VIC da zero +64

di Tommaso Pantuso

Un po' di grafica con il 64

Come gli utenti del Commodore 64 sanno, una carenza di questo computer è rappresentata dalla mancanza di comandi dedicati alla grafica, lacuna che nei modelli che lo hanno seguito (C 16, Plus 4, C 128) è stata colmata con l'introduzione di un Basic più appropriato.

Nell'articolo che segue vogliamo portare il lettore all'interno del 64 e cercare di esaminare insieme in che maniera sia possibile, con i dovuti artifici, sfruttare le possibilità grafiche di questo computer anche in assenza di specifiche istruzioni Basic.

In questo e nell'articolo che seguirà vedremo come porre la macchina in modo grafico e come accedere ai singoli pixel di una mappa di bit sulla quale, a lavoro ultimato, potremo comporre un grafico.

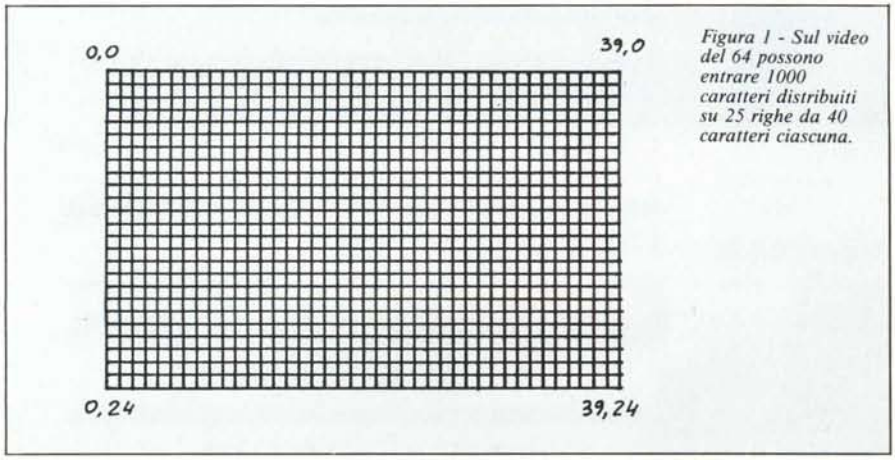

Figura 1 - Sul video del 64 possono entrare 1000 caratteri distribuiti su 25 righe da 40 caratteri ciascuna.

Un po' di conti

Abbiamo da poco terminato di esaminare come programmare sul 64 dei caratteri personalizzati manipolando opportunamente la porta d'ingresso-uscita del microprocessore a gli opportuni registri del Vic-II. Come ben ricorderete, ciascun carattere è contenuto in una griglia quadrata composta da 8x8 = 64 punti elementari, ciascuno dei quali prende il nome di pixel, diminutivo di Picture Cell. Quando un carattere compare sullo schermo, nell'ambito della griglia che lo compone, un certo numero di questi punti elementari è illuminato ed il resto risulta spento. Sappiamo che nella mappa che contiene la configurazione dei caratteri, ciascuna forma viene rappresentata con un insieme di otto byte e, quando modifichiamo un carattere, dobbiamo introdurre in ciascuno di questi byte un pattern di 0 e di 1: gli «1» rappresenteranno sullo schermo un punto illuminato mentre gli «0» rappresenteranno un punto spento. L'insieme dei punti nella griglia relativa a ciascun carattere produrrà sullo schermo la forma programmata.

Se ora noi riusciamo a trovare un modo per accedere in maniera dinamica ciascun pixel, illuminandolo o spegnendolo al momento desiderato, ponendolo per di più nella posizione dello schermo desiderata, potremo facilmente costruire un grafico. Benché la cosa possa sembrare difficile, in effetti non lo è eccessivamente: basta solo capire come modificare la struttura interna della macchina e ricavare l'appropriato algoritmo che ci aiuti a compiere il lavoro.

Per cominciare facciamo un po' di conti. Sullo schermo del 64 possono essere contenute 25 linee, ciascuna composta da un massimo di 40 caratteri, per un totale di 1000 caratