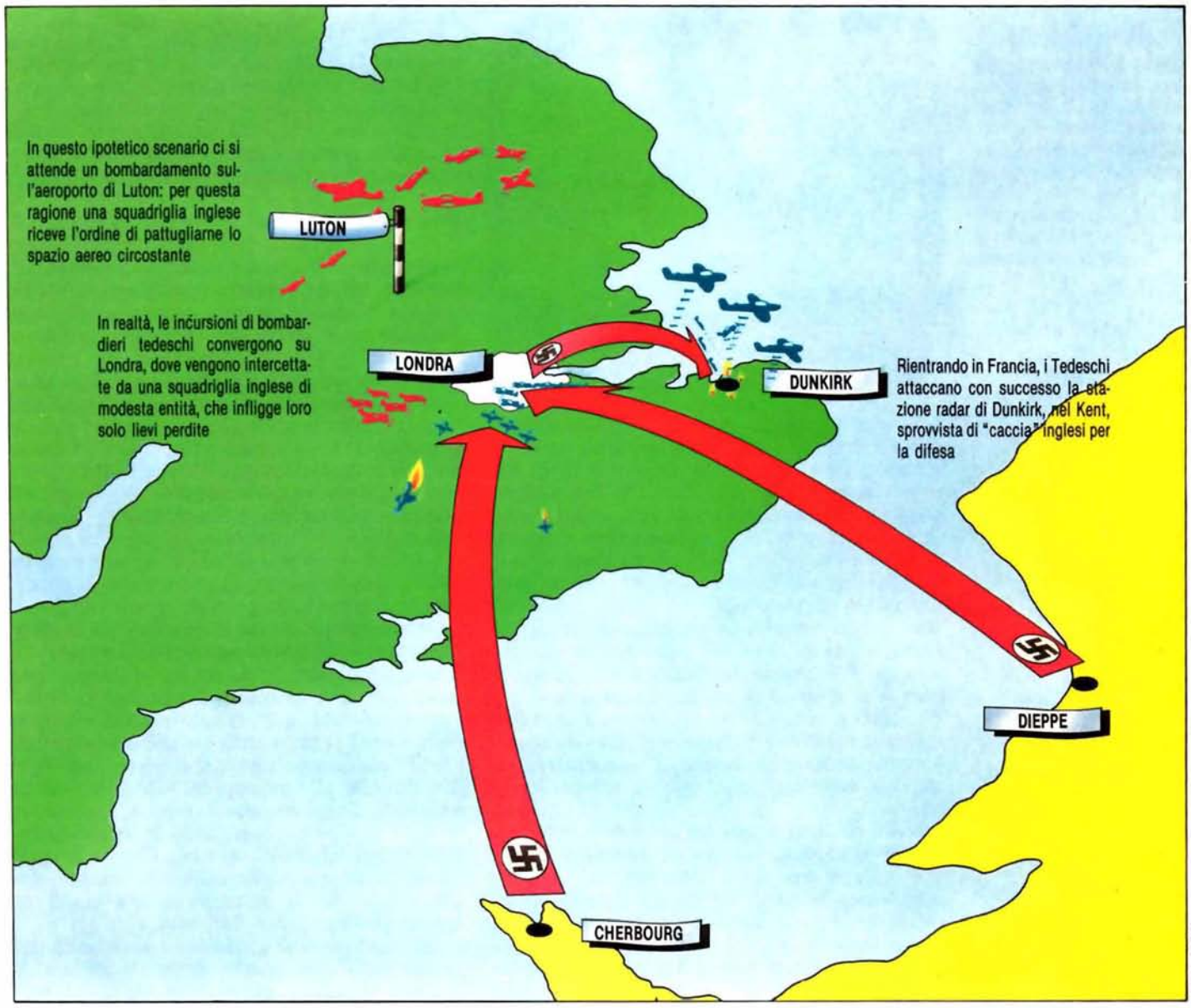
In meno di un anno, la commissione approntò la prima versione del COBOL (COmmon Business Orientated Language, un linguaggio orientato alla generica elaborazione dei dati commerciali), grazie anche alla notevole esperienza di Grace Hopper. Da allora, sono state prodotte nuove e più complete versioni del COBOL, un linguaggio ancora oggi molto diffuso ed apprezzato.

Il COBOL

Il Cobol è stato il primo linguaggio sviluppato specificatamente per facilitare la programmazione nel settore commerciale. La sua struttura incoraggia la stesura di sottoprogrammi d'impiego generico e le istruzioni sono formate da normali parole della lingua inglese, riducendo al minimo la necessità di redigere una documentazione d'accompagnamento ai programmi.

Un programma COBOL si suddivide in quattro sezioni. La prima, la "Identification Division", contiene elementi descrittivi generali (nome del programma, archivi impiegati, ecc.). Poiché il COBOL è stato appositamente studiato per esser utilizzato anche su computer di marche diverse, le caratteristiche specifiche alla particolare macchina vengono riportate nella "Environment Division".

Le descrizioni dei dati coinvolti nei processi di elaborazione sono invece raggruppate nella "Data Division". Infine, le procedure (ossia l'effettiva sequenza delle operazioni necessarie all'elaborazione dei dati) vengono elencate nell'ultima sezione, la "Procedure Division"



KEVIN JONES

Guerra e pace

Alcuni giochi mettono alla prova le doti tattiche e strategiche del giocatore

Oggigiorno le stesse Forze Armate attribuiscono grande importanza ai "Giochi di guerra" (War Games), spesso adoperati per saggiare le reazioni degli ufficiali in situazioni di minaccia simulate, oppure per verificare le risposte strategiche pianificate per difendersi da ipotetici attacchi. Per questi sofisticati giochi sono stati sviluppati sistemi software ed hardware di notevole complessità, capaci di simulare tutti gli aspetti, oggi noti, di un



IAN McKINNELL

La Battaglia d'Inghilterra

"Fighter Command", un gioco prodotto dalla Strategic Simulation Inc. per l'Apple II, è un tipico esempio dei sofisticati War Games per microcomputer. Prima che il gioco abbia inizio, il giocatore deve scegliere tra una vasta gamma di opzioni, che comprendono anche i modelli d'aereo di cui si vuol disporre e le condizioni atmosferiche. Il gioco si sviluppa usando simboli in movimento sulla cartina ed informazioni supplementari visualizzate in forma di testi scritti. La documentazione d'accompagnamento al programma comprende una cartina delle zone d'operazione e delle piccole figure in cartone, che permettono una più chiara visione del gioco

Comando d'Unità Corazzate Tattiche

Questo gioco, prodotto dalla Avalon Hill per i computer Atari, Apple, IBM PC e Commodore 64, simula un conflitto tra unità corazzate durante la Seconda Guerra Mondiale. Può essere giocato da uno o due giocatori che scelgono tra cinque differenti scenari e muovono unità corazzate inglesi, americane, sovietiche e tedesche



IAN McKINNELL

potenziale conflitto (quali il dispiegamento iniziale delle forze alleate e nemiche, i meccanismi di rifornimento, la disponibilità di approvvigionamenti e via dicendo). Tali sistemi sono progettati in modo da tener conto anche di eventuali condizioni atmosferiche avverse, di mutamenti tattici del nemico, degli effetti di sabotaggi compiuti da reparti infiltrati e di numerose altre variabili che possano influenzare positivamente lo sviluppo di un'operazione militare. Uno dei principali scopi del sistema di difesa radar NORAD (descritto nel film *War Games*), è quello di accertare, aggiornare e valutare costantemente le informazioni relative alle forze militari sovietiche e statunitensi e permettere l'elaborazione di risposte adatte ad ogni nuovo sviluppo strategico.

I giochi strategici tradizionali, per raggiungere un sufficiente livello di complessità, si svolgono su grandi tabelloni, su cui si annota manualmente ogni particolare mossa, generalmente basata sul lancio di dadi e sulla consultazione di voluminosi manuali e regolamenti. Ciò restringe l'interesse da essi suscitato ad un numero relativamente piccolo di amatori. Tuttavia, l'enorme diffusione di personal e home computer ha portato alla creazione di programmi per giochi strategici, studiati appositamente, nei quali lo svolgimento, eliminate le noiose fasi "passive" (l'annotazione delle mosse e l'incessante consultazione di manuali, per esempio), è reso molto più semplice ed avvincente.

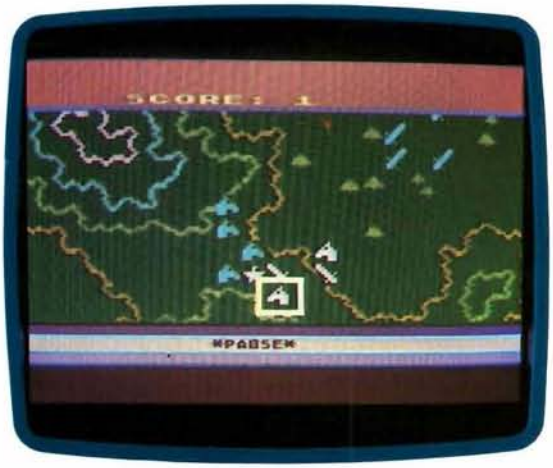
Anche se, concettualmente, i principi del gioco rimangono i soliti, esiste un'immensa gamma di "variazioni sul tema": da battaglie combattute nell'Antica Grecia, a teorici conflitti che, un giorno o l'altro, potrebbero divampare tra le odierne superpotenze; battaglie aeree o navali, guerre spaziali o, addirittura, conflitti ambientati nel mondo mitologico.

Una particolarità dei giochi storici (nei quali le situazioni iniziali vengono scrupolosamente ricreate), è che il giocatore può scoprire, ad esempio, gli errori tattici commessi da Napoleone a Waterloo, oppure riuscire a bloccare l'avanzata di Hitler sul territorio sovietico nel 1941. Le guerre ambientate nel cosmo, invece, danno maggior spazio alla fantasia: non solo è possibile guidare flotte di astronavi attraverso sconfinate galassie, ma anche specificare il modello e le caratteristiche delle astronavi da usare. Infatti, desiderando una maggiore velocità, occorre rinunciare ad una parte dei sistemi difensivi (che appesantiscono il veicolo), mentre la presenza di potenti schermi protettivi influisce negativamente sulle scorte di energia. È necessario, in poche parole, scegliere quei compromessi che più si adattano ad una particolare strategia di combattimento. Diversamente dai loro equivalenti convenzionali, i giochi di strategia per computer non richiedono particolari conoscenze o abilità (magari solo qualche suggerimento per i principianti) e si possono suddividere in tre gruppi principali: i giochi introduttivi, quelli di media difficoltà e quelli avanzati. Come in tutte le cose, è necessario iniziare a giocare con quelli di livello introduttivo per familiarizzare con i concetti basilari e con il genere di simulazioni strategiche offerte, prima di cimentarsi con quelli di maggiore impegno e difficoltà.

Lo svolgimento del gioco varia a seconda del programma impiegato e della situazione simulata, ma, generalmente, le battaglie vengono rappresentate su carte geografiche di grandi dimensioni, riprodotte sullo schermo del computer. Qualora il campo delle operazioni sia più vasto di quanto possa essere visualizzato sullo schermo, allora il video viene usato come una finestra che viene spostata, mediante un joystick, da un punto all'altro dello scenario. Nelle simulazioni sto-

Il Legionario

Questa è una simulazione di una guerra tra legionari di Cesare (il giocatore) e barbari (il computer Apple o Atari) e si svolge in "tempo reale". Fanteria, cavalleria ed altri reparti sono rappresentati da simboli che possono essere selezionati e mossi da un cursore (un quadratino bianco) controllato da un joystick. Questo gioco è stato prodotto dalla Avalon Hill





per concessione della SOFT

IAN McKINNELL

riche, i progettisti del gioco tentano sempre di riprodurre il più fedelmente possibile i terreni sui quali ha avuto luogo l'evento reale. Per esempio, nel gioco *Computer Bismarck* (prodotto dalla Strategic Simulation Inc.) l'azione si svolge principalmente nel Nord Atlantico, per cui non è stato troppo difficile progettare e riprodurre graficamente il campo di operazioni. Al contrario, per il gioco *La battaglia di Normandia*, sempre della SSI, il lavoro dei progettisti è stato alquanto più difficoltoso: non solo era necessario ricostruire in modo esatto la topografia del terreno, ma occorreva anche riprodurre con precisione particolari importanti come la linea costiera, le spiagge, le città, i villaggi ed i corsi dei fiumi. Nei giochi non legati a particolari situazioni storiche, i progettisti si prefiggono l'obiettivo di offrire scenari movimentati, che permettano ai giocatori di sfruttare adeguatamente le forze belliche di cui dispongono, prestando attenzione, comunque, ad inserire difficoltà equamente ripartite, onde evitare palesi situazioni di vantaggio per uno dei giocatori rispetto agli altri.

La rappresentazione grafica delle zone di combattimento è suddivisa in settori mediante un reticolo e somiglia, perciò, ad una scacchiera. In base alle particolari caratteristiche territoriali, a ciascun settore viene associato un valore, che rappresenta il grado di difficoltà di quella particolare area di combattimento. Maggiore è il grado di difficoltà di una specifica area, minori divengono, per le forze militari impegnate, le possibilità di compiere rapidi spostamenti. Quando la capacità di movimento di un reparto o battaglione è uguale a zero (oppure minore rispetto al valore dell'area che ci si propone di occupare), allora il giocatore è costretto a saltare un turno di gioco.

Infatti, per simulare il trascorrere del tempo, in genere, il gioco è condotto in successive fasi o turni: il giocatore, per vincere, deve conquistare un certo numero di obiettivi nel "tempo" a disposizione (ossia entro un certo numero di mosse). Per prima cosa, il giocatore deve valutare le proprie possibilità di gioco e, conseguentemente, determinare gli obiettivi strategici ritenuti di primaria importanza. Tuttavia, in molti casi non è necessario, anche se possibile, raggiungere tutti gli obiettivi prefissati. In scenari di questo tipo il ruolo dell'avversario è quello di contrastare l'attaccante nei suoi tentativi di conquista. Il difensore, d'altro canto, deve decidere su quali postazioni concentrare le proprie forze, in quanto non è possibile, ovviamente, assicurarsi il simultaneo dominio difensivo di ciascuna di esse. Inoltre, è necessario stabilire per quanto tempo difendere una roccaforte e se correre o meno il rischio di lanciare un contrattacco per riconquistare una postazione ormai perduta o per distruggere i preparativi fatti dall'avversario in previsione di un nuovo attacco.

Il programma, mediante rappresentazioni grafiche o testuali, comunica ai giocatori le informazioni relative alle rispettive forze (dislocamento, consistenza, efficienza bellica, possibilità di movimento, ecc.).

Ciascun giocatore, a turno, immette gli opportuni comandi (mediante un joystick o la tastiera) per lo svolgimento del gioco. In certi casi, comunque, è lo stesso computer che, sempre a turno, gestisce le varie fasi di azioni più complesse. Generalmente, ad ogni reparto combattivo è associato un numero identificativo, che viene impiegato per impartire comandi di movimento, anch'essi numerici. Ad esempio, il numero 1 provocherà uno spostamento del reparto verso nord, il numero 2 verso nord-est e così via per tutti i punti cardinali. Un numero sempre crescente di questi giochi fa uso di joystick o di sfere-guida: in tal caso ciascun reparto viene innanzitutto identificato e quindi spostato nella direzione voluta. Per porre fine al movimento di un dato reparto, viene spesso usato il comando FINISH o F. Alcuni giochi danno però la possibilità al giocatore di selezionare nuovamente il reparto e di muoverlo ancora, a meno che le sue possibilità di movimento non siano esaurite. Completato l'intero movimento, il giocatore indicherà al computer la mossa effettuata con il comando EXECUTE o E. Il computer darà allora inizio alla fase di combattimento.

Durante questa fase il programma indica i reparti alleati abilitati a dar battaglia al nemico e fornisce informazioni sulle truppe che compongono i reparti coinvolti nello scontro. Sulla base di questi dati il giocatore può decidere se accettare o rifiutare la proposta di scontro che gli viene offerta. Dopo che il combattimento è terminato e ne sono stati calcolati e visualizzati gli effetti, il secondo giocatore inizia il proprio turno di gioco.

Per molti appassionati, il fascino dei giochi di strategia deriva dal fatto che non esiste una soluzione unica ai problemi che il gioco propone: il maggior divertimento consiste proprio nel riuscire a superare le difficoltà fisiche e logistiche imposte dallo scenario di gioco, nel quale i fattori raramente si ripetono: si combatte, quindi, una battaglia basata sull'intelligenza, sfruttando al meglio gli errori tattici dell'avversario.



Fronte Orientale

In questo gioco, il giocatore si trova a capo dell'esercito tedesco, durante il tentativo di conquistare Mosca nel 1941, mentre il computer guida le forze di difesa sovietiche. Progettato da Chris Crawford e commercializzato dall'Atari, il gioco presenta radicali innovazioni grafiche che lo rendono molto diverso da altri giochi del genere. Una delle caratteristiche visive più interessanti è il fatto che lo scenario cambia al mutare delle stagioni: in autunno le foreste perdono le foglie, in inverno i fiumi gelano ed il terreno si copre di neve. In questo gioco raggiungere Mosca è un obiettivo estremamente difficile

IAN McKINNELL

Gli attrezzi del mestiere

I "Tool Kit" permettono di ampliare le prestazioni del BASIC e facilitano la messa a punto dei programmi

I primi piccoli computer, come l'Apple II ed il PET della Commodore, offrivano prestazioni limitate ed il BASIC, di cui erano dotati, disponeva solo di comandi e funzioni piuttosto elementari per l'elaborazione di numeri e parole. Conseguentemente, è sorta la necessità di disporre di programmi di utilità o di software di supporto, per migliorare la programmazione in BASIC. Questi programmi servono per aggiungere nuovi comandi al BASIC esistente, oppure per facilitare la ricerca e la correzione degli errori.

È vero che le case produttrici di computer offrono talvolta delle nuove versioni di BASIC, arricchite di nuove funzioni, per una migliore gestione degli effetti grafici e sonori sui loro apparecchi, ma spesso anche queste nuove versioni comportano un largo impiego di complesse operazioni, basate su delle PEEK e delle POKE per ottenere gli effetti desiderati. Pertanto, i programmatori sono ancora costretti, in molti casi, a scriversi tutta una serie di sottoprogrammi per poter sfruttare al massimo le capacità delle macchine.

Compaiono, quindi, sempre nuovi programmi di utilità ed estensioni per il BASIC che consentono un più facile accesso alle funzioni già presenti.

I programmi di questo genere possono risiedere nella RAM, nella ROM interna o in cartucce ROM. Un'estensione ROM è preferibile ad una da caricare nella RAM, perché protetta da cancellazioni accidentali ed, inoltre, non occupa memoria che può esser utile al programma applicativo. In linea di massima, un programma scritto con l'aiuto di un pacchetto di supporto può funzionare soltanto su un computer sul quale sia montato il medesimo equipaggiamento. Esistono, comunque, pacchetti che permettono di produrre programmi "universali", in grado, cioè, di funzionare su qualsiasi tipo di computer. Su questo principio si basano la maggior parte di compilatori per sprite e grafica ed alcuni dei compilatori per gli effetti audio.

Speciali comandi grafici (come PAINT, DRAW, PLOT, CIRCLE, ecc.) e comandi di suono (come SOUND, PLAY, ENVELOPE, ecc.) o parole di comando che descrivono un effetto sonoro (come BANG o ZAP) sono ottimi strumenti per farsi un'idea più chiara delle estensioni BASIC.

Altri utili strumenti sono i comandi per la programmazione strutturata, come REPEAT... UNTIL oppure IF...THEN...ELSE. Istruzioni di questo tipo permettono al programmatore di scrivere programmi che si sviluppano secondo una precisa logica ed evitano le difficoltà d'interpretazione nei listati di programmi ove vi sia un uso ossessivo dell'istruzione GOTO.

Pacchetti di supporto software

Vengono oggi venduti alcuni pacchetti software di supporto ed estensioni BASIC (chiamati "Tool Kit", letteralmente "borsa degli attrezzi"), utilizzabili anche su alcuni dei più diffusi home computer. Tra di essi, i programmi che sembrano ottenere maggior successo sono quelli per la creazione e la gestione della grafica a sprite

I "TOOL KIT"	
SUPER TOOL KIT	prodotto dalla Nectarine, è utilizzabile sullo Spectrum da 16 e da 48 Kbyte
SPECTRUM EXTENDED BASIC	prodotto dalla CP Software per lo Spectrum a 48 Kbyte
SPECTRUM KEYDEFINE	prodotto dalla Scientist Software per lo Spectrum a 48 Kbyte
PROGRAMMER'S AID	prodotto dalla Commodore per il VIC-20
BUTI	prodotto dalla Audiogenic per il VIC-20
TOOL BOX	prodotto dalla BBC Software per i Modelli A e B del Micro BBC
SPRITE MAGIC	prodotto dalla Merlin Micro System per il Dragon 32
SPRITE GRAPHICS	prodotto dalla B Side Software per lo Spectrum a 48 Kbyte
SPRITE MASTER	prodotto dalla Micro Dealer UK per il Modello B della BBC

IL "SIMON BASIC"

Oggi la più completa estensione per il linguaggio BASIC è il "Simon BASIC", disponibile per il Commodore 64 sotto forma di cartuccia ROM. Il BASIC standard della Commodore, di cui il "64" è dotato, è piuttosto antiquato, essendo provvisto di un esiguo numero di comandi speciali e di comandi adatti alla programmazione strutturata. Sebbene il computer sia dotato di avanzate caratteristiche hardware, come un completo sintetizzatore di suono, una grafica ad alta risoluzione ed una grafica a sprite, il controllo base di queste funzioni avviene tramite i comandi PEEK e POKE. Il Simon BASIC assicura al Commodore 64 una notevole estensione che comprende le seguenti facilitazioni:

1) Una gamma completa di sussidi alla programmazione (funzioni per la ricerca de-

FERRI DEL MESTIERE

Questi comandi rappresentano solo alcune facilitazioni che ogni buona estensione per il BASIC dovrebbe possedere



DISK: vari comandi per la gestione degli archivi su dischetti: per "formattare" nuovi dischetti, per copiare e cancellare singoli file e per creare copie integrali degli archivi di un dischetto

DISK



DRAW: oltre ai comandi necessari per tracciare linee rette in alta risoluzione, è utile la presenza di un comando che permetta di tirare linee da un punto ad un altro dello schermo, facilitando la creazione delle immagini

DRAW



PAINT: questo comando consente di colorare una particolare zona dello schermo. Su molte macchine è necessario posizionare prima il cursore nel mezzo della figura da colorare, quindi il computer comincerà a "pennellare" il colore partendo dal centro fino ad incontrare le linee di contorno

PAINT



CIRCLE: la funzione di questo comando è abbastanza semplice: tracciare automaticamente un cerchio. Tuttavia, in alcuni casi, esso permette anche di tracciare archi di circonferenza ed ellissi

CIRCLE



MUSIC: le funzioni di suono variano molto da macchina a macchina, ma il comando MUSIC di solito permette di suonare una sequenza di note precedentemente definita da una serie d'istruzioni

MUSIC



DIR: la "Directory" di un disco è una lista che mostra i nomi, la tipologia e le dimensioni di tutti i file presenti sul dischetto. Il comando DIR permette di visualizzare questa lista sullo schermo, senza cancellare il programma che in quel momento si trova nella RAM

DIR



JOY: su molti computer, per abilitare l'uso di joystick, occorre una serie di complesse operazioni, che implicano molteplici PEEK e POKE. I comandi JOYstick provvedono, automaticamente, ad assegnare a delle variabili BASIC i valori corrispondenti agli spostamenti

JOY

KEVIN JONES

gli errori e sistemi di sicurezza contro la produzione di copie non autorizzate di un programma).

2) Comandi addizionali per l'elaborazione delle stringhe e la redazione di testi.
3) Comandi addizionali per operatori matematici e di conversione numerica.
4) Comandi semplificati per la gestione degli archivi su dischetti.
5) Comandi per la grafica ad alta risoluzione, che permettono di fondere assieme testi e grafica di vario genere (spesso comprendono anche la possibilità di colorare in "tratteggio").
6) Comandi di visualizzazione e per la grafica a bassa risoluzione, che permettono di duplicare particolari aree di memoria e di manipolare con semplicità l'area video. Questi comandi permettono anche di salvare le immagini video su disco, registratore o stampante.
7) Comandi per una più facile creazione e manipolazione degli sprite.
8) Nuove istruzioni adatte alla programmazione strutturata, quali: PROC, CALL ed EXEC; cicli e routine di verifica di condizione, quali: REPEAT...UNTIL, LOOP...EXIT, IF...END LOOP ed IF...THEN...ELSE, che consentono di evitare le GOTO e le GOSUB.
9) Programmi di utilità per la creazione di suoni, che permettono di fare uso dell'intera gamma audio del generatore interno al computer.
10) Semplici comandi per l'impiego di penne luminose, joystick e "paddle".

È piuttosto raro che un'estensione includa una serie così completa di routine addizionali. La maggior parte dei pacchetti software prevede comandi e programmi di utilità limitati ad una specifica area della programmazione. Ad esempio, la cartuccia Super Expander, prodotta dalla Commodore per il VIC-20, possiede solo una semplice serie di comandi per la grafica ad alta risoluzione e per musica. Le estensioni di maggior successo, ovviamente, sono quelle che offrono anche validi strumenti di ausilio per la stesura di programmi: di solito dispongono di comandi a tasto unico e di svariate routine automatiche che semplificano in modo diretto la numerazione delle linee, la redazione del listato e la ricerca degli errori.

Difficilmente si trovano routine in grado di accrescere in modo significativo le capacità di un home computer, tuttavia i pacchetti software oggi prodotti si muovono in questa direzione. Ad esempio, i molti vantaggi che la grafica a sprite mette a disposizione, nella creazione di giochi di rapido movimento stile "sala giochi", hanno spinto alcune intraprendenti case di software a scrivere programmi di utilità per la creazione di sprite, da utilizzare su quei computer che nel loro standard non dispongono di queste caratteristiche.

I programmi di utilità BASIC, i pacchetti software di supporto e le estensioni rappresentano solo una piccola parte dei miglioramenti e degli accessori oggi disponibili sul mercato.

Il riconoscimento vocale

Comandare i computer con la sola voce non è più soltanto un sogno fantascientifico

Sebbene la diffusione dei computer, a tutti i livelli, sia già oggi enorme, questi strumenti raggiungeranno la loro massima popolarità soltanto quando sarà abbattuta la barriera che impedisce loro di comprendere il linguaggio umano. Infatti, l'interfaccia più comune, per comunicare col computer, è la tastiera (anche se oggi è possibile sostituirla, parzialmente, con "topi" o joystick) e ciò costringe ad usare dei linguaggi cosiddetti "artificiali": comandi quali CLS, DIRECTORY, RUN, LOAD e SAVE, possono essere significativi per il sistema operativo, ma non vengono certo adoperati nel linguaggio comune.

Fino a poco tempo fa, i progettisti di computer e di linguaggi di programmazione non si erano mai posti il problema di "umanizzare" il colloquio con gli elaboratori: qualsiasi tentativo in tal senso era ostacolato dall'enormità di memoria occorrente, e dal costo proibitivo che simili apparecchiature avrebbero avuto. La soluzione più rapida e meno costosa era senza dubbio far sì che fosse l'uomo ad imparare un nuovo linguaggio e non, viceversa, che fosse il computer ad imparare il linguaggio umano.

Ultimamente, tuttavia, il sensibile calo nel costo delle memorie, la maggiore velocità delle unità centrali, assieme a tecniche di programmazione molto più sofisticate di quelle usate in passato, fanno intravedere la possibilità di realizzare apparecchi capaci d'interpretare la voce parlata. I potenziali impieghi di simili computer sono facilmente intuibili ed i primi a beneficiarne sarebbero, tanto per fare un esempio concreto, i portatori di handicap fisici, ai quali è oggi precluso, nella maggior parte dei casi, l'utilizzazione dei computer.

Teoricamente, un processo di riconoscimento vocale è composto da 4 fasi: l'immissione di suoni, sotto forma analogica; la conversione di questi in segnali digitali, comprensibili dal computer; il confronto con modelli precedentemente memorizzati; l'esecuzione delle operazioni associate al comando riconosciuto.

A tutta prima, la realizzazione di un simile sistema, soprattutto considerando gli alti livelli raggiunti dall'elettronica, non sembrerebbe dover creare grandi problemi. Purtroppo, però, occorre tener presente che, anche in un solo linguaggio umano (l'inglese, per esempio), le parole sono costituite da combinazioni di suoni notevolmente complesse.

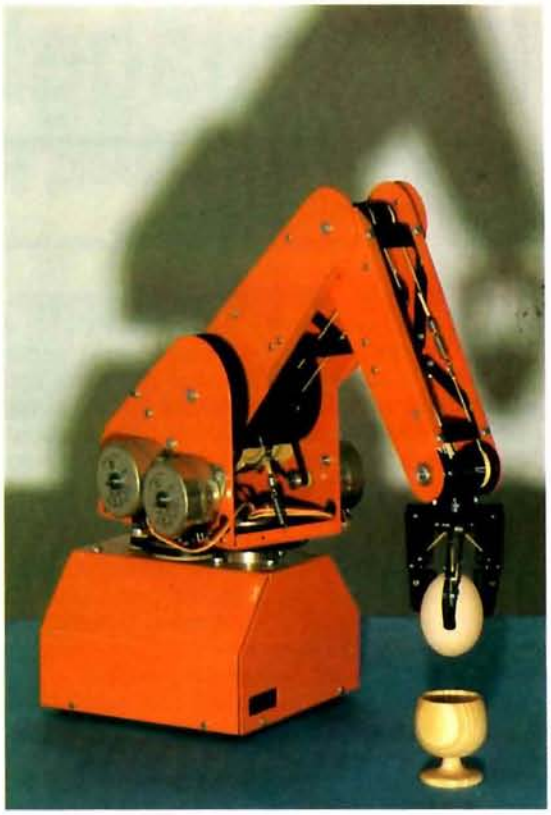
La fantastica ipotesi di un istantaneo e completo riconoscimento delle parole (capacità questa di cui era dotato il computer HAL nel film *2001 Odissea nello spazio*), resterà forse per molti anni ancora una meta difficile da raggiungere. Macchine per scrivere ad input vocale o computer in grado di comprendere la voce sono progetti costosissimi, per il momento, proprio per le difficoltà intrinseche nello sviluppo di sistemi di riconoscimento della voce: una parola può avere lo stesso suono, ma diversi significati a seconda del contesto della frase in cui è inserita. La potenza di elaborazione necessaria per abbattere questo ostacolo non è assolutamente ottenibile a costi ragionevoli. I ricercatori hanno sviluppato sistemi che si avvicinano allo standard richiesto, ma si è visto che, al crescere del numero di voci "riconosciute" dal computer (si ricordi che la voce umana ha caratteristiche che variano da individuo a individuo), corrisponde una sensibile diminuzione del numero di parole "comprese" dall'apparecchio. Tipicamente, un sistema di riconoscimento multivocale permette di riconoscere, simultaneamente, solo 20 o 30 parole alla volta, con probabilità di successo intorno all'85-90%.

Gli usi potenziali di sistemi di riconoscimento della voce sono considerevoli: in Germania, ad esempio, le poste ne fanno già uso per la cernita e distribuzione della posta, ma esistono anche applicazioni in campo aerospaziale, sia militare che civile. Infatti, i piloti di oggi non hanno, letteralmente, mani e piedi a sufficienza per controllare efficientemente tutta la complessa strumentazione di bordo: in queste condizioni, un apparecchio capace di interpretare comandi vocali diviene addirittura indispensabile. Sebbene, come abbiamo detto, il numero di parole riconosciute simultaneamente raggiunga a mala pena le 20 unità, ciò non significa che i sistemi finora realizzati siano limitati nel funzionamento. Infatti, l'utente può selezionare un comando da un menu di 20 parole, ma, una volta riconosciuto dalla macchina, esso produce la comparsa di un secondo menu da cui è possibile scegliere nuovamente. Il computer esegue il lavoro richiesto soltanto quando viene riconosciuta l'intera sequenza di comando. Nel caso di un ufficio di smistamento postale, il primo livello di scelta potrà avere come parametro il nome della nazione, indicato sulla busta; eseguita correttamente questa prima selezione, la scelta seguente potrà avere come parametro la provincia, poi la città o il paese, ecc.

Giunti al livello più basso, la busta avrà ormai imboccato la giusta direzione.

ANALISI DELLA VOCE

Il riconoscimento vocale viene solitamente realizzato secondo due criteri. Il modo più sbrigativo consiste nel semplice immagazzinamento delle parole, mediante un convertitore analogico/digitale. In un secondo momento, il computer esegue tutte le analisi necessarie (fonetiche, strutturali, ecc.).
Sfortunatamente, questo metodo presenta alcuni inconvenienti, il principale dei quali è la gran quantità di tempo necessario per le analisi. I sistemi che adottano questa procedura possono impiegare anche due o tre secondi per riconoscere l'input. Ma, affinché il riconoscimento della voce risulti di effettiva utilità, il computer deve essere in grado di capire le parole molto rapidamente (perlomeno quanto l'uomo), mentre le procedure basate sull'analisi numerica selettiva raramente hanno una risposta così immediata.
Il secondo metodo prevede, invece, una fase di pre-elaborazione. Il segnale parlato non viene analizzato matematicamente e la maggior parte del lavoro viene eseguita mediante normali procedure elettroniche. Nel computer vengono immesse informazioni concernenti le caratteristiche del linguaggio parlato: frequenza, tono, timbro, energia, ecc. Le frequenze si possono misurare filtrando il segnale e rilevando il livello di ciascuna banda di frequenza (un procedimento molto simile a quello usato negli impianti hi-fi per controllare i toni alti e bassi, ottenendo l'esaltazione di particolari frequenze).
Poiché tutte queste elaborazioni elettroniche vengono effettuate contemporaneamente all'input del segnale parlato, l'analisi risulta praticamente istantanea. L'esecuzione di questa operazione su un segnale digitale, ottenuto mediante un convertitore A/D, richiederebbe l'uso simultaneo di più computer: un metodo decisamente poco pratico e molto costoso. Il metodo di pre-elaborazione è ancora in fase di sperimentazione e di ricerca, pertanto non esistono ancora, in commercio, sistemi che ne facciano uso; potenzialmente, però, pare avere un futuro molto roseo nel campo applicativo.
Dopo che dal segnale originale sono stati estrapolati i dati sul contenuto di frequenza, energia, tono, ecc. (indipendentemente dal metodo usato), il riconoscimento effettivo viene realizzato confrontando le parole immesse con una serie di modelli immagazzinati nella memoria del computer. Questi modelli sono stati creati "allenando" il sistema in questo particolare compito. Le parole che formeranno il "vocabolario" vengono immesse nel sistema, una alla volta, e le informazioni elaborate vengono immagazzinate in una biblioteca di segnali digitali, che costituiscono i modelli. La serie completa di parole viene quindi nuovamente immessa ed il computer la confronta con i modelli precedentemente archiviati. Se i due modelli concordano, la seconda serie di informazioni viene aggiunta alla prima per formare una versione più completa del modello presente in biblioteca. Questa procedura può essere ripetuta più volte, permettendo di aggiungere alla biblioteca sempre nuove informazioni, riferite alle diverse voci delle persone che entrano in contatto vocale con la macchina.
Per riconoscere una parola, il computer deve confrontarne la configurazione di input con uno o più modelli immagazzinati in biblioteca. In molti casi, si verificheranno possibili confronti fra parole diverse via via che singole sillabe, quindi anche di altre parole, si accorderanno con la configurazione di input: per esempio, le prime due sillabe della parola "interno" e "interprete" sono identiche per cui entrambe le parole verranno confrontate con l'input. Alla fine della ricerca il computer individuerà una parola che egli riterrà la più simile a quella immessa.
Il meccanismo di riconoscimento della parola troverà certamente, in futuro, molte applicazioni, ma farà quasi certamente parte di normali procedure applicative, in particolare nella consultazione di database di largo impiego.
Sono già in commercio rudimentali unità di riconoscimento della voce da collegare anche agli home computer. Sistemi come il "Big Ears" e lo "Speech Lab" della Heuristic Inc, impegnano tutte le risorse elaborative per riconoscere solo poche parole pronunciate da una sola persona. L'obiettivo da raggiungere, ovviamente, è la capacità di riconoscere parole pronunciate da una qualunque persona, indipendentemente dal dialetto o dall'accento. Un'interessante possibilità è quella di utilizzare dei video dischi per memorizzare dei modelli vocali standard, limitando l'uso della memoria interna al computer, senza influire sulla velocità di risposta del sistema.



IAN McKINNELL

Controllo ambientale
Le più recenti applicazioni di unità per il riconoscimento della voce si trovano nell'area didattica. Una di queste è chiamata "ambiente limitato" ed implica l'uso di un computer, di un braccio robot ed un certo numero di oggetti, da far manipolare al braccio. Semplicemente, parlando in un microfono, l'operatore può impartire elementari comandi al braccio, ad esempio: "METTERE L'UOVO NEL PORTAUOVO". Oltre ad interpretare i comandi, il computer deve anche memorizzare e verificare la posizione degli oggetti nella propria memoria

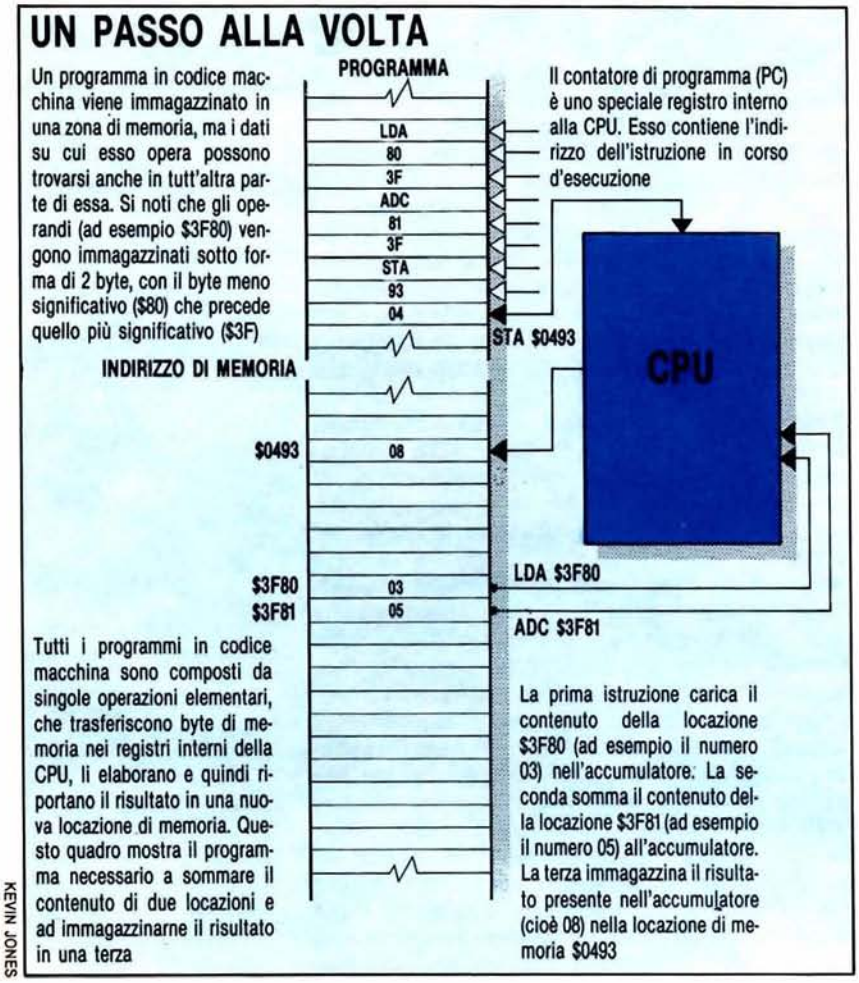
Il linguaggio macchina

Totalmente diverso dal BASIC, questo linguaggio offre una maggiore velocità ed efficienza

Finora, il principale linguaggio di programmazione trattato ne IL MIO COMPUTER è stato il BASIC, ma, con l'accrescersi dell'esperienza e della difficoltà dei problemi che s'intendono risolvere, ci si rende presto conto che il BASIC presenta alcune limitazioni. Prima fra tutte, l'impossibilità di far muovere le immagini grafiche, sullo schermo, con sufficiente rapidità.

Viceversa, la programmazione in codice macchina, pur avendo anch'essa delle limitazioni, dà l'impressione di possedere una velocità pressoché illimitata e moltissimi programmi commerciali sfruttano la maggiore compattezza e velocità di questo linguaggio.

Il codice macchina, come già abbiamo detto, è un linguaggio che il microprocessore del nostro computer è in grado di comprendere direttamente. Il microprocessore, in realtà, può svolgere solo alcune operazioni elementari su dati, ad esempio sommare due cifre di un numero (ma non è capace di moltiplicarle).



KEVIN JONES

Tuttavia, queste operazioni vengono eseguite ad altissima velocità. Il tempo occorrente alla CPU per compiere una specifica operazione è definito dal numero di "cicli di clock" impiegati. Se sul nostro computer la CPU funziona ad 1 MHz allora un ciclo di clock vale 1 microsecondo, pertanto un'operazione che si sviluppa in 4 cicli d'orologio impiegherà 4 milionesimi di secondo.

Tutta la programmazione in codice macchina si basa sulla manipolazione di singoli bit o byte della memoria mediante l'uso di semplici operatori logici, come AND, OR e NOT, e di aritmetica binaria elementare. Ciò comporta, alle volte, un lavoro piuttosto lento.

Inoltre, occorre conoscere l'esatta posizione di ogni dato registrato in memoria. In BASIC, ogni qual volta si incontra un'istruzione come LET A=5, è compito dell'interprete BASIC trovare uno spazio di memoria in cui immagazzinare quella particolare variabile.

Nella programmazione in codice macchina è compito del programmatore fare attenzione a non sovrapporre accidentalmente i dati tra loro.

Vediamo in che cosa consiste il codice macchina (incidentalmente, tutti gli esempi si riferiranno ad una CPU ad 8 bit, tipo lo Z80 ed il 6502, ma il principio è molto simile per gli apparecchi a 16 bit). Il microprocessore è collegato alla memoria del computer mediante due gruppi di canali (un canale non è altro che una connessione elettrica) chiamati bus: abbiamo quindi il bus degli indirizzi ed il bus dei dati (vedere a pag. 144). Esiste anche un terzo bus, quello "di controllo" ma non viene usato dal programmatore.

Il bus degli indirizzi dispone, generalmente, di 16 linee, e, applicando ad esse un valore binario di 16 bit, la CPU è in grado di selezionare uno qualunque dei 65536 byte della "mappa" di memoria (vedere a pag. 329). Nella maggior parte dei personal e degli home computer alcune di queste locazioni si trovano in RAM, altre nella ROM, altre ancora in speciali chip di input/output, altre, infine, non vengono utilizzate affatto. Se la CPU ha bisogno di leggere una particolare locazione di memoria, il contenuto del byte selezionato viene copiato sul bus dei dati (sempre in forma binaria). In modo analogo, la CPU può scrivere un gruppo di 8 bit in una qualunque locazione scelta (una delle linee del bus di controllo indica se si tratta di una lettura o di una scrittura). La CPU non è in grado di distinguere tra le locazioni di memoria ROM e quelle della RAM, perciò un altro compito fondamentale del programmatore è quello di assicurarsi che gli indirizzi delle

locazioni sulle quali intende operare siano corretti.
All'interno del microprocessore c'è una mezza dozzina di "registri", ossia locazioni di memoria "privilegiate", che vengono usati per immagazzinare temporaneamente i risultati di operazioni logiche od aritmetiche (binarie). Alcuni dei registri sono composti da 8 bit, àltri (per esempio quelli in cui vengono depositati indirizzi) da 16 bit. Tra questi ultimi, uno è il registro Contatore di Programma (o PC, da Program Counter) e contiene l'indirizzo di memoria dell'istruzione in codice macchina in corso d'esecuzione. La funzione del Contatore di Programma è assimilabile al numero di linea di un programma BASIC.
Un altro importantissimo registro, questa volta di 8 bit, è l'"accumulatore", nel quale, come il nome stesso suggerisce, vengono accumulati i risultati della maggior parte delle operazioni aritmetiche e logiche. Per illustrare il funzionamento dell'accumulatore, descriviamo le varie fasi di un semplicissimo programma in codice macchina:

1) Caricare l'accumulatore con il contenuto della locazione di memoria $3F80. Gli indirizzi in codice macchina vengono solitamente scritti in esadecimale (vedere a pag. 179) e, per distinguerli dalla forma decimale, usiamo il simbolo "$".
2) Sommare all'accumulatore il contenuto della locazione di memoria $3F81, prevedendo che, se il risultato dovesse occupare più di un byte, occorrerebbe anche un "bit di riporto".
3) Memorizzare il risultato, contenuto nell'accumulatore, nella locazione di memoria $0493.

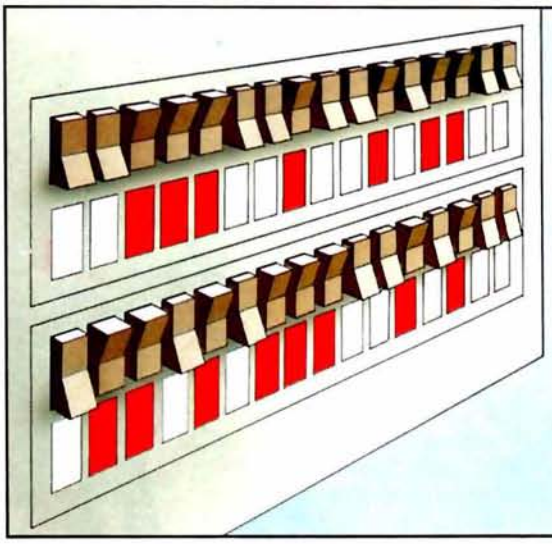
Ciascuno di questi passaggi corrisponde ad un'istruzione in codice macchina ed il programma viene scritto nel seguente modo:

```
LDA $3F80 (caricare l'accumulatore)
ADC $3F81 (sommare con riporto)
STA $0493 (memorizzare il risultato)
```

I commenti tra parentesi, simili all'istruzione REMark del linguaggio BASIC, non hanno alcun effetto pratico. I gruppi di lettere nella prima colonna (LDA, ADC e STA) vengono chiamati "codici mnemonici" o "codici operazionali" (o, dall'inglese: opcode) e si riferiscono al tipo di operazione da eseguire, mentre nella seconda colonna appaiono gli "operandi", ossia i valori sui quali eseguire le operazioni. Un microprocessore dispone, in genere, di alcune dozzine di "codici operazionali" e ciascuno di questi, una volta immesso nella macchina, occupa un solo byte di memoria. Quindi, in realtà, ad ogni singola operazione corrisponde un numero compreso tra 0 e 255 (o, più appropriatamente, tra $00 e $FF). Tuttavia, la stesura di un programma è sicuramente più chiara e comprensibile usando tre lettere simboliche, come LDA, ATC ed STA, che non una lunga sequenza di numeri.
I tre operandi, nell'esempio visto più sopra, non sono altro che dei numeri esadecimali compresi tra $0000 e $FFFF ed occupano, perciò, due byte dello spazio di memoria destinato al programma. Tuttavia, esistono operazioni che elaborano dati di un solo byte o, addirittura, del tutto prive di operandi. Il breve programma mostrato occuperebbe quindi un totale di soli 9 byte, non considerando le tre locazioni di memoria($3F80, $3F81, $0493), usate come deposito di dati. Per fare un significativo paragone, presentiamo un programma BASIC col quale si ottengono i medesimi risultati, ma con un'occupazione di memoria di circa 50 byte (senza contare l'eventuale presenza in RAM del BASIC) ed una lentezza, rispetto alla versione in codice macchina, almeno 100 volte superiore:

```
10 A = PEEK (16256)
20 A = A + PEEK (16257)
30 POKE 1171,A
```

N.B. Le locazioni specificate in questo particolare programma possono non essere adatte ad alcuni tipi di computer.
Più avanti nel corso esamineremo a fondo il linguaggio macchina delle CPU più diffuse (Z80, 6502, ecc.) ed illustreremo il funzionamento degli "Assembler", per la codifica dei programmi in codice macchina.



Luci lampeggianti
In molti film, i computer sono forniti di enormi pannelli, costellati da un'infinità di lampadine colorate. Così erano, più o meno, i quadri di controllo dei computer, fino all'avvento dei microprocessori.
Le file di spie luminose rappresentavano, sotto forma binaria oppure ottale, gli indirizzi ed i dati elaborati dall'unità centrale. Gli interruttori posti sul pannello servivano per immettere o modificare sia i programmi che i dati, secondo il formato in uso

KEVIN JONES

LDA

LDA - Carica l'accumulatore
Trasferisce i contenuti in una singola locazione di memoria (byte) nello speciale registro, chiamato "accumulatore"

STA

STA - Memorizza l'accumulatore
Esegue il procedimento opposto al precedente

ADC

ADC - Somma con riporto
Somma il contenuto di una locazione di memoria al contenuto corrente dell'accumulatore, creando, se necessario, un bit di riporto

SBC

SBC - Sottrae con riporto
Svolge l'operazione inversa alla precedente

JMP

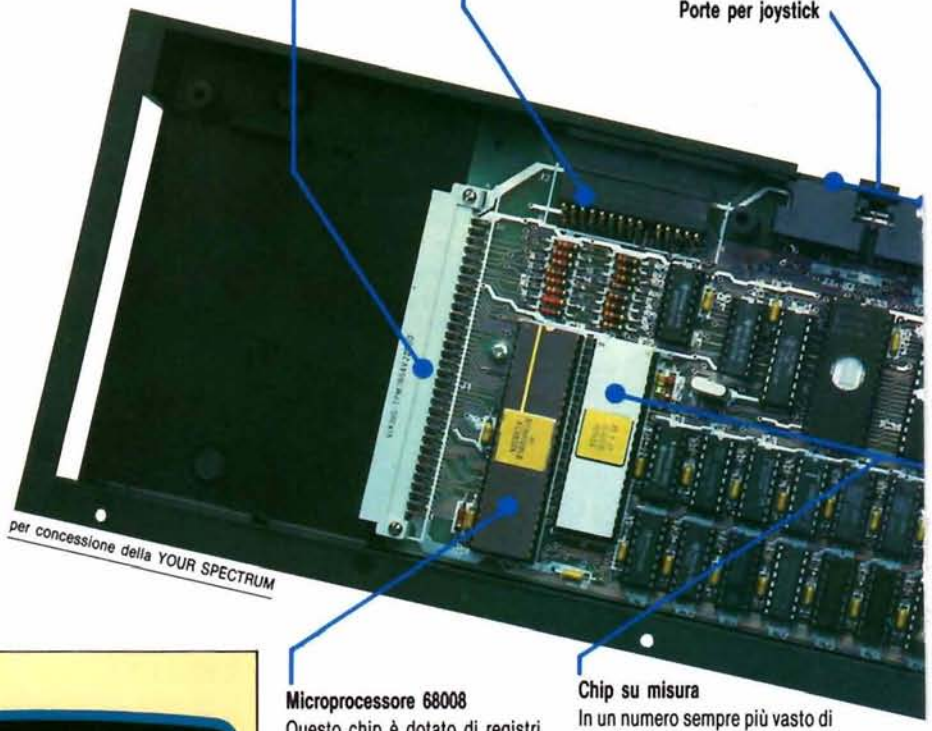
JMP - Salto
Trasferisce l'esecuzione del programma ad una nuova locazione. Il funzionamento è simile a quello dell'istruzione GOTO del BASIC

Gli "Opcode"
Gli Opcode (contrazione di Operational Code, ossia codice operazionale) sono delle combinazioni, in genere di tre lettere, che identificano una particolare operazione da far eseguire alla CPU. Un apposito programma converte questi simboli in codici numerici eseguibili.
Quelli riportati a fianco non sono che un paio di operazioni, tra le molte dozzine a disposizione, che un microprocessore è in grado di eseguire

Il QL Sinclair

Anatomia di un microcomputer a diffusione europea: tecnologie a confronto

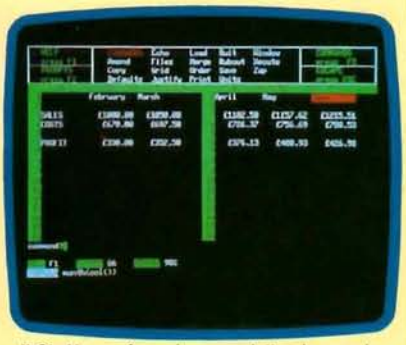
Alla vasta gamma di prodotti Sinclair, noti per la loro qualità ed economicità, si è aggiunto un nuovo gioiello: il QL (Quantum Leap), un computer dalle caratteristiche eccezionali. Rispettando la politica della Sinclair, il QL è un apparecchio dal prezzo decisamente basso, rispetto ai più vicini concorrenti, ed è diretto a quella larga fascia di potenziali acquirenti, interessati ad applicazioni che stanno a cavallo tra il settore professionale e quello domestico.
Interrompendo la consolidata tradizione di usare una CPU Z80 o 6502, nel QL viene impiegato un chip Motorola, della serie 68000, considerato uno dei più sofisticati microprocessori per microcomputer (utilizzato, ad esempio, anche per il Lisa della Apple). Un fatto particolare della CPU 68008 è che, sebbene i registri interni siano a 16 bit (e possano svolgere molte operazioni su numeri binari di 32 bit), il bus dei dati esterni è a soli 8 bit. Tuttavia, la velocità operativa della CPU risente in modo appena percettibile di questa limitazione (le operazioni di caricamento e di memorizzazione dei registri devono essere eseguite a 8 bit per volta), mentre l'aver adottato questo sistema permette di risparmiare notevolmente sui costi delle memorie impiegate.
Il QL viene venduto con una RAM standard di 128 Kbyte ma, con future espansioni, si potranno raggiungere i 512 Kbyte (ossia "mezzo mega" come viene detto in gergo). Questa ampiezza di memoria è particolarmente utile per applicazioni commerciali, perché riduce la frequenza con la quale il sistema deve ricorrere ad unità esterne di memoria (ad esempio, i dischi). Quest'ultima si compone di due Microdrive a cartucce, che dispongono di circa 100 Kbyte ciascuna. In base a tali caratteristiche, il QL può essere considerato un sistema per usi commerciali, ma i Microdrive, a quanto pare, costituiscono un punto piuttosto debole della macchina, soprattutto se confrontati alla notevole efficienza del microprocessore. I tempi di ricerca dei dati sui Microdrive, infatti, si aggirano sui tre secondi e mezzo, mentre la nuova genera-



IL SOFTWARE DEL QL



Il QL Quill è un word processor che visualizza il testo nello stesso formato successivamente usato per la stampa



Il QL Abacus è una lavagna elettronica con la particolare caratteristica che alle varie celle della matrice si possono assegnare nomi



Il QL Archive è un programma per una banca dati. Le dimensioni dei record vengono definite dall'utente con uno "screen editor"



Il QL Easel è un pacchetto software per grafica, utile per grafici e prospetti. È dotato di automatismo per stabilire i valori

La tastiera
La tastiera è basata su una struttura a membrana (protetta quindi dall'umidità o da liquidi accidentalmente versati su di essa) e possiede 65 tasti a corsa completa. Al "tocco" non ha niente da invidiare alle tastiere di macchine per uso commerciale, molto più costose. Ci sono quattro tasti di controllo per il cursore e cinque tasti di funzione programmabili

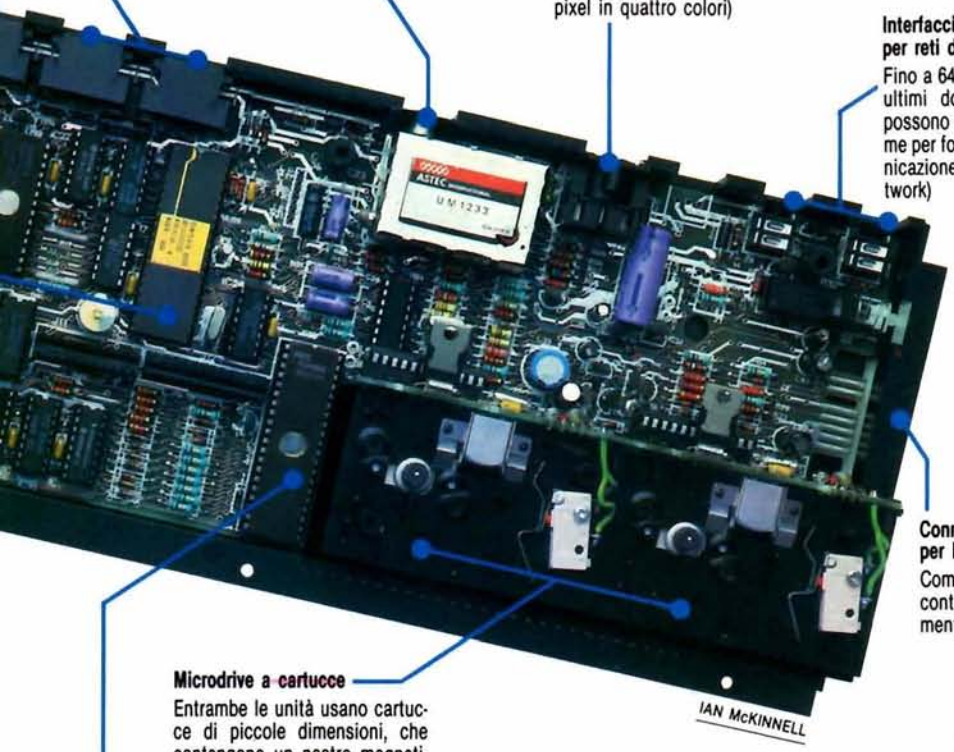


Porte seriali
Sono presenti due porte seriali RS232, utilizzabili per controllare una stampante o un modem. L'interfaccia per la stampante di uso più comune (Centronics) è venduta a parte

Presa TV
Il QL può essere collegato ad un apparecchio televisivo ma, in questo caso, visualizza solo 40 o 60 colonne contro le 80 possibili su monitor

Porta per il monitor
A differenza dello Spectrum, il QL può controllare direttamente un monitor RGB a colori, necessario per sfruttare la massima risoluzione ottenibile (512 x 256 pixel in quattro colori)

Interfaccia per reti di comunicazione
Fino a 64 QL o Spectrum (questi ultimi dotati dell'Interfaccia 1) possono essere collegati assieme per formare una rete di comunicazione locale (Local Area Network)



IAN McKINNELL

Connettore per Microdrive aggiuntivi
Come lo Spectrum, il QL può controllare contemporaneamente fino ad 8 Microdrive

Microdrive a cartucce
Entrambe le unità usano cartucce di piccole dimensioni, che contengono un nastro magnetico ad anello chiuso. La capacità è di 100 Kbyte

Secondo microprocessore
Questo 8049 della Intel controlla la tastiera, il generatore di suoni e le porte seriali, lasciando il 68008 completamente libero per la gestione dei programmi applicativi

PROTOTIPO DI SCHEDA
La foto mostra la scheda del circuito stampato di un prototipo del Quantum Leap: è quindi possibile che alcune componenti del progetto vengano cambiate nel modello definitivo

QL SINCLAIR

DIMENSIONI
mm. 472 x 138 x 46

CPU
68008 Motorola

CLOCK
7,5 MHz

MEMORIA
128 Kbyte di RAM, espandibile a 512 Kbyte. 32 Kbyte di ROM, espandibile a 64 Kbyte

VIDEO
25 linee di 85 caratteri (su monitor). Per la grafica: 512 x 256 pixel (4 colori) oppure 256 x 256 (8 colori)

INTERFACCE
Connettore per TV, per Microdrive a cartucce, per due joystick, per rete locale, per monitor RGB, oltre a due porte seriali RS232

LINGUAGGI FORNITI
BASIC

ALTRI LINGUAGGI DISPONIBILI
Sono annunciati molti linguaggi, tra cui il famoso "C"

ACCESSORI COMPRESI
Manuale operativo e quattro programmi applicativi

DOCUMENTAZIONE
Il manuale è redatto con molta cura: è chiaro ed esauriente. Comprende anche le istruzioni per il funzionamento dei programmi applicativi forniti assieme all'apparecchio

zione di unità a dischetti tradizionali impiega, mediamente, soltanto mezzo secondo.
La Sinclair ha annunciato l'intenzione di produrre un'interfaccia per unità a disco fisso (Winchester), ma non sono previsti progetti per collegare normali unità a dischetti (sebbene a ciò provvederà senz'altro qualche ditta indipendente, come è accaduto per i Sinclair precedenti). Senza dischetti, purtroppo, il QL non può utilizzare il sistema operativo Unix, il cui impiego viene oggi considerato uno dei motivi principali per l'adozione di una CPU Motorola 68000. Questo sistema è destinato a rimpiazzare il CP/M come sistema operativo standard per il software di uso commerciale.
Il QL viene commercializzato con quattro "pacchetti" software standard per applicazioni commerciali, tutti sviluppati dalla casa di software Psion. "Quill" è un word processor, "Abacus" una lavagna elettronica, "Archive" una banca dati e "Easel" un pacchetto per grafica. Tutti girano sotto il sistema operativo in dotazione alla macchina e che la Sinclair ha chiamato QDOS. Del resto, come dimostrato in passato, la forza della Sinclair si identifica spesso con il coraggio di adottare criteri innovativi, piuttosto che adagiarsi su standard industriali comunemente accettati.
La qualità del BASIC fornito con il QL è nettamente superiore a quella del BASIC dello Spectrum e, come se il nome dell'apparecchio non fosse già abbastanza altisonante, la Sinclair ha chiamato SuperBASIC questa nuova versione. Le principali doti del SuperBASIC sono la possibilità di utilizzare procedure (incoraggiando in questo modo una programmazione strutturata) e quella di accedere al sistema operativo direttamente dai programmi in BASIC. Tanto il BASIC che il QDOS sono contenuti nella ROM standard di 32 Kbyte.

Principi del suono

Il generatore audio dell'Atari ha quattro voci indipendenti

La generazione di suoni, sull'Atari, ha notevoli caratteristiche, come si può notare in molte cartucce di giochi, ma, come vedremo, le procedure di controllo sono piuttosto originali. Gli oscillatori audio, ad onda quadra, sono quattro, ciascuno con un'estensione di tre ottave. In più, l'uscita degli oscillatori può essere distorta in sette diversi modi, per dar maggior colore al suono.
Queste possibilità si ottengono facilmente, in BASIC, mediante il comando SOUND che non permette, però, di sfruttare a pieno le caratteristiche del generatore di suono POKEY, esclusivo dell'Atari. Questo chip, dotato di filtri "passa alto" ed altri speciali circuiti, consente una manipolazione dei suoni pressoché illimitata. Di conseguenza, per disporre della gamma completa dei controlli, è necessario ricorrere a complesse sequenze di POKE oppure al linguaggio macchina. I suoni generati vengono emessi dall'altoparlante del televisore al quale è collegato il computer.

Il comando SOUND

Questo è un comando molto semplice ed ha il seguente formato:

SOUND O, P, D, V

O = Oscillatore (0—3)
P = Tono (0—255)
D = Distorsione (numeri da 0 a 14)
V = Volume (1—15)

Ogni comando SOUND seleziona un solo oscillatore: è perciò impossibile mettere in funzione più oscillatori contemporaneamente. Questa limitazione, di solito, non rappresenta un problema, ma desiderando produrre musica polifonica, il ritardo, nell'attacco di un oscillatore rispetto agli altri, risulta assai fastidioso. Il tono viene calcolato in modo piuttosto inconsueto ed alcune frequenze risultano inesatte. La frequenza decresce all'aumentare del numero di tono, producendo una gamma effettiva a partire da un do a 1046,5 Hz, fino ad un do a 130,81 Hz. La tabella indica i valori di tono di alcune note musicali:

	Ottava—1	Ottava—3
(Centr.)	C—121	C—29
	B—128	B—31
	A—144	A—35
	G—162	G—40
	F—182	F— 45
	E—193	E—47
	D—217	D—53
	C—243	C—60

Il parametro di distorsione "D" ha un effetto simile a quello ottenibile con i generatori di rumore presenti sulla maggior parte degli altri computer, ma è molto più versatile. Ciascun numero dispari corrisponde ad una diversa combinazione d'impulsi casuali, che

Luci rinfrescanti

Alcune similitudini tra la grafica dell'Oric e quella dello Spectrum

L'home computer Oric-1 è stato progettato quale rivale dello Spectrum ZX della Sinclair. L'Oric possiede quattro Modi di visualizzazione, solo uno di essi, tuttavia, offre una grafica ad alta risoluzione. I colori disponibili sono 8: i colori dello sfondo e quello di primo piano vengono rispettivamente selezionati dai comandi INK e PAPER. Il BASIC dell'Oric dispone di numerosi comandi speciali per la gestione grafica in alta risoluzione.

Lo schermo è costituito da 28 linee, ciascuna contenente 40 celle di carattere. I caratteri dell'Oric non sono stati progettati usando la matrice standard di 8 × 8 pixel, bensì una particolare matrice di 8 × 6 pixel. In alta risoluzione, lo schermo ha una capacità di 240 × 200 pixel, ma le tre righe inferiori sono riservate ai messaggi di sistema come, ad esempio, i messaggi d'errore. Non esiste un comando PAINT ma, con un minimo d'ingegno, è possibile ottenere risultati grafici del tutto simili usando il comando FILL. Come avviene per lo Spectrum (vedere a pag. 392), è possibile fondere grafica ad alta risoluzione e testi scritti in una stessa immagine. Inoltre, l'Oric-1 permette di colorare tutte le linee che attraversano una stessa cella di carattere.
Esaminiamo più dettagliatamente i Modi a bassa risoluzione di cui l'Oric è fornito. La macchina possiede tre Modi a bassa risoluzione: TEXT, LORES0 e LORES1. L'unica differenza tra gli ultimi due è che fanno uso di differenti gruppi di caratteri. Nel modo TEXT, usando il comando TAB, è possibile posizionare orizzontalmente le lettere alfabetiche. Nei due Modi LORES, tuttavia, questa possibilità viene perfezionata per dar modo al programmatore di visualizzare i caratteri specificando sia la

viene "miscelata" al suono emesso dall'oscillatore. Stranamente, è il valore 10 a produrre un segnale privo di distorsione (e non lo 0 come ci si potrebbe aspettare). Dopo vari esperimenti, un attento uso della distorsione dei suoni permette di produrre interessanti sfumature, particolarmente utili per gli effetti speciali nei giochi.
Il volume "V" può assumere valori che vanno da 1 a 15, ma un livello 7 od 8 è generalmente il più gradevole. Si noti che non è previsto un parametro per influire sulla durata delle note e delle pause tra di esse. Normalmente, si ricorre all'impiego di cicli FOR...NEXT, attentamente calibrati.
Illustriamo l'impiego di SOUND con un esempio, nel quale viene suonato, per 50 passaggi FOR...NEXT, un sol non distorto, nella terza ottava, sull'oscillatore 1 ad un volume di 8:

```
10 SOUND 1, 40, 10, 8
20 FOR N = 1 TO 50: NEXT N
30 END
```

Il comando END in linea 30 spegne tutti gli oscillatori. In alternativa, un nuovo comando SOUND diretto allo stesso oscillatore interrompe l'emissione della precedente nota, per generare la nuova. Un programma che suoni un semplice motivo può essere costruito nel seguente modo:

```
10 REM "DIXIE"
20 FOR H = 1 TO 7
30 READ N: REM LETTURA DELLA NOTA
40 SOUND 3,N,10,7,: REM *EMISSIONE NOTA*
50 FOR P = 1 TO 400: NEXT P: REM *PAUSA*
60 NEXT H
70 DATA 219,162,128,144: REM *Re Sol Si La*
80 DATA 162,193,162: REM *Sol Mi Sol*
90 END
```

È possibile accedere alle funzioni del chip POKEY dell'Atari con programmi in BASIC, depositando, mediante istruzioni POKE, opportuni valori nelle locazioni di memoria 53760 e 53763. Con questo metodo, la generazione di suoni avviene più rapidamente e, inoltre, gli oscillatori possono essere messi in azione tutti assieme. Le informazioni necessarie a realizzare questo procedimento, assieme a numerose e temerarie tecniche in codice macchina, sono contenute in *De Re Atari*, pubblicata dalla Atari Program Exchange (APX) ed anche in *Atari Sound And Graphics* della John Wiles & Son.

posizione orizzontale sia quella verticale, usando il comando PLOT x, y, A$; x ed y sono le coordinate associate ad una particolare posizione, mentre A$ è la variabile che contiene il carattere o la parola da visualizzare. Ecco un programma per chiarire l'uso del comando e ottenere la scrittura verticale di un nome:

```
10 REM LETTERE VERTICALI
20 CLS
30 LORES0
40 A$ = "PAOLO"
50 FOR X = 1 TO 5
60 B$ = MID$(A$,X,1)
70 PLOT 16,11+X,B$
80 NEXT X
90 END
```

Il comando HIRES abilita il Modo ad alta risoluzione dell'Oric. Nel modo HIRES, l'origine del sistema di posizionamento è posto nell'angolo in alto a sinistra dello schermo.
Nel BASIC dell'Oric esistono numerosi comandi utili per la grafica: CURSET x, y, k posiziona il cursore nel punto di coordinate x, y. Il terzo parametro k permette di utilizzare alcune funzioni abbinate al comando CURSET:

Valore di k	Funzioni
0	Traccia un pixel nel colore dello sfondo
1	Traccia un pixel nel colore di primo piano
2	Inverte i colori
3	Non produce alcun effetto

Il comando CURMOV x, y, k è simile a CURSET, ma lo spostamento del cursore è relativo alla sua posizione precedente. DRAW x, y, k traccia una linea retta con inizio nella corrente posizione del cursore e termina nel punto di coordinate x e y. Il comando CIRCLE r, k permette di disegnare circonferenze di raggio "r". PATTERN n è invece un comando che presenta caratteristiche insolite: traccia linee o circonferenze formate da una serie di punti o di brevi tratti. La configurazione esatta viene determinata dal numero "n" che può assumere valori compresi tra 0 e 255. L'Oric interpreta questo valore, trasformandolo nel suo equivalente binario, per poi tracciare una serie ripetuta di punti, brevi linee di varia lunghezza e spazi. I due seguenti esempi chiariscono l'uso di questo comando:

Valore di n	Equivalente binario	Effetto grafico
170	10101010	.....
15	00001111	— —

Infine, c'è il comando FILL a, b, n. Ciascuna riga del video è associata ad un numero, che mette in relazione il carattere presente su di essa con i colori, stabilendo se il carattere è lampeggiante o no. Questo numero prende il nome di "attributo" della linea. Il comando FILL a, b, n, pertanto, serve per assegnare, ad un numero "b" di celle di carattere sulla riga "a", le caratteristiche selezionate da "n".

```
10 REM DISEGNO DI UN CONO
20 HIRES
30 CURSET 120,0,3
40 PAPER 3: INK 4
50 FOR R = 1 TO 65
60 PATTERN 200—R
70 CURMOV 0,2,3,
80 CIRCLE R,1
90 NEXT R
100 END
```

Disegniamo un cono

Il programma evidenzia alcune delle capacità dell'Oric-1 in alta risoluzione. Viene disegnata una figura conica, mediante una serie di circonferenze di raggio crescente. Si noti l'uso del comando PATTERN per tracciare circonferenze con linee tratteggiate

Dpef Dsbdljoh

L'elaborazione di codici cifrati è stata una delle prime applicazioni dei computer

Tutte le nostre comunicazioni con gli altri sono basate su codici: sia nei linguaggi scritti, sia in quelli parlati, le informazioni vengono presentate in modo da risultare comprensibili soltanto a chi è in grado di interpretarne il codice. Lo stesso principio vale per la comunicazione con i computer. La maggior parte dei computer usano particolari dialetti BASIC, comprensibili da un vasto numero di persone, ma noi sappiamo che la stessa macchina deve prima tradurre le istruzioni BASIC in forma esclusivamente numerica, per poi utilizzarle per selezionare le sequenze d'elaborazione definite dal programma ed, infine, produrre i risultati richiesti. Codici di questo genere, sia linguaggi umani sia di programmazione, si incontrano con grande frequenza nella vita di tutti i giorni. Chiunque può imparare il Francese o l'Inglese, oppure linguaggi come il BASIC o il FORTRAN, con un minimo d'impegno e di applicazione.

Ma esistono altri tipi di codici, più precisamente detti "crittogrammi", il cui preciso scopo è quello di risultare incomprensibili a tutti, ad esclusione di un ristretto numero di persone a cui il messaggio crittografato è specificamente indirizzato. Fino alla metà del ventesimo secolo simili codici venivano quasi esclusivamente utilizzati dai governi (o da pochi gruppi industriali di grandi dimensioni) per lo scambio di informazioni molto riservate o addirittura segrete. Ma, recentemente, con il crescente sviluppo ed uso delle linee telefoniche per lo scambio di informazioni (alcune delle quali di notevole valore commerciale), l'uso di crittogrammi è divenuto molto diffuso.

La varietà di cifrari e codici è notevole: dai più semplici, che possono basarsi, ad esempio, sulla somma o sottrazione di un determinato valore a ciascun codice di carattere, oppure sulla costante e ripetuta sostituzione di un carattere con un altro, si passa ai cifrari di immensa complessità, sviluppati in base ai più recenti studi sulla teoria dei numeri. Questi ultimi cifrari non contengono mai elementi di ripetizione, quindi sono pressoché invulnerabili, usando metodi di decodifica basati sull'analisi di frequenza.

La più semplice di tutte le tecniche crittografiche di un certo valore è forse il cosiddetto "Cifrario di Cesare", che fu probabilmente usato per la prima volta al tempo dell'Impero Romano. Per decodificare un messaggio è sufficiente possedere la "chiave" del codice: non occorre, perciò, nascondere in un luogo segreto voluminosi cifrari oppure utilizzare sofisticate macchine di decodifica. Ecco un semplice messaggio crittografato usando il Cifrario di Cesare:

AMPQM GJ KGM AMKNSRCP

Supponiamo, per il momento, di non possedere la chiave: possiamo formulare alcune ragionevoli ipotesi per la decodifica, basate sui gruppi di lettere (ossia sulle spaziature inserite tra di essi, anche se potrebbero essere state introdotte per creare confusione) e sulla ricorrenza delle lettere nel messaggio. La prima ovvia supposizione è che il messaggio sia formato da quattro parole: la prima è composta da cinque lettere, la seconda da due, la terza da tre e la quarta da otto. Possiamo anche rilevare che la prima e l'ultima parola iniziano probabilmente con la stessa lettera così come la prima e la terza finiscono con la stessa lettera. Quest'ultima, cioè la M, è anche la lettera alfabetica che ricorre con maggior frequenza nel messaggio. Questa osservazione è di notevole valore per il crittoanalista (ovviamente se la lingua usata per il messaggio è a lui nota). Nelle lingue europee, ed in particolare in quelle di origine latina, le lettere che ricorrono con maggior frequenza sono le vocali.

Basandosi sul nostro sintetico esempio (18 lettere sono decisamente poche, ed ogni statistico le riterrebbe senz'altro insufficienti per una qualsiasi analisi), i risultati che noi otterremo saranno probabilmente sbagliati. Proviamo comunque ad eseguire una sostituzione e vediamo se i risultati hanno un qualche significato. Sostituiamo, innanzi tutto, la O alla M:

AoPQo GJ KGo AoKNSRCP

Il messaggio è ancora incomprensibile ma abbiamo ancora frecce al nostro arco: che relazione esiste tra la lettera originale e quella che noi abbiamo inserito in sostituzione? Nell'alfabeto, M precede la O di due posizioni. Cosa accadrà se noi applicheremo a tutte le altre lettere del messaggio una sostituzione basata sullo stesso principio? Proviamo con la G:

AoPQo iJ Kio AoKNSRCP

Modifichiamo tutto il messaggio seguendo questa procedura. La A precede la C di due posizioni, la P precede la R e così via. Vediamo subito che la prima parola del crittogramma è CORSO...

COMPRESSIONE DEI DATI

Gli utenti di computer che abbiano bisogno di immagazzinare file di testo di grandi dimensioni sono costantemente alla ricerca di procedure che permettano loro di comprimere i dati contenuti in questi file. Un procedimento che permette di conseguire questo risultato è il cosiddetto "tokenising". Come sui micro computer della serie ZX della Sinclair la battuta di un singolo tasto produce la comparsa sul video di un'intera istruzione BASIC, allo stesso modo un "token" permette di visualizzare una parola o un'intera frase di uso comune.

Inoltre, tecniche di codifica vengono anche usate per comprimere ulteriormente i dati. Compact, un programma di utilità della Unix, viene solitamente utilizzato per comprimere file di testo con un risparmio di memoria di circa il 38% e Clip, che "gira" sotto CP/M, offre risultati ancora migliori. I programmi "compattatori", utilizzati sul Commodore 64 per esempio, operano sui programmi BASIC, eliminando le istruzioni REM, gli spazi non necessari e così via

Il Cifrario di Cesare è quindi un codice di sostituzione che si basa su una procedura di slittamento alfabetico di un certo numero di posizioni, al fine di definire i nuovi valori di ciascun carattere. Può essere reso ancora più complesso usando una stringa di trasformazione della chiave di decodifica, ad esempio 24225. In questo caso la prima lettera slitterà di due posizioni, la seconda di quattro, la terza di due e così via. Raggiunta la fine della stringa, si riprenderà la codifica del messaggio ripartendo dal primo valore di stringa. Se applichiamo questa chiave al messaggio appena interpretato, avremo il seguente risultato:

AKPQJ GH KGJ AKKNPRAP

In questo caso, l'analisi di frequenza risulterà del tutto inutile perché non c'è uniformità nella procedura di sostituzione: ogni lettera corrisponde a lettere diverse, a seconda della posizione che ciascuna occupa all'interno del messaggio. Un altro cifrario basato su principi simili trasformerà il nostro messaggio in questo modo:

CO PROS LMOCMUERIIOT

Se lo esaminiamo con attenzione, ci accorgiamo che questa frase cifrata è l'anagramma di CORSO IL MIO COMPUTER e che possiede anche un numero pari di spazi a quelli presenti nel testo originale. In questi casi, noi abbiamo cercato semplicemente di determinare l'algoritmo del crittogramma, dando esempi sia del testo originale che della sua versione in codice, procedura questa sorprendentemente comune. Se il cifrato deve risultare comprensibile al destinatario del messaggio, allora l'apparente confusione delle lettere deve essere in qualche modo interpretabile. Il seguente cifrario, chiamato Bar Fence (palizzata) per ragioni che tra breve risulteranno evidenti, richiede che il decodificatore sia a conoscenza della chiave (in questo caso il numero 3). Prendiamo i primi cinque caratteri dell'ultimo messaggio e scriviamoli separandoli l'uno dall'altro con tre asterischi:

C***O*** *** ***P***R

Riconosciamo qualcosa? Adesso procediamo scrivendo il testo originale del messaggio su tre linee, saltando dall'una all'altra nel seguente modo:

```
C   O   *   *   P   R
O S * L M O C M U E
  R   I   I   O   T
```

Gli asterischi rappresentano gli spazi tra le parole: la procedura seguita nella codifica adesso ha pieno risalto!
Gli esempi presentati sono stati tutti cifrati, cioè ridefiniti con un metodo di scrittura segreto, basato su sostituzioni o trasformazioni di lettere secondo un criterio prestabilito. La codificazione di un messaggio, invece, può portare a risultati anche notevolmente differenti, poiché vengono sostituiti interi blocchi di lettere con altri, solitamente di minori dimensioni (producendo nello stesso tempo anche una compressione dei dati). Il loro maggior inconveniente risiede nella necessità, sia per il mittente sia per il destinatario dei messaggi, di possedere il cifrario per poter codificare o decodificare i messaggi. Il seguente esempio, riferito a questa tecnica, usa un comune brano di un libro o di un giornale. La decodifica avviene in base ad una sequenza numerica: ogni numero corrisponde alla parola da estrarre dal testo di partenza. Ad esempio, nel seguente brano:

"Il mio libraio preferito è in via del Corso, al 23. Ci sono ben cinque scaffali pieni di libri sui computer"

la chiave del codice potrebbe essere 4,1,2,21.
Proviamo ora a decifrare il messaggio...
Un computer di qualsiasi tipo è di estrema utilità, quando si cerca di codificare o decodificare un messaggio cifrato. Una necessità primaria del Cifrario di Cesare, ad esempio, è la possibilità di muoversi attraverso una stringa alfanumerica, aggiungendo o sottraendo una costante al valore ASCII di ciascun carattere, che può essere quindi inviato alla stampante o al video. Il programma deve poter considerare l'alfabeto come un anello chiuso, ossia, se il valore dello sfasamento è 7 e questo va sommato a Z, deve dedurre che la lettera cercata è la E.
Quindi:

GZTS QFATWT !

IL CIFRARIO DI CESARE

Questo programma, scritto nel dialetto BASIC del Commodore, codifica un testo nel Cifrario di Cesare usando una stringa a multichiave di cinque elementi. Il messaggio viene inserito sotto forma di testo non codificato e, quando viene battuto il tasto RETURN, il programma elabora e stampa la versione in codice. Il testo del messaggio deve essere inserito senza spaziature e segni d'interpunzione

Crittoanalisi

Una delle prime applicazioni dei computer in questo campo è stata la decodifica dei complicatissimi codici di sostituzione a chiavi multiple usati da entrambe le parti in lotta durante la Seconda Guerra Mondiale. I Tedeschi avevano sviluppato una macchina chiamata ENIGMA che generava i propri codici cifrati. I crittogrammi immensamente complicati che essa creava costrinsero gli Alleati a svolgere un improbo lavoro di decodifica. Finalmente il successo arrise al gruppo Colossus che lavorava al Bletchley Park; di questo staff faceva parte anche il famoso scienziato Alan Turing



per concessione del SCIENCE MUSEUM

Una questione di stile

Qualche suggerimento per migliorare lo stile di programmazione ed un paio di nuovi comandi

Nel programma della rubrica computerizzata, presentato in questo corso, vengono impiegate alcune delle più importanti funzioni offerte dal BASIC, ma non tutte. Vedremo in questa lezione qualche altra interessante funzione, utile a chi desideri sviluppare programmi per conto proprio. Naturalmente, è sempre consigliabile tener sotto mano il manuale del BASIC relativo al particolare computer usato, per un diretto confronto sui comandi disponibili.

Programmi in linguaggio macchina

La maggior parte dei BASIC consente l'impiego di sottoprogrammi scritti in linguaggio macchina e collegati al programma principale, stilato in BASIC. Esistono due metodi principali per avvalersi di programmi, per così dire, "misti". Il primo consiste nell'uso delle istruzioni PEEK e POKE per esaminare e modificare particolari locazioni di memoria. Per esempio, LET X = PEEK(1000) legge il valore contenuto nella cella di memoria 1000 e lo assegna alla variabile X. Eseguendo un comando PRINT X, possiamo visualizzare il valore letto. Ecco un breve programma col quale leggere e visualizzare il contenuto di 16 locazioni contigue di memoria:

```
10 INPUT "INDIRIZZO INIZIALE";S
20 PRINT
30 FOR L = 1 TO 16
40 LET A = PEEK(S)
50 PRINT "LA LOCAZIONE ";S;" CONTIENE:";A
60 LET S = S + 1
70 NEXT L
80 PRINT "PREMERE UNO SPAZIO PER LE 16 LOCAZIONI
   SUCCESSIVE"
90 PRINT "OPPURE UN RETURN PER FINIRE"
100 FOR H = 1 TO 1
110 LET C$ = INKEY$
120 IF C$ <> CHR$(13) AND C$ <> " " THEN H = 0
130 NEXT H
140 IF C$ = CHR$(13) THEN GOTO 160
150 GOTO 30
160 END
```

Il ciclo compreso tra le linee 100 e 130 controlla ciò che viene digitato alla tastiera: se il carattere è un RETURN (= 13 in codice ASCII) l'esecuzione passa alla linea 160, dove il programma termina, altrimenti viene fatto un salto alla linea 30, dove la visualizzazione prosegue.
Volendo, si può anche visualizzare il carattere corrispondente al valore letto in memoria. Per far ciò basta inserire PRINT CHR$(A), ma si presti attenzione al fatto che il minimo codice visualizzabile è il 32 (corrispondente al carattere spazio) ed i valori inferiori a questo vengono spesso usati come codici di controllo per le funzioni più varie. L'unica eccezione è costituita dal codice 13, universalmente corrispondente a RETURN (ossia Ritorno Carrello) e, forse, dal codice 7 che provoca l'emissione di un "bip" sull'altoparlante del sistema.
La funzione POKE, opposta alla PEEK, serve per scrivere un valore compreso tra 0 e 255 in una locazione di memoria RAM, specificata nel comando stesso. La POKE deve essere adoperata con estrema cautela, poiché andare a modificare locazioni a caso, nella RAM, può provocare effetti catastrofici sul programma in corso d'esecuzione.
Usando una serie di POKE, si può depositare un sottoprogramma, in linguaggio macchina, in una zona (sicuramente libera) della RAM. Il programma in BASIC può "chiamare" il sottoprogramma mediante l'impiego dell'istruzione CALL. La programmazione in linguaggio macchina viene trattata in una serie di lezioni a parte ed, inoltre, non è possibile offrire un esempio generico di questo particolare modo di procedere. Comunque, basti sapere che ricorrere a sottoprogrammi in linguaggio macchina permette di ottenere una maggiore velocità di risposta, specialmente per quanto concerne la grafica.

Spostiamo il cursore

Molti computer offrono la possibilità d'intervenire facilmente sulla posizione del cursore (ossia del piccolo simbolo che indica dove verrà visualizzato il prossimo carattere), ma anche su quelli meno recenti è spesso possibile spostare a piacimento il cursore, purché si conoscano i codici di controllo relativi ai vari movimenti. Questo corto programma può servire ad individuare i codici ASCII corrispondenti ai vari caratteri della tastiera:

```
1 REM RICERCA DEI CODICI ASCII PER IL CURSORE
10 PRINT "PREMERE UN TASTO";
20 FOR H = 1 TO 1
30 LET K$ = INKEY$
40 IF K$ = " " THEN H = 0
50 NEXT H
60 PRINT ASC(K$)
70 GOTO 10
80 END
```

Possiamo verificare, per esempio, se i codici corrispondenti ad alcuni tasti sono quelli standard: 13 = RETURN, 27 = ESCape, 32 = carattere spazio, ecc.
Nel Sord M5, per esempio, il codice per lo spostamento del cursore a sinistra è 8, a destra è 28, verso l'alto è 29 e verso il basso è 30. Una volta individuati i valori cercati, possiamo verificarne il funzionamento con un programma simile a questo:

```
10 PRINT CHR$(12) : REM PULIZIA DELLO SCHERMO
20 FOR L = 1 TO 39
30 PRINT "*";
40 NEXT L
50 FOR L = 1 TO 32
60 PRINT CHR$(8): REM CODICE PER 'CURSORE A SINISTRA'
70 NEXT L
80 FOR L = 1 TO 4
90 PRINT "@";
100 NEXT L
110 END
```

Il programma genera una visualizzazione di questo tipo:

```
********************** @@@@ **********************
```

Le linee da 20 a 40 visualizzano una riga di 39 asterischi; le linee da 50 a 70 spostano il cursore a sinistra (o indietro se si preferisce) di 22 posizioni; le linee da 80 a 100, infine, visualizzano 4 "chioccioline" (@). Sfruttando opportunamente i codici per lo spostamento del cursore si può visualizzare un carattere o un simbolo grafico in una qualsiasi parte dello schermo, magari calcolando la posizione con una formula matematica. Ciò è particolarmente utile, ad esempio, per il tracciamento di grafici.
Si provi, a tal scopo, il seguente programma:

```
10 PRINT "VISUALIZZAZIONE DI UN GRAFICO A BARRE CON 3
   VARIABILI"
20 INPUT "DIGITARE 3 VALORI (SEPARATI DA VIRGOLE)";X,Y,Z
30 PRINT
40 FOR L = 1 TO 2
50 FOR A = 1 TO X
60 PRINT "*";
70 NEXT A
80 PRINT CHR$(13)
90 NEXT L
100 FOR L = 1 TO 2
110 FOR A = 1 TO Y
120 PRINT "+";
130 NEXT A
140 PRINT CHR$(13)
150 NEXT L
160 FOR L = 1 TO 2
170 FOR A = 1 TO Z
180 PRINT "#";
190 NEXT A
200 PRINT CHR$(13)
210 NEXT L
220 PRINT
230 END
```

Le "barre" vengono visualizzate orizzontalmente, iniziando da sinistra. Si noti l'impiego di PRINT CHR$(13), necessario alle linee 80, 140 e 200, poiché il punto e virgola, usato nelle istruzioni PRINT, sopprime l'emissione automatica di un "ritorno carrello".

Ancora sulle variabili

Finora, abbiamo distinto le variabili in due sole categorie: le variabili numeriche e quelle alfanumeriche. In effetti, esistono altri tipi di variabili numeriche in BASIC ed un buon programmatore specifica sempre quale tipo impiegare ai fini di una maggiore precisione o di un migliore impiego della memoria.
Quando in un programma si "dichiara" una variabile (in BASIC ciò accade quando la si usa per la prima volta) ad essa viene automaticamente riservata una zona di memoria RAM, la cui ampiezza è determinata dal tipo di variabile. Se la variabile è di tipo "intero" (ossia contiene valori compresi tra −32676 e + 32676, ad esempio: LET PUNTEGGIO = TOTALE + PREMIO − PENALI) il BASIC riserverà soltanto 2 byte di memoria (tanti ne bastano per contenere quell'arco di valori); mentre nel caso di una variabile che debba contenere valori molto più grandi (ad esempio, LET AREA = PI * RAGGIO * RAGGIO) è necessario un numero maggiore di byte.
Abbiamo visto che il simbolo "$", associato al nome di una variabile, significa che intendiamo usarla per contenere caratteri alfanumerici (ad esempio N$ = "GIOVANNI"). I nomi di variabili prive di alcun simbolo vengono considerate numeriche, o, più precisamente, variabili numeriche a singola precisione. Riassumiamo brevemente i vari tipi di variabili. L'arco di valori accettato da quelle numeriche dipende dal BASIC, e viene indicato nel relativo manuale.

Tipo	Simbolo	Esempio
Intero	%	CONTO%
Singola precisione	! (facoltativo)	AREA!
Doppia precisione	#	CAPITALE#
Alfanumerico	$	NOME$

Da notare che in alcuni computer (nello Spectrum, per esempio) non esiste distinzione tra variabili intere ed a singola precisione (appartengono tutte al secondo tipo). Tuttavia, le variabili in singola precisione, nello Spectrum, hanno 9 cifre significative, mentre nel BASIC della Microsoft ne hanno solo 7. Nel seguente brano di programma viene mostrato l'uso dei vari tipi di variabili, sui quali avremo ancora l'occasione di parlare più avanti nel corso.

```
70 LET GIOCATORE$ = "GIANNI" : REM VARIABILE STRINGA
80 LET PUNTEGGIO% = 0 : REM VARIABILE INTERA
90 LET PI! = 3.1416 : REM VARIABILE A SINGOLA PRECISIONE
100 LET AREA# = PI! * R * R : REM VARIABILE DOPPIA
    PRECISIONE
110 LET TURNO = 6: REM VARIABILE A SINGOLA PRECISIONE
```

Nella prossima lezione vedremo ancora i pro ed i contro del linguaggio BASIC.

Basic & Basic



Sul Lynx, nel primo programma, sostituire le linee 110, 120 e 140 con:

110 C = KEYN
120 IF (C<>13) AND (C<>32) THEN H = 0
140 IF C = 13 THEN GOTO 160



Il terzo programma non produce i risultati attesi



Sul Micro BBC sostituire PEEK(S) con ?S

RUBRICA COMPUTERIZZATA PER LO SPECTRUM

Questa è la versione del programma, adattato allo Spectrum della Sinclair.
Le modifiche necessarie per il Dragon 32, il Micro BBC, il Commodore 64 ed il Vic-20 saranno riportate nelle prossime lezioni

```
1 REM * Creazione del file dati *
2 DIM N$(50,30)
3 LET N$(1)="@PRIMO"
4 INPUT"INSERIRE IL NASTRO, AVVIARLO E PREMERE RETURN";A$
5 SAVE "NFLD" DATA N$()
6 INPUT"RIAVVOLGERE IL NASTRO, AVVIARLO E PREMERE RETURN";A$
7 VERIFY "NFLD" DATA N$()
8 STOP
```

Questo è il programma per la creazione, su nastro, dell'archivio dati. L'operazione va effettuata una sola volta, prima di lanciare il programma principale e non sarà più necessaria, salvo non si desideri creare una nuova rubrica su di un nastro diverso.

```
10 REM 'MAINPG'
20 REM == INITL ===
30 GOSUB 1000
40 REM == GREETS ==
50 GOSUB 3000
60 FOR M=1 TO 1
70 LET M=0
80 REM == SCELTA ==
90 GOSUB 3500
100 REM == EXECUT ==
110 GOSUB 4000
120 IF SCEL () 9 THEN 60
130 NEXT M
140 STOP

1000 REM SUBROUTINE == INITIL ==
1010 GOSUB 1100 : REM == CREMAT == (CREAZIONE MATRICI)
1020 GOSUB 1400 : REM == RDINFL == (LETTURA DEL FILE)
1030 GOSUB 1600 : REM == SETFLG == (ASSEGNAZIONE DELLE FLAG)
1090 RETURN

1100 REM SUBROUTINE == CREMAT == (CREAZIONE MATRICI)
1110 DIM N$(50,30)
1120 DIM M$(50,30)
1130 DIM S$(50,30)
1140 DIM T$(50,15)
1150 DIM C$(50,15)
1160 DIM R$(50,15)
1170 DIM X$(50,30)
1180 DIM B$(30):DIM Z$(30)
1190 DIM U$(30):DIM W$(15)
1210 LET DIMENSIONE = 0
1220 LET RMOD = 0
1230 LET SRTD = 1
1240 LET CORR = 0
1250 LET Z$="@PRIMO"
1260 LET Q$=B$
1300 RETURN

1400 REM SUBROUTINE == RDINFL == (LETTURA DEL FILE)
1405 INPUT"INSERIRE IL NASTRO, AVVIARLO E PREMERE RETURN";A$
1410 LOAD "NFLD" DATA N$()
1420 IF N$(1)=Z$ THEN LET Q$=Z$:RETURN
1430 LOAD "MFLD" DATA M$()
1440 LOAD "SFLD" DATA S$()
1450 LOAD "TLFD" DATA T$()
1460 LOAD "CFLD" DATA C$()
1470 LOAD "TELFLD" DATA R$()
1480 LOAD "NDXFLD" DATA X$()
1485 INPUT"FERMARE IL NASTRO E PREMERE RETURN";A$
1490 REM 'FLSIZE'
1500 LET DIMENSIONE=51
1510 FOR L=1 TO 50
1520 IF N$(L)=B$ THEN LET DIMENSIONE=L:LET L=50
1530 NEXT L
1540 RETURN

1600 REM SUBROUTINE == SETFLG == (ASSEGNAZIONE DELLE FLAG)
1610 REM LE FLAG VENGONO ASSEGNATE DOPO LA LETTURA DEL FILE
1640 IF Q$=Z$ THEN LET DIMENSIONE = 1
1690 RETURN

3000 REM SUBROUTINE == GREETS == (MESSAGGIO INTRODUTTIVO)
3010 CLS : REM PULIZIA DELLO SCHERMO
3020 PRINT:PRINT:PRINT:PRINT
3060 PRINT TAB(5);"RUBRICA COMPUTERIZZATA"
3070 PRINT TAB(6);"PRESENTATA NEL CORSO"
3080 PRINT TAB(7);"'IL MIO COMPUTER'"
3090 PRINT
3100 PRINT TAB(1);"(PREMERE UNO SPAZIO PER CONTINUARE)"
3110 FOR L = 1 TO 1
3120 IF INKEY$ () " " THEN L = 0
3130 NEXT L
3140 CLS
3150 RETURN

3500 REM SUBROUTINE == SCELTA == (MENU DELLE OPZIONI)
3510 REM
3520 IF Q$ = Z$ THEN GOSUB 3860: RETURN
3540 REM == MENU ==
3550 CLS
3560 PRINT TAB(10);"MENU DELLE OPZIONI"
3570 PRINT:PRINT:PRINT
3600 PRINT "1. TROVARE UN RECORD (USANDO IL NOME)"
3610 PRINT "2. TROVARE UN RECORD (USANDO PARTE DEL NOME)"
3620 PRINT "3. TROVARE RECORD (USANDO LA LOCALITA')"
3630 PRINT "4. TROVARE UN RECORD (USANDO LE INIZIALI)"
3640 PRINT "5. ELENCO DI TUTTI I RECORD"
3650 PRINT "6. INSERIMENTO DI UN RECORD"
3660 PRINT "7. MODIFICA DI UN RECORD"
3670 PRINT "8. ANNULLAMENTO DI UN RECORD"
3680 PRINT "9. USCITA DAL PROGRAMMA"
3690 PRINT:PRINT
3710 REM == INCHOI ==
3750 PRINT "QUALE (1 - 9)"
3760 FOR L = 1 TO 1
3770 FOR I = 1 TO 1
3780 LET A$ = INKEY$
3790 IF A$ = "" THEN I = 0
3800 NEXT I
3810 LET SCEL = CODE A$-48
3820 IF (SCEL ( 1) OR (SCEL ) 9) THEN LET L = 0
3840 NEXT L
3850 RETURN

3860 REM SUBROUTINE == FIRSTM == (FILE VUOTO)
3870 LET SCEL = 6
3880 CLS
3890 PRINT
3900 PRINT TAB(4);"L'ARCHIVIO E' VUOTO ED OCCORRE"
3910 PRINT TAB(4);"INIZIARE INSERENDO IL PRIMO"
3920 PRINT TAB(4);"RECORD DEL FILE"
3930 PRINT
3940 REM * CONTINUA *
3950 GOSUB 3100
3990 RETURN

4000 REM SUBROUTINE == EXECUT ==
4040 IF SCEL = 1 THEN GOSUB 5700
4050 REM 2 E' == FNDNMS ==
4060 REM 3 E' == FNDTWN ==
4070 REM 4 E' == FNDINT ==
4080 IF SCEL = 5 THEN GOSUB 8100
4090 IF SCEL = 6 THEN GOSUB 4200
4100 IF SCEL = 7 THEN GOSUB 6600
4110 IF SCEL = 8 THEN GOSUB 7500
4120 IF SCEL = 9 THEN GOSUB 5000
4140 RETURN

4200 REM SUBROUTINE == AGGREC ==
4210 CLS
4220 INPUT "NOMINATIVO ";N$(DIMENSIONE)
4230 INPUT "INDIRIZZO ";S$(DIMENSIONE)
4240 INPUT "LOCALITA' ";T$(DIMENSIONE)
4250 INPUT "NAZIONE ";C$(DIMENSIONE)
4260 INPUT "NUMERO DI TELEFONO ";R$(DIMENSIONE)
4270 LET RMOD = 1 : LET SRTD = 0
4280 LET X$(DIMENSIONE) = STR$(DIMENSIONE)
4290 LET Q$ = ""
4300 GOSUB 4500
4310 LET SCEL = 0
4320 LET DIMENSIONE = DIMENSIONE + 1
4330 RETURN

4500 REM SUBROUTINE == MODNOM ==
4510 REM CONVERTE IL CONTENUTO DI N$ IN MAIUSCOLE
```

```
4520 LET D$ = N$(DIMENSIONE):LET P$=""
4530 FOR L = 1 TO LEN(D$)
4540 LET A$ = D$(L)
4550 LET T = CODE A$
4560 IF T )= 97 THEN T = T - 32
4570 LET A$ = CHR$ T
4580 LET P$ = P$ + A$
4590 NEXT L
4600 LET D$=P$:LET P$="":LET A$="":LET T=LEN(D$):LET S=0
4610 REM LOCALIZZA IL PRIMO SPAZIO
4620 FOR L = 1 TO T
4630 IF D$(L) = " " THEN S = L:LET L=T
4640 NEXT L
4650 REM RIMUOVE CARATTERI ESTRANEI,NOME IN P$
4660 FOR L=1 TO S-1
4670 IF CODE(D$(L)) ) 64 THEN LET P$=P$+D$(L)
4680 NEXT L
4690 REM RIMUOVE CARATTERI ESTRANEI, COGNOME IN A$
4700 FOR L=S+1 TO LEN(D$)
4710 IF CODE(D$(L)) ) 64 THEN A$=A$+D$(L)
4720 NEXT L
4730 LET M$(DIMENSIONE)=A$ + " " + P$
4740 LET P$ = "":LET A$ = "":LET S=0
4750 RETURN

5000 REM SUBROUTINE == EXPROG ==
5010 IF (RMOD = 0) AND (SRTD = 1) THEN RETURN
5020 IF (RMOD = 1) AND (SRTD = 0) THEN GOSUB 5200
5030 GOSUB 5600
5040 RETURN

5200 REM SUBROUTINE == SRTREC ==
5210 FOR K=1 TO 1
5220 LET S = 0
5230 FOR L=1 TO DIMENSIONE-2
5240 LET T=L+1
5250 IF M$(L) ) M$(T) THEN GOSUB 5400
5260 NEXT L
5270 IF S=1 THEN LET K=0
5280 LET SRTD = 1
5290 RETURN

5400 REM SUBROUTINE == SWPREC == (SCAMBIO RECORD)
5410 LET U$ = N$(L):LET N$(L)=N$(T):LET N$(T)=U$
5420 LET U$ = M$(L):LET M$(L)=M$(T):LET M$(T)=U$
5430 LET U$ = S$(L):LET S$(L)=S$(T):LET S$(T)=U$
5440 LET W$ = T$(L):LET T$(L)=T$(T):LET T$(T)=W$
5450 LET W$ = C$(L):LET C$(L)=C$(T):LET C$(T)=W$
5460 LET W$ = R$(L):LET R$(L)=R$(T):LET R$(T)=W$
5470 LET X$(L)=STR$(L)
5480 LET X$(T)=STR$(T)
5490 LET S=1
5500 RETURN

5600 REM SUBROUTINE == SAVREC ==
5605 INPUT"INSERIRE IL NASTRO E PREMERE RETURN";A$
5610 SAVE "NDFL" DATA N$()
5620 SAVE "MFLD" DATA M$()
5630 SAVE "SFLD" DATA S$()
5640 SAVE "TFLD" DATA T$()
5650 SAVE "CFLD" DATA C$()
5660 SAVE "TELFLD" DATA R$()
5670 SAVE "NDXFLD" DATA X$()
5680 INPUT"FERMARE IL NASTRO E PREMERE RETURN";A$
5690 RETURN

5700 REM SUBROUTINE == FNDREC ==
5710 CLS
5720 IF SRTD = 0 THEN GOSUB 5200
5730 PRINT:PRINT
5740 PRINT TAB(5); "RICERCA DI UN RECORD"
5750 PRINT TAB(5); "IN BASE AL NOME"
5760 PRINT
5770 PRINT TAB(5); "IMMETTERE IL NOMINATIVO"
5780 PRINT TAB(5); "(NEL FORMATO: NOME, COGNOME)"
5790 PRINT:PRINT
5800 INPUT "NOMINATIVO ";N$(DIMENSIONE)
5810 GOSUB 4500
5820 LET U$ = M$(DIMENSIONE)
5830 LET FDO = 1
5840 LET CMA = DIMENSIONE - 1
5850 FOR X = 1 TO 1
5860 LET MED = INT((FDO+CMA)/2)
5870 IF M$(MED) () U$ THEN X = 0
5880 IF M$(MED) ( U$ THEN FDO=MED+1
5890 IF M$(MED) ) U$ THEN CMA=MED-1
5900 IF FDO ) CMA THEN LET X=1
5910 NEXT X
5920 IF FDO ) CMA THEN LET CORR=0
5930 IF FDO ( = CMA THEN LET CORR = MED
5940 IF CORR = 0 THEN GOSUB 6400:RETURN
5950 CLS
5960 PRINT
5970 PRINT TAB(9);"RECORD TROVATO"
5980 PRINT
5990 PRINT "NOMINATIVO:",N$(CORR)
6000 PRINT "INDIRIZZO:",S$(CORR)
6010 PRINT "CITTA`:",T$(CORR)
6020 PRINT "NAZIONE:",C$(CORR)
6030 PRINT "N. TEL.:",R$(CORR)
6040 PRINT
6050 PRINT TAB(3);"PREMERE QUALSIASI TASTO COPIA SU
     STAMPANTE"
6060 PRINT TAB(3);"OPPURE UNO SPAZIO PER CONTINUARE"
6070 FOR I = 1 TO 1
6080 LET A$ = INKEY$
6090 IF A$ = "" THEN LET I = 0
6100 NEXT I
6110 IF A$ () " " THEN GOSUB 6200
6120 RETURN

6200 REM  SUBROUTINE ==LSTCUR== (COPIA SU STAMPANTE)
6210 LPRINT
6220 LPRINT "NOMINATIVO:",N$(CORR)
6230 LPRINT "INDIRIZZO:",S$(CORR)
6240 LPRINT "CITTA`:",T$(CORR)
6250 LPRINT "NAZIONE:",C$(CORR)
6260 LPRINT "N. TEL.:",R$(CORR)
6270 LPRINT:LPRINT
6280 RETURN

6400 REM SUBROUTINE ==NOTREC== (RECORD NON TROVATO)
6410 CLS
6420 PRINT TAB(7); "RECORD NON TROVATO"
6430 PRINT TAB(2); "SOTTO LA FORMA: ";N$(DIMENSIONE)
6440 PRINT
6450 REM *CONTINUA*
6460 GOSUB 3100
6470 RETURN

6600 REM SUBROUTINE ==MODREC== (MODIFICA RECORD)
6610 CLS
6620 PRINT:PRINT:PRINT
6630 LET E$=CHR$ 13
6640 PRINT TAB(2);"PER MODIFICARE UN RECORD OCCORRE
6650 PRINT TAB(2);"PRIMA INDIVIDUARLO"
6660 GOSUB 5720
6670 IF CORR=0 THEN RETURN
6680 PRINT
6690 PRINT "MODIFICA AL NOMINATIVO ?"
6700 GOSUB 8000
6710 IF A$ () E$ THEN GOTO 6780
6720 INPUT "NUOVO NOMINATIVO ";N$(CORR)
6730 LET RMOD = 1
6740 LET SRTD = 0
6750 N$(DIMENSIONE)=N$(CORR)
6760 GOSUB 4500
6770 LET M$(CORR)=M$(DIMENSIONE)
6780 PRINT
6790 PRINT "MODIFICA ALLA VIA ?"
6800 GOSUB 8000
6810 IF A$ () E$ THEN GOTO 6850
6820 LET RMOD = 1
6830 INPUT "NUOVA VIA ";S$(CORR)
6840 PRINT
6850 PRINT "MODIFICA ALLA CITTA` ?"
6860 GOSUB 8000
6870 IF A$ () E$ THEN GOTO 6910
6880 LET RMOD = 1
6890 INPUT "NUOVA CITTA` ";T$(CORR)
6900 PRINT
6910 PRINT "MODIFICA ALLA NAZIONE ?"
6910 GOSUB 8000
6920 IF A$ () E$ THEN GOTO 6960
6930 LET RMOD = 1
6940 INPUT "NUOVA NAZIONE ";C$(CORR)
6950 PRINT
6960 PRINT "MODIFICA AL N. TELEFONICO ?"
6970 GOSUB 8000
6980 IF A$ = E$ THEN RMOD = 1
6990 IF A$ = E$ THEN INPUT "NUOVO TELEFONO ";R$(CORR)
7000 RETURN

7500 REM SUBROUTINE ==DELREC== (ANNULLAMENTO RECORD)
7510 CLS
7520 PRINT:PRINT:PRINT:PRINT
7530 LET E$ = CHR$ 13
7540 PRINT TAB(6); "PER CANCELLARE UN RECORD OCCORRE"
7550 PRINT TAB(6); "PRIMA INDIVIDUARLO"
7560 GOSUB 5720
7570 IF CORR = 0 THEN RETURN
7580 PRINT
7590 PRINT "E` QUESTO IL RECORD DA CANCELLARE ?"
7600 PRINT "ATTENZIONE: VALE LA PRIMA RISPOSTA!"
7610 PRINT
7620 PRINT TAB(5);"PREMERE RETURN PER CANCELLARE"
7630 PRINT TAB(5);"OPPURE SPAZIO PER CONTINUARE"
7640 FOR I = 1 TO 1
7650 LET A$ = INKEY$
7660 IF (A$ () E$) AND (A$ () " ") THEN LET I = 0
7670 NEXT I
7680 IF A$ = " " THEN RETURN
7690 FOR L = CORR TO DIMENSIONE - 2
7700 LET T=L+1
7710 LET N$(L) = N$(T)
7720 LET M$(L) = M$(T)
7730 LET S$(L) = S$(T)
7740 LET T$(L) = T$(T)
7750 LET C$(L) = C$(T)
7760 LET R$(L) = R$(T)
7770 LET X$(L) = X$(T)
7780 NEXT L
7790 LET RMOD = 1
7800 LET DIMENSIONE = DIMENSIONE - 1
7810 RETURN

8000 REM SUBROUTINE PER IMMISIONE RISPOSTA
8010 PRINT "(RETURN = MODIFICA   SPAZIO = NESSUNA MODIFICA)";
8020 FOR I = 1 TO 1
8030 LET A$ = INKEY$
8040 IF (A$ () E$) AND (A$ () " ") THEN I = 0
8050 NEXT I
8060 PRINT
8070 RETURN

8100 REM SUBROUTINE ==LSTREC==
8110 FOR CORR = 1 TO DIMENSIONE - 1
8120 GOSUB 5980
8130 NEXT CORR
8140 CORR = 0
8150 RETURN
```

Il MARK 1

Il primo computer programmabile è stato sviluppato presso l'università di Manchester

Alla fine della II Guerra Mondiale, l'università di Manchester (GB) assegnò due nuove cattedre: una al matematico Max Newman, a seguito del suo lavoro di decifrazione di codici, operato sul Colossus (il primo computer elettromeccanico del mondo); l'altra cattedra andò a F.C. Williams, ingegnere di impianti radar, che venne, nello stesso periodo, nominato preside della facoltà di Ingegneria Elettrica. Williams condusse con sé un giovane assistente, Tom Kilburn, un esperto in problemi riguardanti unità di memoria elettroniche ad impulsi, che aveva studiato queste apparecchiature nel suo lavoro sui radar durante la guerra. Kilburn più tardi divenne il primo docente della nuova materia che riguardava lo studio dei computer all'Università di Manchester.

Durante una visita ad alcune fabbriche di radar americane, nel 1946, a Williams venne mostrato il prototipo del computer a valvole ENIAC (vedere a pag. 46). Al suo ritorno in Inghilterra, egli convinse la Royal Society ad investire 35.000 sterline nella realizzazione, a Manchester, di un laboratorio di ricerca sui calcolatori. L'università di questa città non fu l'unica a dedicarsi alla costruzione di un computer programmabile: anche l'università di Pennsylvania si dedicò alla progettazione dell'EDVAC, mentre alla Cambridge University si lavorava sull'EDSAC ed al National Phisical Laboratory era in corso lo sviluppo dell'ACE (vedere a pag. 88). Tutti questi altri progetti, comunque, usavano una memoria costituita da linee di ritardo al mercurio. Il gruppo di lavoro di Manchester, invece, costruì un sistema di memorizzazione, escogitato da Williams, basato sull'impiego di un tubo a raggi catodici. Nell'autunno del 1947, Williams riuscì a memorizzare, per un periodo di alcune ore, ben 2048 bit.

Nel giugno del '48, sul computer Manchester Mark I, dotato di un tubo "Williams", fu possibile eseguire con successo un programma di calcolo, assegnando così all'apparecchio la qualifica di "primo computer programmabile del mondo". Il Mark I era capace di eseguire un'istruzione in appena 1,2 millisecondi. Grazie al particolare sistema di conservazione delle informazioni, l'apparecchio aveva il vantaggio di disporre di una memoria ad accesso casuale, oltre a permettere la visualizzazione del contenuto della memoria centrale o del registro di controllo.

Una volta accertata l'affidabilità del tubo Williams, venne costruita una nuova versione del Mark I, in grado di lavorare su problemi di progettazione ottica e sulla generazione di numeri primi. Sir Ben Lockspeiser, un illustre scienziato del tempo, rimase talmente colpito dalle prestazioni del computer, che appoggiò in pieno la realizzazione di una versione commerciale dell'apparecchio, da parte di un'industria di Manchester. Il Mark I Ferranti fu pronto nel febbraio del 1951 e, precedendo l'UNIVAC di 5 mesi, divenne il primo computer ad esser commercializzato.

Un'importante innovazione, presente sul Mark I Ferranti, era la capacità di modificare le istruzioni del programma, durante il processo di elaborazione, per mezzo di una seconda memoria chiamata tubo "B". Questo rivoluzionario principio rendeva più rapida l'elaborazione dei programmi. L'IBM utilizzò alcuni dei brevetti del Mark I sui suoi primi computer.

L'arrivo di Alan Turing a Manchester (vedere a pag. 200) nel 1948 dette un notevole impulso alle attività di programmazione. Nel 1950 Turing pubblicò il primo manuale di programmazione per il Manchester. Due anni dopo, il gruppo di Manchester ebbe l'idea di costruire un computer più compatto ed economico. I loro piani di lavoro subirono un'accelerazione grazie all'invenzione del transistor e nel novembre del 1953 divenne operativo, a Manchester, il primo computer a transistor del mondo.

Alla fine degli anni '50, l'America passò al primo posto nella tecnologia dei computer ed il governo inglese decise di lanciare un nuovo progetto, per la riconquista del primato: il computer Atlas, costruito sotto la direzione di Tom Kilburn. Commissionato nel dicembre del 1962, l'Atlas usava parole a 48 bit con un singolo formato d'indirizzo, una memoria centrale di 16 Kbyte ed una ROM a tamburo magnetico di 8 Kbyte. Ne furono venduti modelli al Gruppo di Ricerca per l'Energia Atomica di Harwell ed alla British Petroleum e, per molti anni, l'Atlas fu considerato il computer più avanzato tra quelli esistenti.

Il Manchester Mark I

Il Manchester Mark I detiene, a buon diritto, il titolo di "primo computer programmabile del mondo", poiché, nel 1948, su di esso venne fatto "girare" il primo programma di calcolo per computer. La Ferranti, allora un'industria di modeste dimensioni, venne incaricata di realizzare una versione commerciale del Mark I, che fu immessa sul mercato agli inizi del 1951



per concessione della UNIVERSITY OF MANCHESTER



REX FEATURES LTD

Una giornata alle corse
Le corse ippiche possono essere un interessante campo d'applicazione per gli appassionati sia di cavalli che di home computer, ma pare che, per il momento, nessuno sia riuscito a formulare un programma in grado di assicurare laute vincite!

Rischio calcolato

I computer nelle lotterie, negli stadi e negli ippodromi

Il gioco d'azzardo è essenzialmente una questione di probabilità, anche se molti giocatori preferirebbero dire che soprattutto si tratta di vincere! Questa seconda affermazione è, naturalmente, infondata poiché invece molti giocatori perdono cifre anche consistenti ed il gioco può diventare un vizio. In realtà, nei giochi quali la roulette, il lotto ed il totocalcio, le probabilità di vittoria favoriscono maggiormente il "banco" che non il giocatore.
Per valutare se i computer possono in qualche modo modificare questo stato di cose, occorre innanzitutto esaminare a fondo quali siano queste probabilità.
Spogliati delle loro attrattive esteriori, tutti i giochi affidati al caso finiscono col dipendere dal verificarsi o meno di un evento, prodotto da un qualche strumento governato dal caso (come, ad esempio, una pallina lanciata all'interno di una roulette o una carta estratta da un mazzo ben mescolato). Se i parametri sono noti (per esempio, il numero delle carte), la teoria delle probabilità consente di formulare alcune previsioni.
Facciamo un esempio: la ruota delle roulette possiede 37 "buche", numerate da 0 a 36. Vi sono quindi 18 numeri pari e 18 dispari, più lo zero. Le probabilità che la pallina cada in una buca dispari si può esprimere come 18/37, ossia 0,4864864 o, in percentuale, il 48,6% circa. Questo valore è leggermente inferiore a quello del lancio di una moneta (testa o croce), che risulta del 50%. La differenza, provocata dalla presenza della buca 0, costituisce la "fetta" o margine di profitto della casa da gioco.
Nonostante vi sia chi afferma il contrario, un computer non può influenzare affatto le probabilità di base di un particolare gioco. Addirittura, il professor Hans Sagan, una voce autorevole nel campo delle probabilità, ha calcolato che, a lungo andare, è impossibile vincere in un qualunque gioco da casinò, escluso forse il "blackjack".
Queste affermazioni, fondate su analisi teoriche, non hanno tuttavia scoraggiato gli appassionati inventori di "sistemi". Oggi, ai possessori di personal computer, viene offerta tutta una gamma di sistemi computerizzati, alcuni dei quali danno persino l'impressione di funzionare. I sistemi, per i giochi da casinò, sono quasi sempre basati sul principio del "raddoppio ripetuto", un metodo che implica, però, considerevoli disponibilità di denaro per aver successo. Ma esiste anche un altro problema: per quanto tecnicamente

Espedienti per il totocalcio

Ci sono alcuni pacchetti software, anche per home computer, il cui obiettivo è quello di facilitare le vincite al totocalcio. I migliori fanno uso di una vasta gamma di informazioni sui precedenti incontri e possono avere una qualche possibilità di successo nel prevedere il risultato di alcuni incontri. Come per tutti i tipi di gioco assistiti da computer, questi programmi aumentano in modo molto marginale le possibilità di vittoria dei giocatori. Per questa ragione i fabbricanti, all'atto della vendita, non si assumono alcuna responsabilità in caso di risultati poco soddisfacenti per il giocatore



non sia illegale, la maggior parte dei casinò vieta di usare calcolatori di quasiasi genere nelle proprie sale. I computer sono probabilmente di maggior aiuto in quei giochi che richiedono una certa dose di abilità o di strategia e possono essere impiegati per suggerire al giocatore una particolare linea di condotta o per aiutare la sua memoria. Tuttavia, il valore di una simile assistenza tende ad essere inversamente proporzionale all'abilità del giocatore.

Purtroppo, nelle più diffuse forme di gioco, le capacità del giocatore contano poco o nulla; il totocalcio ne offre un esempio significativo. Il più sofisticato programma di previsione dei risultati è il celebre "F4 Football Forecast" del Professor Frank George, che è disponibile in versioni utilizzabili sui più comuni home computer. Basato su un'analisi statistica dei risultati di 10 anni, il programma attribuisce un valore alla media dei risultati di ciascuna squadra. Il sistema, per prevedere il possibile risultato di un incontro, tiene conto dei risultati più lontani nel tempo (dedotti dalle tabelle pubblicate dalla Lega del Calcio), di quelli più recenti e del risultato dell'ultima partita giocata.

Se prendiamo in esame, ad esempio, una partita Juventus-Catania, il risultato potrebbe apparire scontato, ma nel caso di due squadre più equilibrate il computer talvolta stupisce per l'esattezza delle sue previsioni. Ciò non significa, comunque, che il sistema garantisca una vincita sicura. L'analisi statistica dimostra che, in teoria, l'uso del programma del professor George accresce di circa tre volte le probabilità di vittoria, ma solo in teoria.

Anche usando questi sistemi le probabilità restano dunque sfavorevoli. In Gran Bretagna, dove gli home computer sono diffusissimi, anche tra i giocatori del totocalcio, non risulta che nessuna delle maggiori vincite sia stata ottenuta mediante l'impiego di questi sistemi computerizzati. «Se ci fosse un sistema veramente funzionante, noi lo sapremmo, ma... non esiste!», afferma l'addetto stampa della Littlewoods, una delle società che gestiscono, in Gran Bretagna, il gioco delle schedine. Anche se la compilazione dei pronostici viene effettuata da speciali macchine automatiche, come avviene anche in Italia, la Littlewoods usa i computer solamente per scopi amministrativi.

Le corse dei cavalli sembrano offrire agli appassionati di computer e di gioco qualche opportunità in più. Uno studente di Darlington (GB) ha sviluppato un programma per computer con lo scopo di predire i cavalli vincenti. Scritto originariamente per lo ZX81 e successivamente adattato per lo Spectrum, il programma di David Stewart ha dato con una certa frequenza previsioni vincenti. David ricava le sue informazioni da alcune stazioni radio locali della BBC ma, personalmente, non ha accumulato una grande fortuna.

È certamente significativo che quasi tutti gli scommettitori "professionisti" sulle corse di cavalli abbiano disdegnato l'uso dei computer. L'"handicap" ufficiale viene ancora conteggiato manualmente dal Jockey Club (anche se i dati vengono registrati su computer). Anche il Timeform, la «bibbia» dei frequentatori degli ippodromi, redige a mano la maggior parte dei dati. Vengono soltanto calcolati i tempi medi standard per ciascun cavallo, tenendo conto dell'incidenza del vento.

I computer vengono, invece, usati con crescente frequenza nel campo delle scommesse, anche se in questo caso non sono adoperati per calcolare le probabilità. Nel Regno Unito, il personale delle famose organizzazioni di "bookmaker" calcola le vincite realizzate sulle scommesse con l'ausilio di speciali elaboratori. Uno scommettitore, che abbia un "conto corrente" presso una di queste catene di "bookmaker", può semplicemente telefonare la sua scommessa direttamente al centro computerizzato. Nel computer vengono inserite alcune informazioni riguardanti la scommessa effettuata e sul conto viene addebitata la quota della scommessa. In caso di vincita del cavallo, la somma viene calcolata ed accreditata automaticamente sul conto del giocatore.

Molti bookmaker sono assai scettici nei confronti dei computer e ritengono che nessuno abbia ancora sviluppato dei sistemi "sicuri", computerizzati o meno. Tuttavia, è stata proprio una nota agenzia di scommesse che ha mostrato al pubblico una delle più straordinarie e controverse simulazioni per compu-

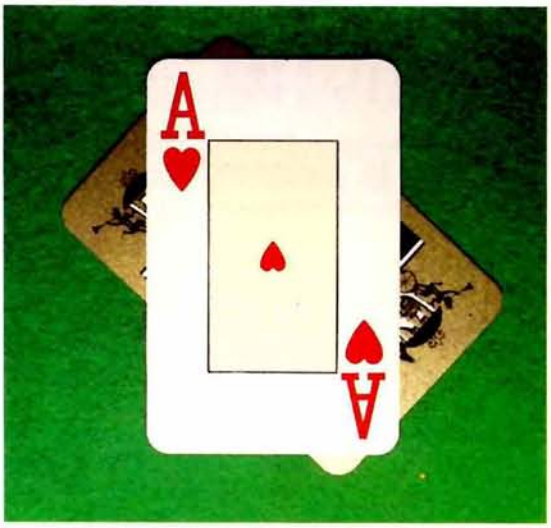
Azione e... rotazione

Nella roulette, il numero 0 consente al "banco" di avere un più ampio margine di probabilità rispetto ai giocatori. Non esiste alcun metodo per migliorare le probabilità di giochi di questo tipo, perciò i programmi sviluppati in questo campo mirano soprattutto a sistemi di analisi sulle scommesse



DAVID W. HAMILTON

ter di tutti i tempi. Le prestazioni dei cavalli che hanno vinto negli ultimi anni il classico "Derby" sono state inserite in un programma appositamente commissionato. I lettori di giornali specializzati furono invitati a fare previsioni su quale sarebbe stato l'ordine di arrivo dei primi sei cavalli. Sorse una disputa sul piazzamento del famoso cavallo italiano Ribot, che non aveva mai perso una corsa. Il computer gli assegnò il quarto posto!
Il più famoso computer, impiegato per una lotteria, si trova, ancora una volta, in Gran Bretagna, e si chiama ERNIE ossia "Ernesto", sebbene la sigla derivi da Electronic Random Number Indicator Equipment, ovvero "apparecchio elettronico per l'estrazione di numeri casuali". Questo computer estrae a sorte, periodicamente, un gruppo di numeri di serie dei "Premium bond", una specie di "certificati di deposito". Questa macchina fu costruita dalla Plessey nel 1973, in sostituzione del computer precedente, fabbricato nel 1957. Il suo compito è quello di generare a caso circa 200.000 numeri e di registrarli su un nastro magnetico. Questi numeri vengono generati entro un arco di valori che inizia dal più basso numero di serie mai emesso e finisce col più alto. Il nastro viene quindi caricato su un'unità elaborativa della ICL per effettuare un confronto con i numeri già estratti. Eliminati questi ultimi, il computer provvede a stampare i mandati di premio e le lettere per i vincitori.
Da quando sono stati commissionati, i due ERNIE hanno generato numeri per 22,2 milioni di premi, per un valore totale di quasi due miliardi di sterline. Una cifra colossale, ma la probabilità di estrazione di un certificato di "Premium bond" vincente, nell'arco di un mese, è pari a 1 su 15.000.



L'inganno e il fato

Il "Blackjack" è un gioco di carte disponibile anche in versione computerizzata. La capacità di memorizzare le carte già giocate accresce le possibilità di vittoria di un giocatore e, poiché i casinò non permettono di usare computer ai loro tavoli da gioco, sono rimasti celebri i fantasiosi trucchi escogitati per sfuggire a questa regola. C'è stato chi, nonostante i controlli, è riuscito ad introdurre una piccola calcolatrice, ma anche chi ha tentato di usare piccoli ricetrasmettitori collegati ad un computer situato al di fuori del casinò

UN LANCIO DI DADI

L'ingrediente fondamentale del gioco, cioè la generazione casuale di numeri, può essere facilmente simulata sul personal computer.
Molte versioni BASIC sono provviste di una funzione adatta a questo scopo. In molti casi, però, i numeri prodotti non sono realmente casuali, come dimostrato dal seguente breve programma:

```
10 LET A = RND
20 LET B = RND
30 LET C = RND
40 PRINT A, B, C
```

Nelle prime tre linee vengono assegnati, alle variabili A, B e C, altrettanti numeri apparentemente casuali. Questi numeri poi vengono stampati. I risultati potrebbero essere i seguenti (possono essere diversi a seconda del computer in uso):

.014007 .964370 .457397

Ma se facciamo girare nuovamente il programma, la maggior parte dei microcomputer produrrà nuovamente la stessa sequenza. Infatti, quando usiamo la funzione RND, il computer risponde producendo il numero "successivo" di una sequenza fissa di numeri. Tipicamente, questa sequenza comprende il milione di numeri frazionari a sei cifre che vanno da 0,000000 a 0,999999; ciascuno di questi valori si presenta una sola volta all'interno di un ciclo completo ma, naturalmente, non in ordine crescente o decrescente.

Alcuni BASIC usano una sintassi leggermente diversa, che richiede la presenza di una espressione tra parentesi, chiamata "argomento". L'istruzione avrà questa forma: LET A = RND (X). Il risultato è assai simile: tanto RND che RND (X) possono essere usate nello stesso modo delle altre variabili.
Alcuni tipi di BASIC forniscono anche una funzione RANDOMIZE, per far partire la sequenza di numeri casuali da un punto non prevedibile. L'inserimento del comando RANDOMIZE all'inizio di tutti i programmi nei quali verrà usata l'istruzione RND, assicurerà la produzione di una sequenza diversa tutte le volte che il programma sarà fatto funzionare.
Per simulare il lancio di un dado, occorrono numeri interi compresi tra 1 e 6. Pertanto, è necessario eliminare i valori frazionari e ciò si ottiene usando la funzione INT; ad esempio, PRINT INT (6,99) visualizza il valore 6, così come avviene con PRINT INT (6,01). Tutte le cifre presenti dopo il punto decimale vengono eliminate.
Dal momento che il numero più grande che RND può generare è 0,999999 (che, espresso in forma "intera", assume il valore 0), è necessaria una piccola moltiplicazione. La formula, ormai divenuta uno standard, è:

```
LET A = INT (6 * RND)+1
```

La moltiplicazione è per 6 perché un dado ha sei facce. Il +1 è semplicemente per avere la certezza che i risultati vadano da 1 a 6, e non da 0 a 5.

Qualche passo in Assembly

Continua la panoramica sul linguaggio macchina

Qualsiasi dato immagazzinato nella memoria del computer corrisponde ad un numero binario a 8 bit. Tuttavia, usando questo formato nella stesura di un programma, il risultato sarebbe difficile da interpretare o da ricordare. Inoltre, non mancherebbero errori di battitura o di trascrizione. Per questa ragione, si ricorre all'impiego dei numeri esadecimali che consentono di rappresentare il contenuto di ciascun byte con un numero di due sole cifre. Inoltre, un qualsiasi indirizzo compreso nella memoria del computer (da 0 a 65535, in decimale) può essere rappresentato con sole quattro cifre.

Per convenzione, i numeri esadecimali vengono preceduti, nei listati, dal simbolo $, per evitare confusione con la notazione decimale. Naturalmente, tale simbolo serve solo al programmatore e non deve essere immesso, come facente parte del codice, nella memoria del computer. Inoltre, quando un operando di una delle istruzioni è composto da due byte (ad esempio: LDA $3F80), questi vengono inseriti nella memoria della macchina in ordine inverso, ossia prima il byte meno significativo (80 nell'esempio), poi quello più significativo (3F nell'esempio). Riassumendo, nell'esempio precedente i tre byte saranno: AD (la rappresentazione esadecimale dell'operazione LDA in ASSEMBLY 6502) seguito da 80 ed infine da 3F. Questa procedura facilita il compito del microprocessore, ma può generare confusione nel programmatore alle prime armi.

I programmi in codice macchina vengono spesso visualizzati o stampati sotto forma di "hex dump", ossia di un tabulato in codice esadecimale. Per inserire un programma nella memoria del computer, occorre specificare l'indirizzo di partenza ed il contenuto da depositare nella cella che ha tale indirizzo. L'indirizzo viene incrementato (automaticamente in certi programmi) e si provvede a depositare il successivo codice. L'operazione si ripete fino ad esaurimento dei codici del programma. L'operazione di caricamento nella memoria del computer può essere eseguita, in BASIC, per mezzo del comando POKE. Supponendo che l'indirizzo di partenza sia $1000 (4096 in decimale), un hex dump avrebbe il seguente aspetto:

(Indirizzo)	(Contenuto)
1000 AD	(173 in decimale)
1001 80	(128 in decimale)
1002 3F	(63 in decimale)

allora il programma può essere caricato in memoria usando le seguenti istruzioni BASIC:

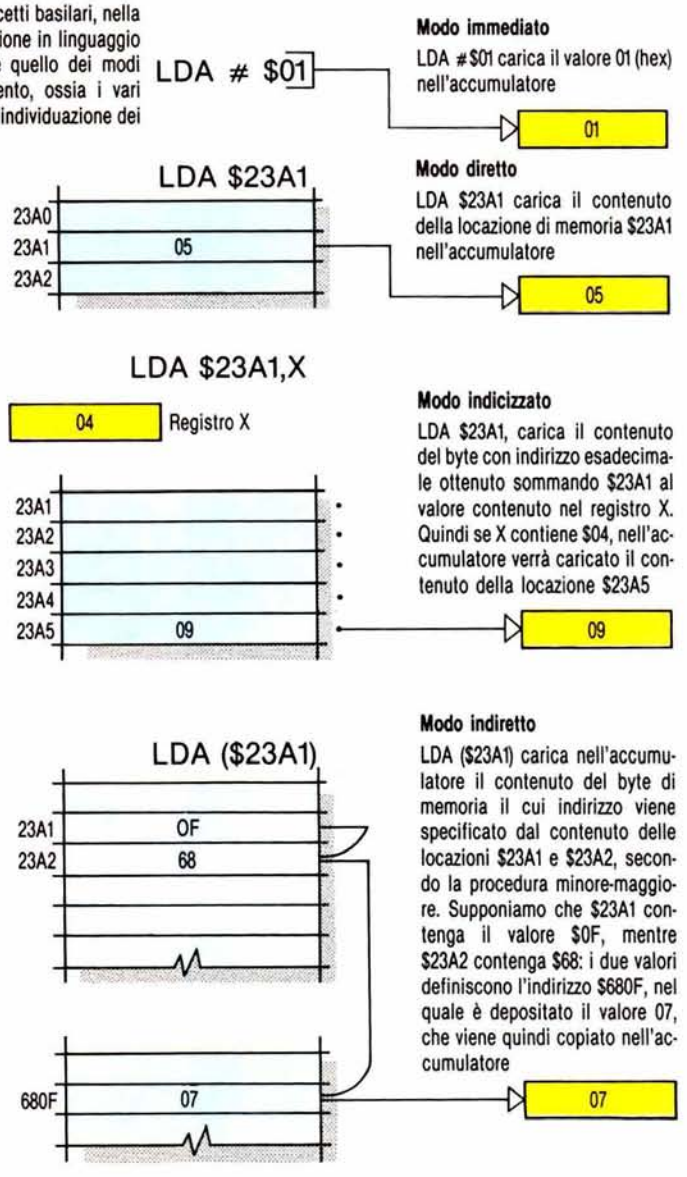
```
POKE 4096,173
POKE 4097,128
POKE 4098,63
```

È da notare che tutti i valori sono stati convertiti dalla notazione esadecimale a quella decimale prima di essere usati nelle istruzioni POKE: è l'interprete che provvede a convertirli nel formato binario.

Molti microcomputer sono dotati di alcuni comandi BASIC che permettono una tempo-

MODI D'INDIRIZZAMENTO

Uno dei concetti basilari, nella programmazione in linguaggio macchina, è quello dei modi d'indirizzamento, ossia i vari metodi per l'individuazione dei dati



UN LINGUAGGIO ABBREVIATO

Un programma in codice macchina può essere scritto in diverse forme. Di solito viene scritto dal programmatore in forma di linguaggio ASSEMBLER, che fa uso di codici mnemonici per gli opcode e di etichette per gli operandi, ad esempio:

```
LDA  PESO
ADC  BENZINA
STA  PESO
```

Occorre, comunque, specificare gli indirizzi di queste etichette, ossia:

```
BENZINA = $03EE
PESO = $031F
```

Un assemblatore trasforma queste istruzioni e valori in codice esadecimale (o binario), successivamente eseguibile. Qualora non si disponga di un assembler, ma di un programma monitor sufficientemente sofisticato, è possibile usare i codici mnemonici, ma non le cosiddette "etichette", e l'esempio precedente va scritto:

```
LDA  $031F
ADC  $03EE
STA  $031F
```

Un hex dump si compone di un indirizzo di partenza (posto, in genere, a sinistra) e di una sequenza di valori, espressi in forma esadecimale, del contenuto delle locazioni di memoria. Si noti che gli operandi di due byte, come $031F, vengono memorizzati in ordine inverso (1F 03) e che gli opcode vengono sostituiti dai corrispondenti valori esadecimali:

```
19C4 AD 1F 03 6D EE 03 8D 1F 03
```

ranea interruzione del programma BASIC in corso, per passare l'esecuzione ad un programma in linguaggio macchina, come se si trattasse di una normale subroutine. Un comando di questo tipo è SYS 4096 che provoca l'avvio di un programma in codice macchina sistemato a partire dalla locazione di memoria decimale 4096. Un altro esempio è CALL $E651, col quale viene "chiamata" la subroutine in codice macchina memorizzata dalla locazione esadecimale $E651 in poi.

Il programma, o subroutine, in codice macchina eseguirà quindi le proprie operazioni (producendo risultati più o meno visibili, a seconda delle funzioni richieste) e poi, se correttamente codificata, provvederà a restituire il controllo al programma BASIC. Nella programmazione in codice macchina gli errori sono ancor più fastidiosi che negli altri linguaggi. Infatti, il computer non visualizza nemmeno la tanto odiata, ma utile, frase SYNTAX ERROR. Nella maggior parte dei casi la macchina si blocca completamente, restando muta ad ogni richiesta di chiarimenti da parte del programmatore. Normalmente, ciò non danneggia in alcun modo l'apparecchio, ma il programmatore è costretto a premere il pulsante di reset (oppure a spegnere e riaccendere la macchina) e quindi a reinserire nuovamente tutto il programma. È questo il motivo per cui non è possibile fare prove di funzionamento del programma in codice macchina come si è soliti fare in BASIC: lo sviluppo del programma deve essere interamente verificato sulla carta prima di inserire il listato nella memoria del computer.

Tuttavia, esiste uno strumento software di grande aiuto nella verifica e nell'inserimento del codice macchina: il cosiddetto "machine code monitor", ossia "programma di controllo per codice macchina", spesso chiamato semplicemente "monitor" (da non confondersi con l'omonimo termine che indica però l'apparecchio di visualizzazione). Questo programma è talvolta inserito nella ROM del computer, ma, dato che esistono in commercio delle versioni molto sofisticate, spesso viene acquistato separatamente, su cassetta o cartuccia. I programmi monitor permettono di visualizzare e di modificare, direttamente sullo schermo, il contenuto di una qualsiasi area di memoria, oltre ad offrire una più o meno vasta gamma di funzioni accessorie, tra le quali il trasferimento diretto dei programmi tra la RAM e la memoria di massa, senza dover ricorrere al BASIC. I più progrediti programmi di utilità, scritti in codice macchina, sono anche in grado di mostrare il contenuto di ciascun registro interno del microprocessore.

Gli hex dump, comunque, sono strumenti piuttosto rudimentali: esistono, invece, delle versioni di monitor che consentono di usare anche il linguaggio mnemonico. È il caso dello "spot assembler", ossia di assemblaggio "immediato", che permette di trasformare immediatamente i codici mnemonici in valori numerici senza dover ricorrere ad un programma compilatore vero e proprio.

Arriviamo così a parlare dei programmi ASSEMBLER, nei quali non solo vengono impiegati codici mnemonici per rappresentare gli opcode, ma è anche possibile assegnare nomi (o etichette) sia agli operandi che agli indirizzi utilizzati nel programma da "assemblare". Ad esempio, se la locazione $07B2 contiene il numero corrente di missili sparati durante un gioco, noi possiamo "caricare" questo valore nell'accumulatore usando l'istruzione:

```
LDA MISSIL
```

All'inizio del programma dovremo ovviamente specificare che la locazione di MISSIL vale $07B2 ed, inoltre, indicare che inizialmente questa locazione deve contenere il valore $09 (cioè 9 missili).

Una volta terminata la stesura del listato "sorgente" del programma si usa l'assemblatore, che opera automaticamente la conversione in codice esadecimale (o, per meglio dire, binario) di tutto il programma, creandone cioè una versione eseguibile, chiamata codice o programma "oggetto". Questa procedura è molto simile alla "compilazione" (vedere a pag. 84) ma, nel caso ora esaminato, c'è una corrispondenza diretta (cioè uno a uno) tra il codice sorgente e quello oggetto. L'ASSEMBLY, che è un linguaggio di livello superiore rispetto al codice macchina, si dimostra notevolmente più facile da scrivere ma, in pratica, produce gli stessi risultati.

Poiché ogni tipo di CPU possiede un proprio insieme di codici, esistono ASSEMBLER specifici per ognuna di esse.

GLI OPCODE

Vengono qui presentati altri opcode (codici mnemonici) generalmente usati nei microprocessori

JSR — **Jump SubRoutine** (Salto al sottoprogramma)

Questa funzione è equivalente al GOSUB del BASIC. JSR $354D cambia il contenuto del registro contatore di programma (PC) e passa ad eseguire le istruzioni a partire dalla locazione $354D

RTS — **ReTurn from Subroutine** (Ritorno dal sottoprogramma)

Quando il processore incontra l'istruzione RTS, riprende l'esecuzione del programma dalla locazione successiva a quella che contiene la chiamata al sottoprogramma (equivale cioè al RETURN del BASIC). RTS non ha operandi poiché l'indirizzo di ritorno è automaticamente immagazzinato in una speciale area di memoria chiamata Stack

BMI — **Branch if MInus** (Salta se negativo)

È una delle varie forme di salto condizionale in codice macchina (in BASIC equivale alla forma IF...THEN GOTO). Se il risultato dell'ultima operazione produce un valore negativo nell'accumulatore, l'esecuzione del programma "salta" all'indirizzo specificato. BPL significa Branch if PLus (Salta se positivo)

LDX — **LoaD X register** (Carica il registro X)

X è un altro registro interno al microprocessore, ma su di esso non si possono eseguire operazioni aritmetiche. Può essere usato, invece, per l'"indirizzamento indicizzato" (vedere figura a pag. 464). LDX carica un valore in X, mentre STX, STore X (Conserva X), deposita il contenuto di X in memoria

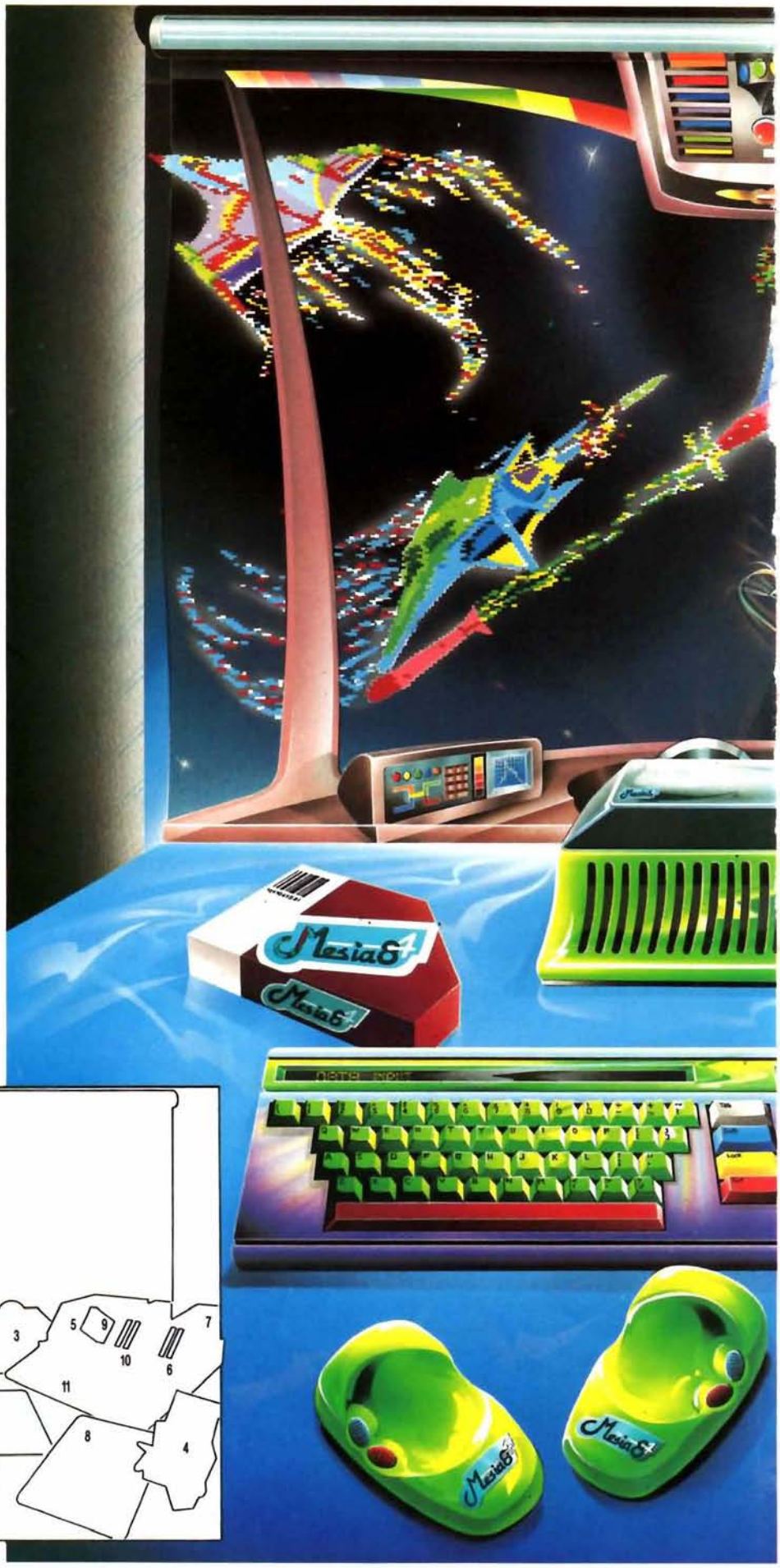
INX — **INcrement X** (Incrementa X)

Questa istruzione serve per aumentare di 1 il valore contenuto nel registro X (DEX, cioè DEcrement X, invece, diminuisce di 1 il valore di X). Questa operazione è utile nell'indirizzamento indicizzato, consentendo di accedere, in modo molto semplice, a locazioni di memoria adiacenti

Il Futuro

Home computer: passato e futuro a confronto

Quale aspetto avranno gli home computer delle prossime generazioni e, soprattutto, come funzioneranno? Tentiamo qui di dare alcune risposte, esaminando i principali elementi di una ipotetica macchina del domani. Alcune ipotesi si basano sulle tecnologie che solo oggi cominciano ad affacciarsi sul mercato (sia nel campo informatico che in altri settori), mentre altre illustrano quelli che noi riteniamo gli sviluppi futuri.
Una delle caratteristiche fondamentali su cui poggiano le nostre ipotesi è il concetto di "modularità". Dopo aver acquistato un'unità base di elaborazione, l'utente potrà disporre di una vasta gamma di scelte per espanderla. Potrà quindi, in pratica, progettare il proprio sistema di elaborazione scegliendo, ad esempio, i moduli per la grafica e le facilitazioni per la creazione di suoni che più si confanno ai suoi desideri. Di una cosa possiamo essere certi: nei prossimi anni l'evoluzione tecnologica produrrà cambiamenti sempre più rapidi nel mercato dei computer.



1. Display su tastiera

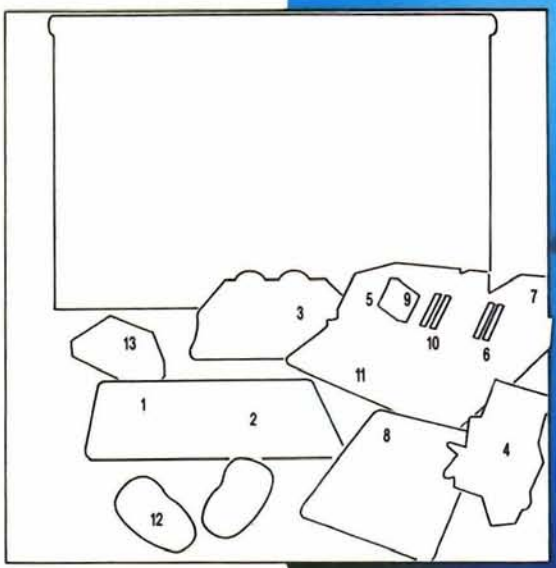
Le capacità dei processori a 32 bit permetteranno di visualizzare le informazioni simultaneamente ed in varie forme. Lo schermo principale potrà mostrare, ad esempio, le immagini viste dal sedile di comando di un'astronave, mentre su di un display sussidiario, posto nella parte superiore della tastiera, appariranno le informazioni di controllo tipiche di una cabina di pilotaggio

3. Monitor

All'inizio del 1980 vennero introdotti i primi proiettori televisivi, ma i loro sbocchi commerciali risultavano limitati dallo scarso potere di emissione luminosa del tubo a raggi catodici. Gli sviluppi della tecnologia CRT stanno rapidamente portando a sistemi di proiezione a schermo panoramico

2. Tastiera

Malgrado la tradizionale scarsa funzionalità della tastiera di tipo QWERTY, è assai improbabile che vengano fatti seri tentativi di introdurre tastiere di tipo alternativo. I tasti a molla del tipo macchina da scrivere sono di gran lunga i più popolari anche se le nuove tastiere ad effetto Hall, che impiegano magneti al posto delle molle, stanno divenendo di uso sempre più comune. Gli stessi interruttori elettronici potrebbero essere vantaggiosamente sostituiti da un sistema basato su tasti che, quando vengono premuti, vanno ad interrompere una matrice composta da raggi laser





TONY LODGE

4. Processori alternativi

Oltre al processore principale a 32 bit, è probabile che il micro degli anni '90 ospiterà processori addizionali sotto forma di moduli ad innesto. Alcune procedure di elaborazione potranno essere "subappaltate" dal processore principale al sub-processore più adatto per quella funzione. In alternativa, moduli ad innesto di basso costo potranno svolgere le funzioni tipiche del computer degli anni '80, in modo da utilizzare software proveniente da un qualsiasi altro computer

5. Memoria ed Accesso Casuale

Il processore a 32 bit può indirizzare fino ad un massimo di 4 miliardi e 300 milioni di locazioni di memoria, quantità ben superiore al limite di 65536 byte, imposto dai microprocessori ad 8 bit comunemente usati nei microcomputer domestici

6. Comunicazioni

Nel 1990 le antenne a riflettore parabolico per la ricezione di segnali emessi da satelliti, saranno di uso comune e la maggior parte delle linee telefoniche si baseranno su segnali digitali e non più su quelli analogici come avviene oggi. Speciali unità di controllo delle comunicazioni svolgeranno alcune funzioni oggi affidate ai modem (modulatori/demodulatori)

7. Alimentazione

La crescente varietà di apparecchi che verranno collegati al microcomputer porterà verosimilmente ad una richiesta di corrente d'alimentazione notevolmente superiore rispetto a quella attuale. Verranno incorporati, negli apparecchi, filtri di livellamento e batterie ricaricabili di riserva, affinché le fluttuazioni della linea d'alimentazione o le cadute di corrente non producano perdite o danneggiamenti dei dati

8. Schermo portatile

Gli sviluppi tecnologici nel campo degli schermi piatti, probabilmente, consentiranno l'uso di matrici a cristalli liquidi a risposta veloce. I display saranno pilotati, forse, mediante raggi infrarossi (oppure microonde). Addirittura, questi dispositivi potrebbero essere usati come una tastiera "a sfioramento", per eseguire scelte da un menu, o come "bit pad" o tavolette grafiche

9. CDROM

I dischi a tecnologia laser sostituiranno le attuali memorie ROM. La quantità delle informazioni immagazzinabili, codificate mediante un procedimento ottico, supera di gran lunga le attuali capacità dei dischetti. Una tipica CDROM potrà contenere oltre 4 megabyte

10. Dischetti "floppy"

Entro la fine degli anni '80 i dischetti si saranno evoluti ad un livello tale da competere su un piano di parità con le unità a disco fisso Winchester, sia per la velocità che per la densità di registrazione dei dati. Al tempo stesso, le dimensioni dei dischetti dovrebbero ridursi fino ad un diametro inferiore ai 3 pollici degli attuali micro dischetti

11. Pannello frontale

Sui primi computer, precedenti allo sviluppo dei linguaggi di alto livello e delle tastiere, i programmi dovevano essere immessi nel computer in notazione binaria per mezzo di un pannello frontale, dotato di una serie di lampadine ed interruttori: questo pannello permetteva all'utente di controllare ciascun bit degli indirizzi, dei dati e dei canali di controllo. Per coloro che programmano con entusiasmo in codice macchina, un pannello frontale può ancora essere uno strumento utile e potrebbe tornare di moda

12. "Topi" all'infrarosso

Il computer PC-Junior della IBM fa già uso di raggi infrarossi per trasferire dati dalla tastiera al computer, eliminando la necessità di collegamenti con cavi elettrici. Questa tecnologia potrebbe essere utilizzata per collegare tra loro tutte le periferiche, inclusi i topi, eliminando quindi l'"effetto spaghetti" (= groviglio di fili)

13. Microprocessori a 32 bit

I primi home computer basati su microprocessori a 32 bit sono apparsi nel 1983, ma sono stati costretti ad utilizzare canali di trasferimento dati a 16 o anche ad 8 bit, per mantenere compatibilità con le memorie e le periferiche esistenti. Con l'introduzione di componenti come il chip Motorola 68032, che offre un'elaborazione ed un trasferimento dati a 32 bit, la capacità e la velocità di elaborazione aumenteranno notevolmente

Salti generazionali

La tecnologia VSLI dà inizio alla Quarta Generazione, ma in Giappone si pensa già alla Quinta!

"L'uomo vecchio non fa rivoluzioni" dice un proverbio e la persona a capo del progetto nipponico, per la creazione della Quinta Generazione di computer, sembra aver preso alla lettera questo detto. Nella scelta di 40 collaboratori, tutti scienziati provenienti dalle 10 maggiori industrie e laboratori di ricerca del suo paese, il Dottor Kazuhiro Fuchi ha selezionato solo individui con età inferiore ai 35 anni. L'"Institute for New Generation Computer Technology", a Tokio, è stato fondato nel 1972 e gode di un finanziamento di circa 700 miliardi di lire da spendersi in un arco di 10 anni. Esso nasce da una cooperazione tra governo e industria privata. Compagnie come la Fujitsu, la Sharp e la Toshiba partecipano a questo ambizioso progetto, che si propone di compiere un salto tecnologico, rispetto alla attuale tecnologia dei computer e di creare macchine molto più progredite di quelle che vengono attualmente progettate altrove.

Il termine stesso Quinta Generazione mette perfettamente a fuoco l'evoluzione nel progetto dei computer e fa pensare agli ancor maggiori progressi che si avranno in futuro. L'obiettivo principale della ricerca scientifica si è spostato, negli ultimi cento anni, dall'imbrigliamento dell'energia naturale (sotto varia forma, come l'elettricità ed i motori a combustione interna) allo studio della più impalpabile forma di ricchezza, l'informazione. La terra, il lavoro, il capitale e l'industria possono essere state, in passato, le sorgenti del potere, ma il futuro favorirà quei gruppi che avranno il controllo dell'informazione.

Questa nuova società avrà bisogno di un «motore», ossia di una macchina capace di ragionare automaticamente, che possa essere applicata ad ogni problema reale e ad ogni campo della ricerca umana e che offra la precisione matematica e l'affidabilità operativa di un computer. La macchina sulla quale oggi stanno lavorando in Giappone è chiamata Sistema per l'Elaborazione dell'Informazione e della Conoscenza (la sigla inglese è KIPS, che sta per Knowledge and Information Processing System).

Il corpo umano è una macchina capace di trasformare in modo perfetto le percezioni sensorie in conoscenze (ad esempio, la situazione di gioco in una partita a scacchi può essere afferrata con una sola occhiata), ma, quando si tratta di prendere decisioni che dipendono da una grande quantità di parametri, allora ci accorgiamo di quanto siano grandi i nostri limiti. Le regole degli scacchi possono essere spiegate in pochi minuti, ma lo sviluppo del gioco è talmente complesso che i grandi maestri riescono a prevedere appena una dozzina di mosse. Tuttavia, un qualsiasi problema, anche il più complesso, può venir suddiviso in una serie di passaggi più elementari: su ciascun passaggio è quindi possibile prendere una decisione applicando il metodo della deduzione. Le regole che governano questo sistema vanno sotto il nome di "logica dei predicati". Le regole logiche della deduzione vengono da noi applicate alla soluzione di tutti i problemi, anche se questo modo di procedere è spesso inconscio.

Per uno studioso essere dotato di buon cervello è condizione necessaria, ma non sufficiente: ad un dottore, per esempio, occorrono molti anni di esperienza per poter accumulare solide conoscenze mediche. Allo stesso modo, un KIPS deve essere dotato di una vasta banca dati, sulla quale applicare le regole della deduzione. Inoltre, il sistema deve essere di facile impiego, senza che si debba ricorrere ad un gruppo di esperti per farlo funzionare. Sarà la ricerca nel controverso campo dell'intelligenza artificiale (AI) a produrre macchine con le quali si potrà dialogare in un linguaggio "normale", scelto dalla controparte umana. Le materie che i Giapponesi hanno deciso di studiare ed approfondire abbracciano, dunque, una vasta area dell'informatica: hardware, software, interfacce, "expert system" e l'intelligenza artificiale.

Il progetto giapponese è stato concepito per esplorare settori che trascendono gli sviluppi delle attuali tecnologie. Con l'aumentare della densità dei transistor nei circuiti integrati, gli elettroni devono attraversare distanze sempre più brevi nel loro viaggio da un componente all'altro e quindi i circuiti operano più velocemente. Tuttavia, i giapponesi hanno compreso che la velocità, in sé, non è sufficiente: ecco perché si sono dedicati allo sviluppo del software. In un gioco di scacchi, ad esempio, c'è un numero così elevato di mosse possibili (circa 10^{120}) che il tempo stimato per l'esplorazione di tutte le combinazioni

LINGUAGGIO LOGICO

Il PROLOG ("Programmazione Logica") è stato sviluppato agli inizi degli anni '70 dal gruppo di ricerca per l'Intelligenza Artificiale dell'Università di Marsiglia, ma uno dei suoi teorici è l'americano Robert Kowalski. Basato su principi della logica umana, il PROLOG è il linguaggio che probabilmente sarà il più usato nei computer della Quinta Generazione. È anche assai utile per la creazione e l'interrogazione di database e per applicazioni educative

possibili supera il tempo di vita residuo del sole. Il progetto ha l'obiettivo di produrre una macchina che possa eseguire 100 milioni di deduzioni logiche (cioè possa applicare 100 milioni di regole) al secondo, ovvero, 100 milioni di LIPS (Logical Inferences Per Second, in italiano Deduzioni Logiche per Secondo).

Un'altra via, mediante la quale si potrebbe accrescere la velocità, è l'introduzione di funzioni software all'interno del progetto del chip, invece di caricarle in memoria e quindi elaborarle per mezzo di un chip per operazioni generiche. Questo progressivo assottigliarsi della distinzione tra hardware e software è uno degli aspetti più interessanti del progetto. Esistono già delle memorie "associative", in possesso di circuiti di ricerca logica costruiti all'interno delle stesse celle di memoria. Questi dispositivi sono in grado di localizzare un dato, basandosi esclusivamente sul suo significato intrinseco e non sul suo indirizzo in memoria.

Simili progressi aumenteranno la velocità di interazione tra il processore e le banche dati. L'incorporazione delle routine di programmazione all'interno del computer richiama alla memoria i primi elaboratori, come l'ENIAC, ma le macchine della Quinta Generazione si diversificheranno dall'architettura progettata da Von Neumann per un aspetto fondamentale: esse non avranno più una sola unità di elaborazione centrale, ma incorporeranno, invece, molti distinti processori, simultaneamente al lavoro. Questo tipo di realizzazione richiederà la massima cura nella temporizzazione e nel controllo delle operazioni interne, ma eliminerà i limiti imposti alla velocità dall'elaborazione sequenziale delle istruzioni. Il linguaggio interno scelto per i KIPS è il PROLOG, un linguaggio sviluppato in Francia ed in Inghilterra, basato sulla logica dei predicati. I KIPS saranno però in grado di comunicare con i loro utenti in molti altri linguaggi.

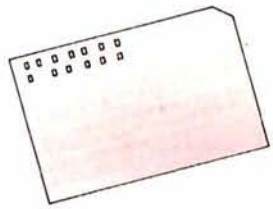
La traduzione simultanea del linguaggio umano è un altro ambizioso obiettivo del progetto: si mira ad un grado di precisione del 95%. Attualmente, i risultati ottenuti nel riconoscimento di parole, pronunciate da differenti voci, non sono certo paragonabili a quelli ottenuti per la sintesi vocale. Tuttavia, la NEC Corporation ha già sviluppato una macchina capace di comprendere un "parlato" continuo. Un limite di questa macchina è che essa può riconoscere le parole pronunciate da un solo individuo e che ciascuna parola deve essere precedentemente registrata per permettere al computer di ritrovare e quindi riconoscerne la configurazione.

Per quanto riguarda la parola scritta, il progetto sta preparando un dizionario Giapponese-Inglese di 100.000 vocaboli ed un programma che permetterà di effettuare traduzioni con una precisione del 90%.

I giapponesi hanno già realizzato progetti a lungo termine coronati da successo: il progetto PIPS (Pattern Information Processing System, ossia Sistema di Elaborazione per Gruppi di Informazioni) del 1970 si è dimostrato utile nello sviluppo delle banche dati per immagini e nelle interfacce di facile impiego. Un KIPS dovrà essere capace di esaminare un'immagine e di estrarne le caratteristiche salienti, in modo da farsene un'idea generale. Nella metropolitana di Tokyo viene già usata una macchina che è in grado di svolgere un lavoro simile: essa "scruta" i corridoi mediante delle telecamere e produce un modello corrispondente al flusso dei passeggeri.

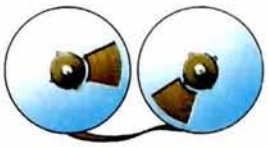
Nel 1983, la tecnologia dell'informazione ha rappresentato, nei soli Stati Uniti, un giro d'affari valutabile intorno agli 88 miliardi di dollari. Alla luce di tutto ciò, il progetto giapponese appare ancora più ambizioso, in quanto dovrà anche tener conto che il gigantesco impatto sociale, dovuto a computer "superintelligenti", dovrà essere diluito in un certo numero di "passaggi", accuratamente programmati, per non causare effetti catastrofici nell'economia, ma anche e soprattutto nella mentalità della gente.

SCONTRO FRA GENERAZIONI



Primo Round

La prima generazione di computer elettronici fu sviluppata attorno alla tecnologia della valvola termoionica. Questi computer avevano poca memoria "in linea" e i dati venivano di solito registrati su schede perforate



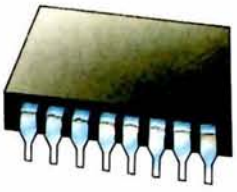
Secondo round

La seconda generazione si sviluppò con i transistor, che accrebbero le capacità di memoria, anche se la memorizzazione di tipo "off-line" (ossia nastri e dischi magnetici) era già in uso



Terzo round

La realizzazione dei circuiti integrati accrebbe massicciamente la potenza dei computer e fu grazie ad essi che nacquero i primi microcomputer, caratterizzati da unità a dischetti "floppy"



Quarto round

Ci stiamo oggi muovendo tra la terza e la quarta generazione, che sarà basata sulla tecnologia VLSI. La memoria RAM sarà così grande che la registrazione di tipo "off-line" perderà gran parte della sua importanza



Quinto round

La Quinta Generazione di computer, in corso di sviluppo soprattutto in Giappone, punta molto di più al software che non all'hardware; tuttavia, si basa sull'ipotesi che la memoria delle apparecchiature sarà talmente grande che la dimensione dei programmi cesserà di essere un problema

KEVIN JONES

Il 380Z della Research Machines

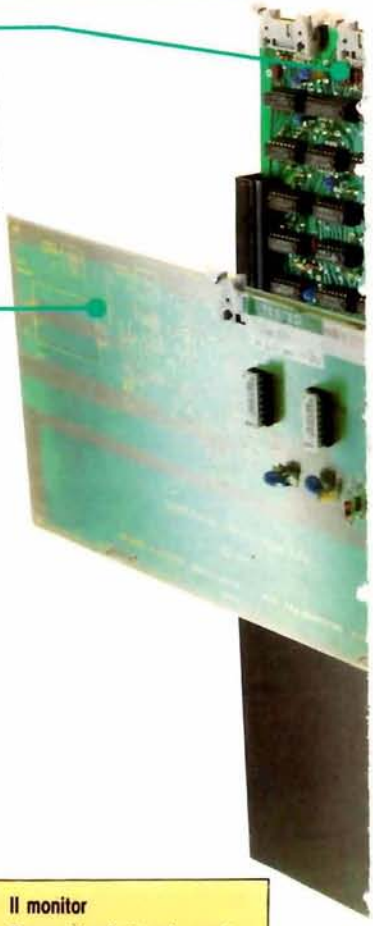
Anatomia di un microcomputer a diffusione europea: tecnologie a confronto



La tastiera

La tastiera che correda il 380Z è montata in un piccolo, ma robusto, involucro di metallo. I tasti sono disposti secondo i normali criteri e sono di buona qualità, offrendo una discreta comodità di battitura. I tasti sono stati progettati per resistere a lungo, fattore, questo, che rende la tastiera del 380Z una scelta ideale per le scuole

CHRIS STEVENS



Scheda di controllo per l'unità a dischetti

Oltre ad avere un chip specializzato per il controllo del dischetto, questa scheda è dotata di un chip di temporizzazione Z80 (CTC) e di un chip 8521 di Input/Output seriale che fornisce impressionanti capacità di comunicazione

Scheda terminale del bus

Questa scheda viene collocata all'estremità opposta del bus, rispetto alla scheda CPU e protegge le varie linee elettriche interne da eventuali interferenze esterne

I prodotti della Research Machines costituiscono una delle più longeve categorie di computer. Sebbene queste macchine non presentino particolari innovazioni tecnologiche e non abbiano un prezzo particolarmente competitivo, tuttavia vengono progettate e costruite in modo estremamente affidabile. Il computer di maggior successo prodotto da questa casa, il 380Z, ha un larghissimo campo d'applicazione nella didattica dei paesi d'oltremanica e probabilmente rappresenta, per molti ragazzi, il primo approccio con i problemi connessi all'elaborazione.

Paragonato alla maggior parte dei computer della stessa categoria, il 380Z sembra un gigante: le principali schede dei circuiti sono contenute in un robusto contenitore largo 48 cm., dotato di maniglie ai lati. Sollevando il coperchio di questa "scatola nera", il motivo delle sue grandi dimensioni appare ovvio: quasi un quarto dello spazio interno è occupato dall'apparato d'alimentazione. Quest'ultimo non è progettato secondo i più recenti criteri dell'alimentazione "a scatti": si tratta, invece, di un trasformatore a nucleo magnetico che presenta il vantaggio di essere completamente protetto da sovraccarichi di corrente o da guasti di altro genere.

Forse, proprio per questa sua affidabilità, il Ministero della Difesa britannico ha deciso di adottarne numerosi modelli per la gestione di alcuni propri magazzini. Inoltre, sempre in Gran Bretagna, questo computer è molto diffuso negli istituti scolastici nei quali vengono insegnate materie quali la fisica e le scienze. Infatti, la sua grafica ad alta risoluzione si dimostra assai utile per impieghi didattici.

Il software di grafica ad alta risoluzione, fornito in dotazione (la sigla inglese è HRG, da High Resolution Graphics) è composto da una serie di routine in codice macchina che vengono richiamate dal programma applicativo, consentendo di modificare le caratteristiche video, controllate dalla scheda HRG. Sebbene questa scheda sia stata progettata parecchi anni fa, essa costituisce ancor oggi uno dei migliori sistemi disponibili. Modificando il contenuto di alcune locazioni di memoria, si possono produrre immagini a colori in varie risoluzioni. A seconda del grado di risoluzione adottato, che può variare da 190 × 96 a 320 × 192 pixel, i sei colori fondamentali, più il bianco, possono essere modificati per ottenere fino a 255 diversi gradi di luminosità, portando così la gamma di sfumature disponibili a 1786 (cioè 6 volte 255) più il nero. In alternativa, alcuni bit, solitamente riservati alla definizione dell'intensità, possono venir utilizzati per produrre ulteriori pagine grafiche; in questo caso si avrà una proporzionale diminuzione nel numero delle differenti sfumature di colore.

Il BASIC RLM, versione estesa, comprende una serie completa di chiamate al modulo grafico, oltre ad altre necessarie per controllare procedure ad esse correlate, come, ad esempio, i trasferimenti su stampante delle immagini presenti sul video. Questo BASIC è molto simile alla versione Microsoft e molte istruzioni sono identiche. La sola differenza



Il monitor

Per meglio sfruttare le caratteristiche grafiche del 380Z, è necessario un monitor a colori con interfaccia RGB. Il computer può essere acquistato con uno standard video di 40 oppure 80 colonne. Questo rappresenta il primo elemento di scelta, che determinerà poi il tipo di monitor adatto ad una particolare applicazione

CHRIS STEVENS

Scheda RAM

La scheda principale, sulla quale è montata la CPU, non può contenere una quantità di RAM sufficiente per la maggior parte delle applicazioni, cosicché è possibile inserire una scheda aggiuntiva, che permette di raggiungere la massima capacità di indirizzamento dello Z80, cioè 64 Kbyte

Scheda video

Il generatore video non fa parte della scheda principale, come nelle macchine di dimensioni più ridotte, ma è invece una scheda a se stante, che ha il compito di controllare i segnali provenienti dalla CPU. Sono disponibili unità video a 40 o 80 colonne

Connettore al bus dello Z80

Scheda CPU

Su di essa si trovano la CPU Z80, una parte della RAM, la ROM e molte componenti essenziali per il funzionamento della macchina, come, ad esempio, la porta per la tastiera

Scheda HRG

Basta inserire questa scheda nel bus del sistema ed il 380Z viene dotato di grafica a colori ad alta risoluzione

Alimentatore

Di grandi dimensioni, è molto pesante, ma estremamente resistente agli urti

Tasto di reset

Dispositivo di bloccaggio

Permette di "sigillare" la macchina, eliminando così il pericolo di una qualsiasi interferenza durante l'esecuzione dei programmi

Ventilatore

Poiché l'alimentatore è dimensionato in modo da servire, senza alcuna modifica, per un alto numero di schede accessorie, è necessario un ventilatore per garantire un adeguato raffreddamento di tutti i componenti

Due unità a dischetti

CHRIS STEVENS

380Z RML

DIMENSIONI

mm. 595 x 425 x 215

CPU

Z80

CLOCK

4 MHz

MEMORIA

56 Kbyte di RAM, fino a 6 Kbyte di ROM

VIDEO

24 linee di 40 oppure 80 caratteri. Grafica in alta risoluzione (320 x 192 oppure 160 x 96 pixel) a 7 colori (con un totale di 255 sfumature per colore)

INTERFACCE

Seriale RS232, parallela per stampante e connettore per registratore a cassette

LINGUAGGI FORNITI

BASIC (versione estesa, della Research Machines)

ALTRI LINGUAGGI DISPONIBILI

ALGOL, FORTRAN e tutti i linguaggi disponibili sotto CP/M

ACCESSORI COMPRESI

Manuale d'istallazione, sistema operativo CP/M, programmi d'utilità in BASIC su dischetto

TASTIERA

60 tasti di alta qualità

DOCUMENTAZIONE

Eccellente, sebbene piuttosto sintetica. Tutte le informazioni sono facilmente rintracciabili ed espresse in forma sufficientemente chiara

notevole riguarda l'uso di etichette, al posto dei numeri di linea, per la chiamata di sottoprogrammi. L'interprete ed il pacchetto HRG occupano, tuttavia, una considerevole quantità di memoria e non lasciano molto spazio ai programmi applicativi scritti dall'utente. In considerazione di ciò, vengono proposte tre versioni BASIC, ciascuna delle quali può comprendere il pacchetto HRG, a seconda della capacità di RAM disponibile.

Il pacchetto di funzioni HRG, essendo totalmente indipendente, può essere usato non solo assieme al BASIC, ma anche a qualsiasi altro linguaggio e, dato che il 380Z funziona di solito con il sistema operativo CP/M, la gamma dei linguaggi disponibili è molto vasta. Parlando di linguaggi, una particolarità che distingue il 380Z da tutti i suoi simili è la possibilità di usare il linguaggio ALGOL, da molti preferito anche al più noto FORTRAN.

Suoni ideali

L'eccezionale generazione di suoni sul Commodore 64

Tra gli home computer più diffusi, il Commodore 64 possiede le caratteristiche più sofisticate per la creazione di suoni, frutto dell'impiego di un chip specializzato, chiamato in inglese Sound Interface Device (Unità d'Interfaccia per il Suono), da cui la sigla SID.
Il SID possiede caratteristiche simili a quelle dei sintetizzatori monofonici comunemente in commercio. Ci sono tre oscillatori con una serie di otto ottave (da 0 a 3900 Hz, in 65536 passi complessivi); un controllo principale di volume, da 0 a 15; quattro forme d'onda per ogni oscillatore (triangolare, a dente di sega, ad impulso variabile e rumore); sincronizzazione dell'oscillatore; generatori di inviluppo, che permettono di effettuare un controllo ADSR su ciascun oscillatore. Altre caratteristiche comprendono una modulazione "ad anello"; un filtro programmabile passa basso, passa banda, e passa alto (è possibile anche selezionare ed eliminare una ristretta banda di frequenze); un filtraggio d'inviluppo; due interfacce analogico/digitali a potenziometro, per controllare le funzioni del SID. Un ingresso audio esterno offre la possibilità di collegare alla macchina altri chip SID. Pertanto, segnali audio esterni possono essere immessi, filtrati e miscelati assieme agli output standard del SID.
È piuttosto difficile descrivere con poche parole gli effetti ottenibili con tutte queste funzioni: tenteremo di chiarire a grandi linee il significato di quanto abbiamo or ora esposto. Innanzi tutto, la sincronizzazione dell'oscillatore produce due segnali (in questo caso due particolari "timbri" o voci) che vengono armonizzati tra loro per creare un suono singolo e più complesso.
La modulazione è, invece, l'alterazione di un segnale ad opera di un secondo segnale che influisca o sulla frequenza o sull'ampiezza (volume) del suono. La modulazione ad anello è la modulazione di ampiezza di un timbro o voce in dipendenza da una seconda voce: ciò produce un suono limpido, ma con effetto stridente, dissonante, utilizzabile, ad esem-

Luci di posizione

La grafica Giocatore-Missile

La grafica Giocatore-Missile, in inglese Player-Missile o PM, ha un ruolo di primo piano fra le caratteristiche grafiche dell'Atari. Le figure mobili generate agiscono indipendentemente da qualsiasi immagine dello sfondo e possono essere programmate in modo da passare sia davanti sia dietro ad una qualsiasi altra figura presente sullo schermo. Grazie a questa possibilità, si può dare un aspetto tridimensionale alle scene che compaiono sul video. La grafica PM può essere mossa velocemente ed in modo lineare sul video: è quindi un'applicazione ideale per i giochi di rapido movimento del tipo "sala giochi". Inoltre, può essere usata per creare immagini statiche più vivide di quelle realizzabili con i normali metodi grafici.

Il segreto di questa grafica risiede nelle componenti hardware ad essa dedicate (ciò vale anche per la grafica a sprite). Sono stati progettati speciali registri per il controllo del movimento, del colore e della visualizzazione delle figure PM. Il compito del programmatore consiste nell'inserire determinati valori in questi registri per manovrare, in seguito, le figure create. In BASIC questa operazione viene eseguita mediante il comando POKE. Quando inseriamo un numero nell'apposito registro (POKE), l'hardware dell'Atari svolge autonomamente il resto del lavoro, alle alte velocità consentite dal linguaggio macchina. Questa procedura risulta, quindi, molto più veloce rispetto a quella controllata da un programma scritto in BASIC.
Esaminiamo adesso come avviene la creazione delle figure PM ed osserviamo i registri che ne controllano il movimento. Le figure dei giocatori vengono progettate partendo da una matrice o "striscia" disposta verticalmente di 8 × 128 (o 256) pixel. Ciascuna riga della matrice viene rappresentata nella memoria del computer da una configurazione di 8 bit (1 byte). Inserendo in memoria, con il comando POKE, appropriati codici in notazione binaria, è possibile definire la figura abbinata ad un giocatore: la procedura è quindi simile a quella usata per realizzare speciali caratteri o simboli grafici (vedere a pag. 246). Si possono definire fino a quattro figure di giocatori,

pio, per simulare il suono di un campanello o di un tamburo di latta.
I filtri permettono di eliminare da un segnale determinati gruppi di frequenze. I diversi tipi di filtraggio possibili sul Commodore 64 producono effetti che i loro stessi nomi esplicitano: i filtri passa basso escludono le frequenze superiori ad una data frequenza, i filtri passa banda escludono le frequenze che si trovano al di sopra o al di sotto di una determinata gamma di frequenze; i filtri d'assorbimento producono il risultato opposto ai filtri passa banda, escludono cioè una specifica banda di frequenze; i filtri passa alto eliminano le frequenze inferiori ad una frequenza data; la risonanza variabile può essere applicata a tutti i filtri finora esaminati per accentuare le frequenze che precedono e seguono i punti d'esclusione. Il filtraggio d'inviluppo rappresenta un caso particolare: esso produce un effetto diverso dai precedenti, in quanto i valori ADSR codificati numericamente e selezionati per l'inviluppo 3, possono essere applicati ad un segnale in modo da trasformare la struttura armonica durante la stessa emissione del suono. L'effetto è simile a quello prodotto da un filtro variabile.
Per generare anche semplici effetti sonori è quindi necessario scrivere una serie di istruzioni BASIC e, in alcuni casi, questo linguaggio non garantisce una velocità tale da far completa giustizia dell'intera gamma di possibilità sonore del SID. Una descrizione completa dei registri di controllo presenti nel chip, richiederebbe più spazio di quello disponibile in queste pagine. Tuttavia, a titolo d'esempio, riportiamo un breve programma dimostrativo (vedere a fianco).
Sebbene il programma sia composto da ben 22 linee, esso permette di suonare su un oscillatore solo cinque note di un semplice accordo. La linea 20 interrompe il collegamento tra il filtro e gli oscillatori; la linea 30 seleziona il volume principale al suo massimo valore; le linee 40 e 50 definiscono un inviluppo del tipo "a pianoforte"; la linea 80 seleziona la frequenza della nota; la 90 e la 100 avviano ed interrompono il ciclo d'inviluppo ADSR e selezionano una forma d'onda a dente di sega per la voce 1; la scansione del tempo è prodotta dal ciclo FOR...NEXT (linee 100, 120 e 140).
Chi desidera dedicarsi a questo impiego "musicale" del computer dovrebbe acquistare uno dei tanti programmi, appositamente progettati, per la generazione dei suoni.

```
10 SID=54272
20 POKESID+23,0
30 POKESID+24,15
40 POKESID+5,40
50 POKESID+6,201
60 FOR N=1 TO 5
70 READ FH,FL,D
80 POKESID+1,FH:  POKESID,
   FL: REM *SUONA LA NOTA*
90 POKESID+4,33
100 FOR I=1 TO 300*D: NEXT I
110 POKESID+4,32
120 FOR I=1 TO 100: NEXT I
130 NEXT N
140 FOR I = 1 TO 2000 : NEXT I
150 POKESID+24,0
160 REM **DATI**
170 DATA 57, 172, 1
180 DATA 64, 188, 1
190 DATA 51, 97, 1
200 DATA 25, 177, 1
210 DATA 38, 126, 1
220 END
```

ciascuna delle quali occupa 128 o 256 byte di memoria. Ad ogni giocatore viene associata l'immagine di un missile, definito da due bit. Per creare i giocatori ed i missili è necessario inserire la configurazione di bit, che ne definisce le caratteristiche individuali, in una ben precisa area di memoria. L'area RAM, utilizzata per questo scopo, viene definita dal programmatore, fornendo un "puntatore" all'inizio dell'area di memoria interessata.
Selezionando una risoluzione verticale ad un solo pixel, è necessario il doppio di memoria rispetto ad una risoluzione verticale a due pixel. Il seguente programma abbina al giocatore 0, in risoluzione verticale a due pixel, la figura di una nave spaziale:

```
 10 REM ***DEFINIZIONE DI UN GIOCATORE***
 20 P=PEEK(1,06)—8: REM P=  PUNTATORE ALLA RAM
 30 POKE 54279,P: REM CONSERVA IL PUNTATORE
 40 BASE=256*P: REM INDIRIZZO INIZIALE
    DELLA ZONA RAM
 50 FOR I=BASE+512 TO BASE+640
 60 POKE I,0: REM AZZERAMENTO ZONA GIOCATORE 0
 70 NEXT I
 80 FOR I=BASE+512+50 TO BASE+530+50
 90 READ A: POKE I,A: REM DEFINISCE LA FIGURA
100 NEXT I
110 DATA 16,16,16,56,40,56,40,56,40
120 DATA 56,56,186,186,146,186,254,186,146
```

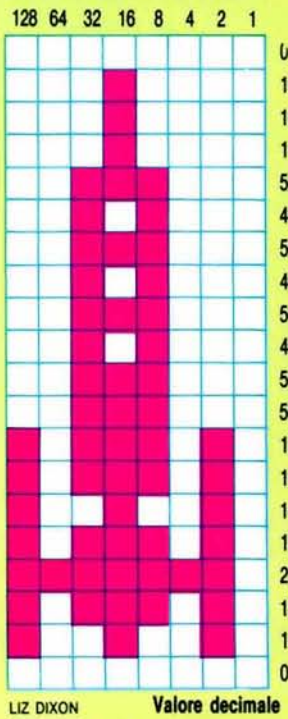
A ciascuna figura di giocatore vengono associati alcuni registri, per il controllo del colore, della posizione orizzontale e del formato. Quest'ultimo registro permette di raddoppiare o quadruplicare le dimensioni di una figura. Ulteriori registri controllano la "priorità" della figura abbinata al giocatore rispetto alle immagini dello sfondo. I missili assumono il colore del giocatore a cui appartengono, ma le loro dimensioni possono essere modificate autonomamente. Nei giochi, vengono usati alcuni registri, con il compito di rivelare le collisioni, sul video, tra giocatori, missili ed immagini dello sfondo. Non esistono, però, dei registri di posizione verticale per i missili ed i giocatori. Il movimento verticale viene quindi prodotto muovendo il contenuto di ciascuna locazione, nella quale è stata inserita la configurazione di bit per quella particolare figura, all'interno dell'area di memoria assegnata in precedenza a quel giocatore. Questa procedura viene eseguita abbastanza rapidamente in linguaggio ASSEMBLER, mentre in BASIC risulta piuttosto lenta. È quindi preferibile progettare figure di giocatori il più possibile tozze e di dimensioni ridotte.
La grafica PM accresce in modo considerevole le possibilità grafiche dell'Atari, anche se non si dimostra così versatile e facile da usare come la grafica a sprite del Commodore 64. Il seguente listato, che costituisce la seconda parte del programma sopra mostrato, permette di colorare la nave spaziale e di farla muovere da destra a sinistra sullo schermo:

```
130 POKE 559,46
140 POKE 53277,3
150 POKE 704,88: REM COLORA DI ROSA IL GIOCATORE 0
160 GRAPHICS 0
170 SETCOLOUR 2,8,2: REM COLORA LO SFONDO DI BLU
180 FOR I=0 TO 320
190 POKE 53248,I: REM ASSEGNA POSIZIONE ORIZZONTALE
200 NEXT I
210 END
```

Un razzo grafico

Prima di definire la figura assegnata a ciascun giocatore, è necessario estrarre i valori in notazione decimale per ciascuna riga di pixel prevista

"Striscia" o matrice per il giocatore



Quanto vale il BASIC?

Esaminiamo questo linguaggio con occhio critico e vediamo quali alternative esistono

Il BASIC è considerato una diretta conseguenza del FORTRAN, uno dei primi linguaggi di programmazione. Una caratteristica che lo distingue dalla maggior parte degli altri linguaggi è quella di essere disponibile in versione "interprete". Ciò significa che quando viene eseguito un programma BASIC, altrove nella memoria è presente uno speciale programma per la conversione di ogni istruzione in codici numerici (codice macchina) eseguibili dalla CPU. Prendiamo, ad esempio, il seguente programma:

```
10 CLS
20 PRINT "IMMETTERE UN NUMERO"
30 INPUT X
40 PRINT "IMMETTERE UN ALTRO NUMERO"
50 INPUT Y
60 PRINT "IL PRODOTTO DEI DUE NUMERI È ";
70 PRINT X * Y
80 PRINT
90 PRINT "RIPROVIAMO?"
100 PRINT "PREMERE 'S' PER CONTINUARE"
110 PRINT "OPPURE 'N' PER TERMINARE IL PROGRAMMA"
120 FOR X = 1 TO 1
130 LET A$ = INKEY$
140 IF A$ <> "S" AND A$ <> "N" THEN X = 0
150 NEXT X
160 IF A$ = "S" THEN GOTO 10
170 END
```

Quando l'interprete BASIC trova, alla linea 10, l'istruzione CLS (pulizia dello schermo), deve creare, o cercare nella propria libreria di moduli "preconfezionati", il gruppo di istruzioni in codice macchina per portare a termine l'operazione richiesta; alla linea 20, il processo si ripete, ma per trovare un modulo che consenta di visualizzare il messaggio racchiuso tra doppi apici. Alla linea 30, invece, l'operazione è più complessa, in quanto occorre associare al nome X una zona di memoria adatta a contenere un numero reale, attendere l'immissione del valore e quindi convertire e memorizzare il numero, sotto forma binaria, nello spazio assegnato ad X. Il procedimento si ripete per ogni linea di programma, finché non si arriva all'istruzione END.

Nei "compilatori", al contrario, il metodo è alquanto diverso: avendo stilato il programma in ogni sua parte, sotto forma di listato, lo si "passa" al compilatore. Questo speciale programma vaglia, linea per linea, il codice cosiddetto "sorgente" (ossia il programma nella sua forma originaria), costruendo via via una versione composta da più moduli in codice macchina, opportunamente collegati tra loro. Il prodotto finale è il programma "oggetto" (ossia eseguibile dalla macchina). I programmi compilati hanno il notevole pregio di essere molto più veloci dei loro equivalenti interpretati, poiché tutta la faticosa opera di traduzione e conversione viene eseguita una sola volta per tutte.

Sembrerebbe ovvio, dunque, optare sempre per i compilatori, piuttosto che per gli interpreti, ma questi ultimi offrono, specialmente nel caso del BASIC, non pochi vantaggi: primo fra tutti, quello di poter eseguire anche singole istruzioni, nel cosiddetto "livello comandi".

Ciò riduce drasticamente le difficoltà di verifica e di correzione degli errori nei programmi. Infatti, si ha modo di interrompere l'esecuzione di un programma in un qualsiasi momento, di esaminare (o modificare) il contenuto delle variabili impiegate e di far riprendere il programma dal punto in cui si era interrotto. Ciò non è possibile nelle versioni compilate, ove non esiste un "livello comandi". Per fare una semplice prova, possiamo lanciare il programma della rubrica computerizzata, presentato precedentemente.

Selezioniamo l'opzione 9 (uscita da programma): il computer visualizza un messaggio o un simbolo (generalmente "ok", o "ready" o semplicemente un ">") per comunicarci di aver concluso l'esecuzione del programma e di essere adesso a "livello comandi", in attesa di un nostro nuovo comando. Ebbene, proviamo a digitare PRINT RMOD<CR>. Come risultato dovremmo ottenere uno 0, poiché il valore della variabile RMOD, azzerata all'inizio del programma, viene modificato soltanto se aggiungiamo un nuovo record. Proviamo adesso con PRINT DIMENSIONE<CR>. Vedremo comparire un valore pari al numero di nominativi presenti nella rubrica, aumentato di 1.

I vantaggi del BASIC

Oltre al vantaggio, appena descritto, di poter facilmente correggere e verificare i programmi durante la stessa fase di sviluppo, il BASIC offre un'altra apprezzabile dote: è relativamente semplice da imparare, anche

per i meno esperti. Una prova di ciò è che, nel corso di programmazione BASIC, in meno di 100 pagine abbiamo esaminato tutte le principali funzioni disponibili ed anche alcuni degli aspetti più complessi del linguaggio. Inoltre, anche la segnalazione di eventuali errori è sufficientemente analitica, consentendo di individuare con rapidità la causa dell'errore. La stessa correzione degli errori non è molto complicata, facilitata da comandi quali EDIT 60 <CR>, che permette di modificare il contenuto di una linea di programma, senza doverla riscrivere da capo.

Gli svantaggi del BASIC

Il BASIC, comunque, presenta anche degli inconvenienti, più o meno appariscenti. Uno è la lentezza degli interpreti che, tuttavia, può passare inosservata in applicazioni nelle quali i tempi d'esecuzione non sono particolarmente critici (ad esempio, nei word processor), mentre può risultare inaccettabile in programmi basati sull'animazione grafica o sul controllo di apparecchiature esterne.
Se la velocità è un fattore critico, l'alternativa è imparare la programmazione in linguaggio macchina (vedere a pag. 448) oppure servirsi di compilatori, ricorrendo a linguaggi quale il PASCAL oppure il FORTH. Questi linguaggi non presentano grosse difficoltà d'apprendimento, ma causano un altro genere d'inconvenienti: il processo di verifica e di correzione (debugging) dei programmi compilati è lungo e difficoltoso, se paragonato all'immediatezza offerta da un linguaggio interprete. Infatti, gli eventuali errori, presenti nel programma "sorgente", vengono segnalati o durante la compilazione o durante l'esecuzione. In ambedue i casi, occorre intervenire sul programma sorgente, apportando le modifiche necessarie, e quindi compilare di nuovo il programma. Dunque, arrivare alla formulazione definitiva di un programma, con un compilatore, può essere molto più laborioso che non con un interprete.
Dal punto di vista formale, il BASIC viene da molti criticato in quanto non consente di applicare i canoni della programmazione strutturata, anzi, permette la formulazione di programmi rappezzati alla meno peggio, costellati da qualche manciata di "maledette" GOTO, che complicano qualsiasi intervento successivo ai programmi.

... e dopo il BASIC?

Il BASIC, abbiamo visto, è facile da imparare, possiede utili funzioni per il trattamento delle stringhe, ma è lento e non sfrutta al massimo le capacità di un computer. D'altro canto, linguaggi più moderni, quali il PASCAL ed il FORTH, offrono al programmatore funzioni e strutture impossibili da ottenere in BASIC.
Anche il PASCAL è stato concepito, inizialmente, per l'insegnamento della programmazione, ma è particolarmente rivolto allo sviluppo di programmi ben strutturati, costringendo ad una verifica logica del programma, prima ancora di poter scrivere le istruzioni che lo compongono. Questo linguaggio esiste soltanto sotto forma di compilatore, pertanto i meno esperti tendono a scoraggiarsi di fronte al faticoso e frustrante ripetersi delle compilazioni, prima di ottenere programmi privi di errori. Inoltre, alcuni aspetti tipici del linguaggio, come l'obbligo di dover definire, all'inizio del programma, le caratteristiche di tutte le variabili impiegate (ad esempio, se si tratta di una variabile reale, intera, ecc.), costituiscono, agli occhi del principiante, un inutile appesantimento.
In PASCAL è forse facile commettere errori di sintassi, ma raramente, proprio grazie alla sua tipica struttura, si riescono ad introdurre errori di logica.
L'altro linguaggio al quale abbiamo accennato, il FORTH, sta divenendo molto diffuso tra i possessori di microcomputer e rappresenta una felice combinazione tra la velocità del linguaggio macchina e le strutture tipiche invece dei linguaggi ad alto livello.
In FORTH non esiste, come ad esempio in BASIC, un limitato vocabolario di comandi: lo stesso programmatore può definire nuovi comandi, partendo da quelli esistenti, allargando così le possibilità del linguaggio.
Facciamo un esempio. In BASIC, la parola PRINT serve a visualizzare i caratteri, racchiusi tra doppi apici, che seguono il comando ed il programmatore non può modificare in alcun modo questo stato di cose. In FORTH, al contrario, la parola PRINT può essere impiegata, ad esempio, come comando per visualizzare sullo schermo i valori esadecimali corrispondenti a ciascun carattere di una stringa. Questo linguaggio offre quindi un'ampia elasticità nella definizione dei comandi e delle funzioni, tanto che i programmi risultanti, una volta compilati, arrivano spesso ad essere compatti quasi quanto quelli scritti in linguaggio macchina.
Qui di seguito riportiamo una breve sintesi sulle caratteristiche dei quattro linguaggi esaminati.

BASIC
Facile da imparare
Facile da ricordare
Facile nella correzione dei programmi
Lento nell'esecuzione
Usa molta memoria
Non incoraggia una programmazione strutturata

Linguaggio macchina (Assembly)
Non molto facile da imparare
Non molto facile da ricordare

LDifficile nella correzione dei programmi
Molto veloce nell'esecuzione
Permette un controllo totale sulle funzioni dell'apparecchio

PASCAL
Moderatamente facile da imparare
Moderatamente facile da ricordare
Correzione dei programmi più complessa del BASIC
Incoraggia all'uso della programmazione strutturata
Esecuzione più veloce del BASIC, ma più lenta dell'Assembly
Comporta le difficoltà comuni a tutti i compilatori (lentezza nella produzione di programmi privi di errori)
Permette un controllo sull'apparecchio, minore di quello offerto dall'Assembly
La manipolazione delle stringhe non è facile quanto in BASIC

FORTH
Non molto facile da imparare (specialmente per chi è abituato al BASIC)
Moderatamente facile da ricordare
La correzione degli errori è facile, se si usa un interprete, difficile se si adopera un compilatore
Nella versione compilata, i programmi sono veloci quanto quelli in linguaggio macchina
Permette il controllo totale sull'apparecchio
Molto economico in termini di memoria
Più facile dell'Assembly

Basic & Basic

LYNX 96

Programma di creazione del file dei dati.
N.B. Questo programma, e quello principale, sono per il Lynx da 96K.

```
1 REM * CREAZIONE FILE DEI DATI *
2 DIM N$(30)
3 LET N$ = "@PRIMO"
4 DIM F$(15)
5 LET F$ = "DUMMY"
6 LET Z = 2
7 EXTBACK 1
8 EXTSTORE 1,Z,N$,N$,N$
9 EXTSTORE 1,F$,F$,F$,F$
10 INPUT "INSERIRE IL NASTRO< AVVIARLO
   E PREMERE RETURN/ENTER";A$
11 SSAVE 1,"RUBRDAT"
12 PRINT "FERMA NASTRO E RIAVVOLGI"
13 END
```

Programma principale

Copiare il listato per lo Spectrum, modificando i seguenti nomi di variabili:

Sostituire	con
DIMENSIONE	Z
RMOD	R
SRTD	D
CORR	C
SCEL	H
FDO	b
MED	m
CMA	t

ed apportare le seguenti modifiche:

```
1100 REM SUBROUTINE **CREMAT**
1110 DIM N$(30)(50)
1120 DIM M$(30)(50)
1130 DIM S$(30)(50)
1140 DIM T$(15)(50)
1150 DIM C$(15)(50)
1160 DIM R$(15)(50)
1170 DIM X$(15)(50)
1180 DIM Z$(30)
:
1210 LET Z=0
1220 LET R=0
1230 LET D=1
1240 LET C=0
1250 LET Z$ = "@PRIMO"
1260 LET Q$=""
1300 RETURN
:
1400 REM SUBROUTINE **RDINFL**
1405 PRINT"INSERIRE IL NASTRO ED AV-
     VIARLO"
1410 GOSUB 3100
1420 SLOAD 1, "RUBRDAT"
1430 PRINT "FERMARE IL NASTRO"
1440 GOSUB 3100
1450 EXTBACK 1
1460 EXTFETCH 1,Z
1470 FOR K=1 TO Z—1
1480 EXTFETCH 1,N$(K),M$(K),S$(K),T$(K),
     C$(K),R$(K),X$(K)
1490 NEXT K
1500 LET Q$=N$(1)
1510 RETURN
:
3120 IF KEYN<>32 THEN LET L=0
:
3780 LET A$ = KEY$
:
3810 LET H=VAL(A$)
3820 IF (H<1) OR (H>9) THEN LET L=0
:
4500 REM SUBROUTINE **MODNOM**
4510 REM CONVERSIONE IN MAIUSCOLE
4520 LET D$=UPC$(N$(Z))
     (cancellare le linee da 4530 a 4590)
:
4600 LET P$=""
4601 LET A$=""
4602 LET T=LEN(D$)
4603 LET S=0
:
4610 REM LOCALIZZA ULTIMO SPAZIO
:
4630 IF MID$(D$,L,1)=" " THEN LET S=L
:
4670 IF MID$(D$,L,1)>"@" THEN
     LET P$=P$+MID$(D$,L,1)
:
4710 IF MID$(D$,L,1)>"@" THEN
     LET A$=A$+MID$(D$,L,1)
```

Le linee da 5410 a 5460 devono essere spezzettate, ossia:

```
5410 LET  U$=N$(L):LET  N$(L)=N$(T):LET
     N$(T)=U$
```

diventa:

```
5410 LET U$=N$(L)
5411 LET N$(L)=N$(T)
5412 LET N$(T)=U$
```

e così via.

```
5600 REM SUBROUTINE **SAVREC**
5605 PRINT "INSERIRE IL NASTRO ED AV-
     VIARLO"
5610 GOSUB 3100
5620 EXTBACK 1
5630 EXTSTORE 1,Z
5640 FOR K=1 TO Z—1
5650 EXTSTORE 1, N$(K),M$(K),S$(K),T$(K),
     C$(K),R$(K),X$(K)
5660 SSAVE 1,"RUBRDAT"
5670 PRINT "FERMARE IL NASTRO"
5680 GOSUB 3100
5690 RETURN
:
```

```
5855 LET X=0
5860 LET m=INT((b+t)/2)
5870 IF M$(m)=U$ THEN LET X=1
5880 IF U$>M$(m) THEN LET b=m+1
:
6080 LET A$=KEY$
:
6110 IF A$="" THEN RETURN
6120 GOSUB 6200
6130 RETURN
:
6730 FOR I=1 TO 1
6735 LET I=0
6740 LET A$=KEY$
6750 IF(A$=E$) OR (A$="") THEN LET I=1
6760 NEXT I
```

Questo frammento deve essere riprodotto nelle linee 6880-6910, 6990-7030, 7110-7140, 7220-7250, 7640-7670.

DRAGON 32

Ecco il programma per la creazione del file dei dati per il Dragon 32:

```
1 REM *CREAZIONE FILE DEI DATI*
2 LET Z = 2
3 LET N$ = "@PRIMO"
4 OPEN "O",#-1,"RUBRDAT"
5 INPUT"INSERIRE IL NASTRO, AVVIARLO E
  PREMERE ENTER (RETURN)";A$
6 PRINT #-1,Z,N$,N$,N$,N$,N$,N$,N$
7 CLOSE #-1
8 PRINT "FERMA NASTRO E RIAVVOLGI"
9 STOP
```

BBC MICRO

Nel Micro BBC sostituire le linee 4, 6 e 7 con:

```
4 F1=OPENOUT("RUBRDAT")
6 PRINT #F1,Z,N$,N$,N$,N$,N$,N$,N$
7 CLOSE #F1
```

COMMODORE 64

VIC-20

Sul Commodore 64 e sul Vic-20 sostituire le linee 4, 6 e 7 con:

```
4 OPEN 1,1,2,"RUBRDAT"
6 PRINT #1,Z,N$,N$,N$,N$,N$,N$,N$
7 CLOSE 1
```

Variabili del programma principale

Per il Dragon, i Commodore ed il BBC, copiare il listato per lo Spectrum (a pagina 458), sostituendo i seguenti nomi di variabili:

Sostituire	con
DIMENSIONE	Z
RMOD	R
SRTD	D
CORR	C
SCEL	H
FDO	FD
CMA	CM
MED	MD

ed apportare le seguenti modifiche:

```
1100 REM SUBROUTINE **CREMAT**
1110 DIM N$(50)
1120 DIM M$(50)
1130 DIM S$(50)
1140 DIM T$(50)
1150 DIM C$(50)
1160 DIM R$(50)
1170 DIM X$(50)
```

Cancellare le linee da 1180 a 1190

```
1210 LET Z=0
1220 LET R=0
1230 LET D=1
1240 LET C=0
1250 LET Z$="@PRIMO"
1260 LET Q$=""
1300 RETURN
```

Ecco la versione Dragon per la subroutine alla linea 1400:

```
1400 REM SUBROUTINE **RDINFL**
1410 OPEN "I",#-1,"RUBRDAT"
1420 PRINT"INSERIRE NASTRO E AVVIARE"
1430 GOSUB 3100
1440 INPUT #-1,Z
1450 FOR K=1 TO Z-1
1460 INPUT#-1,N$(K), M$(K),S$(K),T$(K),
     C$(K),R$(K),X$(K)
1470 NEXT K
1480 Q$=N$(1)
1490 CLOSE#-1
1500 PRINT "FERMARE IL NASTRO"
1510 GOSUB 3100
1520 RETURN
```

Per il Micro BBC sostituire, nel precedente listato:

```
1410 F1=OPENIN("RUBRDAT")
```

e sostituire #-1 con #F1 nelle linee 1440, 1460 e 1490.

Per il Commodore 64 ed il Vic-20, sostituire, nel precedente listato:

```
1410 OPEN 1,1,0,"RUBRDAT"
```

sostituire #-1 con #1 nelle linee 1440, 1460 e cambiare la linea 1490:

```
1490 CLOSE 1
```

Nel Micro BBC sostituire INKEY$ con INKEY$(0) e ogni INPUT"..messaggio..";A$ con INPUT"..messaggio..",A$.
Sui Commodore sostituire tutte le LET A$=INKEY$ con GET A$. Sostituire IF INKEY$ ... con GET GT$:IF GT$ ...

Sul Micro BBC, sul Dragon e sui Commodore, nella subroutine alla linea 4500 sostituire tutti i riferimenti a D$(L) con MID$(D$,L,1).
Sostituire CODE ... con ASC(...) e cambiare PRIMO in ULTIMO nella linea 4610. Eliminare LET L=T alla fine della linea 4630.

Ecco la versione Dragon per la subroutine alla linea 5600 (per le varianti Micro BBC e Commodore vedere le note al programma di creazione del file dati):

```
5600 REM SUBROUTINE **SAVREC**
5610 OPEN "O",#-1,"RUBRDAT"
5620 PRINT "INSERIRE NASTRO E AVVIARE"
5630 GOSUB 3100
5640 PRINT #-1,Z
5650 FOR K=1 TO Z-1
5660 PRINT#-1, Z,N$(K),M$(K),S$(K),
     T$(K),C$(K),R$(K),X$(K)
5670 NEXT K
5680 CLOSE #-1
5690 PRINT "FERMARE IL NASTRO"
5693 GOSUB 3100
5695 RETURN
```

Per il Micro BBC, nella subroutine alla linea 6200 inserire:

```
6205 VDU 2
6275 VDU 3
```

e cambiare tutte le LPRINT in PRINT.

Sui Commodore, inserire:

```
6205 OPEN 4,4:CMD 4
6275 PRINT#4:CLOSE 4
```

e cambiare tutte le LPRINT in PRINT.

Sul Dragon sostituire tutte le LPRINT con PRINT#-2.

Tutti gli uomini del... computer

Lo sviluppo dell'hardware e quello del software sono saldamente connessi

Ci sono stati particolari periodi storici durante i quali la rapidità dei cambiamenti tecnologici ha lasciato l'uomo comune completamente frastornato. Ma niente, fino ad oggi (né il volo dei fratelli Wright, né l'esplorazione lunare), può essere paragonato alla rapidità della rivoluzione microelettronica.

Nel 1971, alcune nuove case produttrici di chip, in California, arrivarono alla conclusione che le principali funzioni di un computer potevano essere raccolte su di una singola piastrina di silicio. Nulla ancora faceva prevedere la futura rivoluzione tecnologica e nessuno parlava di "tecnologie dell'informazione".

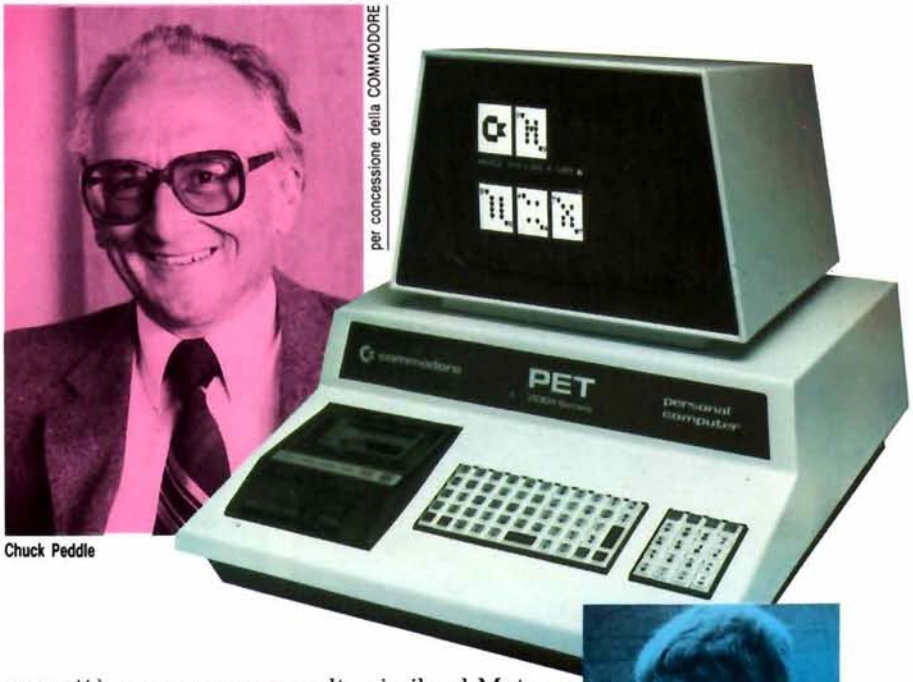
L'idea era di produrre un computer poco costoso e di piccole dimensioni che potesse essere usato per controllare i macchinari di un'industria o gli ascensori: i primi microprocessori furono progettati proprio per questo scopo.

Una di queste case produttrici di chip, la Intel, viene universalmente considerata la produttrice del primo microprocessore, chiamato 4004. I numeri "quattro" presenti nella sigla si riferiscono alla sua capacità: era infatti un processore a 4 bit in grado di elaborare dati in blocchi di quattro cifre binarie. Il chip era in grado di gestire solo una piccola quantità di memoria, appena sufficiente, ad esempio, ad un programma di controllo per un ascensore.

Nel 1972, la Intel sviluppò il chip 8008, un processore ad 8 bit e gli hobbisti cominciarono a costruirsi dei computer basati su questo nuovo chip. Nelle riviste specializzate si potevano trovare descrizioni sulla realizzazione di piccole unità di processo e, per quanto quei piccoli computer non prevedessero né l'uso di monitor, né di tastiere alfanumeriche complete, si può dire che siano stati i primi home computer! Fu proprio da uno di questi progetti che nacque il primo home computer commerciale, l'Altair 8800, che veniva venduto solo in scatola di montaggio.

L'anno seguente comparve sul mercato il primo "vero" microprocessore, l'8080 prodotto sempre dalla Intel. Questo operava su blocchi di dati di 8 bit e poteva controllare fino a 64 kbyte di memoria per i programmi di maggiori dimensioni. Nello stesso tempo gli altri fabbricanti di chip si avviavano a recuperare il terreno perduto. Il chip 6800 della Motorola aveva le stesse caratteristiche hardware dell'8080, ma un diverso gruppo di istruzioni operative. Fu in questo periodo che nacquero i primi problemi di compatibilità software: i programmi scritti per l'8080 non funzionavano sul 6800 e viceversa. Durante lo stesso periodo, altre industrie avevano sviluppato processori di concezione simile a questi: le più importanti erano la National Semiconductor, la Signetics e la Advanced Micro Devices. Ma la prima casa produttrice di grandi dimensioni fu la MOS Technology, alle cui dipendenze era uno dei personaggi chiave della storia dei computer, Chuck Peddle (vedere a pag. 180). Egli lavorava alla MOS Technology quando questa industria progettò un processore molto simile al Motorola 6800, il 6500. In effetti, la somiglianza era tale che dovette essere modificato in quello che divenne il notissimo 6502.

La Commodore era già molto conosciuta in Canada per i suoi apparecchi da ufficio e per le calcolatrici elettroniche. Peddle si trasferì in quella ditta con l'idea di sviluppare un computer completo di schermo, di tastiera e di registratore a cassette per immagazzinare programmi: tutti accessori, cioè, di cui un vero computer poteva aver bisogno. Ovviamente, queste unità dovevano adattarsi al processore 6502. La macchina venne realizza-



Chuck Peddle



Bill Gates

Padri fondatori

Per quanto Chuck Peddle abbia progettato sia il Commodore Pet che il microprocessore 6502 (sul quale è basato il PET), il contributo di Bill Gates, creatore del BASIC Microsoft, è stato altrettanto importante

ta nel 1976 col nome di PET 2001. La sigla (che in inglese significa "cucciolo") fu scelta per suggerire l'idea di un computer "addomesticato", ossia di facile impiego.
Ma, mentre i primi PET cominciavano ad apparire sul mercato, due innovatori si stavano preparando a commercializzare un altro computer, progettato in un piccolo garage della California: Steve Wozniak (vedere a pag. 155) progettò un computer a scheda singola ed assieme all'amico Steve Jobs cominciò a costruirlo in serie ed a venderlo sotto il nome di Apple I. L'Apple II nacque subito dopo, includendo la scheda dell'Apple I in un contenitore dotato di una tastiera alfanumerica.
La Tandy Corporation di Fort Worth, nel Texas, aveva idee tutte proprie per il mercato dei piccoli computer. Questa industria produceva, e produce tutt'oggi, una grande varietà di apparecchiature elettriche come hi-fi, sintetizzatori e radio, commercializzati attraverso una propria catena di negozi (il "Radio Shack") estesa in tutti gli Stati Uniti. L'home computer rappresentava un naturale sviluppo nella propria gamma di prodotti ed il risultato si chiamò TRS-80 Modello 1. TRS significa semplicemente Tandy Radio Shack,

L'"anima" dell'industria
Steve Wozniak ha progettato e costruito, nel proprio garage di casa, il primo Apple I (una scheda senza contenitore). Da quando il progetto venne modificato e dotato di involucro, dando così vita all'Apple II, il suo amico Steve Jobs ha guidato la Apple al suo attuale successo commerciale



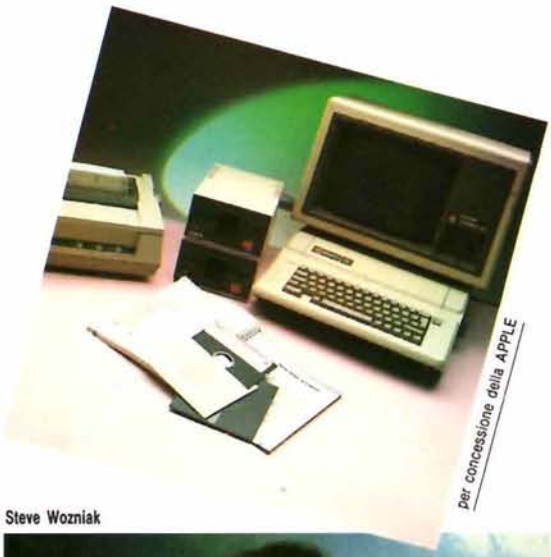
per concessione della APPLE

Steve Jobs

mentre il numero 80 fa riferimento al microprocessore usato, lo Zilog Z80. La Zilog era una delle nuove industrie di chip e aveva prodotto un processore simile all'Intel 8080 ma provvisto di sostanziali miglioramenti. Con il TRS-80 Modello 1, basato sul microprocessore Z80, e l'Apple II, assieme al Commodore PET, basati invece sul 6502, gli home computer cominciarono a mostrare diversità anche nell'hardware. Ma a questa prima divergenza si aggiunsero presto i problemi causati dall'incompatibilità fra macchine e un mercato software non standard. Il tipo di microprocessore usato nelle prime macchine ha grande importanza, perché la scelta condiziona la successiva disponibilità del software. Per questa ragione, mentre si andava sviluppando l'hardware, venivano allo stesso tempo definiti gli standard del software.
Nel 1972, la Intel si serviva di un giovane consulente, Gary Kildall. La sua ditta, la Microprocessor Application Associates, stava lavorando allo sviluppo di un linguaggio per computer, che gli ingegneri della Intel potessero usare per scrivere del software adatto ai nuovi microprocessori prodotti dalla stessa Intel. Kildall riteneva possibile collegare un microprocessore, dotato di memoria RAM, ad una unità a dischetti da 8 pollici e ad una telescrivente, affinché ogni ingegnere disponesse del proprio computer, ma la Intel scelse di seguire il suo precedente metodo di lavoro che prevedeva di utilizzare una sola unità di elaborazione per tutti gli ingegneri. Kildall ed il suo amico John Torode, in un



Gary Kildall
I più recenti sistemi operativi sono stati sviluppati grazie ad un lavoro di équipe, ma il CP/M fu scritto individualmente ed a mano da Gary Kildall. Anche alcune delle ultime versioni riflettono il fatto che questo sistema fu progettato per un hardware molto primitivo



per concessione della APPLE

Steve Wozniak



altro garage californiano, misero a punto un proprio sistema. Torode costruì l'hardware che permetteva al dischetto di lavorare con il microprocessore e Kildall scrisse il software necessario al controllo dell'unità a dischi. Questo programma fu chiamato CP/M (da Control Program/Microcomputers), un nome derivato dal lavoro svolto dallo stesso Kildall con il linguaggio di programmazione della Intel, che si chiamava PL/M (Programming Language/Microcomputers).
Il primo sistema operativo per dischetti montato su micro, fu in breve tempo utilizzato dai produttori di hardware che sentivano l'esi-



per concessione della OSBORNE

Adam Osborne
Adam Osborne è stato per molti anni un giornalista di successo nel campo dei microcomputer, prima di fondare una sua società e di produrre il primo computer portatile del mondo

genza di inserire le unità a dischetti sulle loro macchine. Questo tipo di software ebbe influenza anche sulla progettazione: il CP/M poteva funzionare soltanto sul microprocessore 8080 e sul più veloce 8085 prodotti dalla Intel oppure sullo Z80 della Zilog. Lo Z80 diventò il chip standard per ogni macchina CP/M e la compatibilità CP/M il principale obiettivo delle case di software.

Oltre ai sistemi operativi, gli home computer avevano bisogno di linguaggi di programmazione che permettessero agli utenti di scrivere i loro programmi. Il BASIC, sviluppato presso il Dartmouth College, negli Stati Uniti, era un linguaggio facile da imparare e costituì una scelta quasi ovvia.

Bill Gates, un laureato di Seattle, produsse un interprete BASIC per i micro, un programma di traduzione che poteva essere inserito in un chip a memoria limitata e poteva essere incorporato in un home computer. L'industria di Gate, la Microsoft, diventò la produttrice standard dei linguaggi, così come la Digital Research lo divenne per i sistemi operativi: in questi campi esse sono ancora le compagnie leader.

A questi sviluppi, seguirono rapidi progressi dell'hardware e del software applicativo. Dan Bricklin e Bob Frankston produssero, nella loro industria, la Software Arts, il primo programma di lavagna elettronica per i micro, chiamato VisiCalc. Commercializzato sull'Apple II dalla Personal Software, questo programma divenne il pacchetto applicativo col più alto indice di vendita di ogni tempo e, per sottolineare il suo stretto rapporto con questo prodotto, la Personal Software cambiò il suo nome in VisiCorp. Il WordStar fu prodotto dalla MicroPro di Seymour Rubinstein divenendo in breve tempo il Word Processor CP/M di maggior successo commerciale.

L'hardware, nel frattempo, diventò sempre meno costoso e più potente. Adam Osborne lanciò un computer per uso commerciale di grande successo, corredato da una notevole quantità di costoso software, ad un prezzo altamente competitivo. Naturalmente, non si può tacere in questa panoramica Sir Clive Sinclair, che ha stabilito dei veri e propri "primati" nel livello dei prezzi, con i suoi computer ZX80, ZX81 e ZX Spectrum, offrendo, per la prima volta, a milioni di utenti la possibilità di possedere un computer.

Negli ultimi tempi, lo standard dei microcomputer è stato impostato dalla IBM con il suo IMB PC. Il microprocessore viene dalla Intel, la capostipite di questa tecnologia; il sistema operativo è quello della Microsoft di Bill Gates o della Digital Research di Gary Kildall; due dei primi pacchetti di software che corredavano la macchina erano il VisiCalc ed il WordStar.



per concessione della HEWLETT PACKARD

Cominciando dai "piccoli"...

Sorprendentemente, la tecnologia dei microcomputer deriva più da quella delle calcolatrici programmabili sofisticate (come, ad esempio, l'HP65 della Hewlett-Packard) che dalla prima generazione dei minicomputer



JUDY GOLDHILL

Herman Hauser

Dalla piccola Acorn...

Il contributo portato da Chris Curry e Herman Hauser (come progettisti e direttori della Acorn Computer), anche se si è dimostrato meno rivoluzionario di quello della Sinclair, non è certo stato di minore importanza. L'Atom della Acorn, il Micro della BBC e l'Electron vengono considerati pietre miliari del settore



JUDY GOLDHILL

Chris Curry

Sir Clive Sinclair

La linea innovativa dei suoi prodotti (hi-fi, calcolatrici, radio di piccole dimensioni, TV tascabili ed orologi digitali), ma soprattutto l'ineguagliato successo dei suoi microcomputer (ZX80, XZ81 e Spectrum) hanno fatto sì che nel 1983 gli fosse conferito il titolo di "Sir"



per concessione della SINCLAIR RESEARCH



IAN McKINNELL

Il grande personal

Solo nel 1982 l'IBM ha deciso di dedicarsi in modo massiccio alla produzione su larga scala di microcomputer ed il suo ingresso in questo campo ha avuto l'effetto previsto: quasi tutti i nuovi microcomputer per ufficio sono oggi compatibili con il PC IBM: ciò permette loro di sfruttare al meglio la gran quantità di software disponibile.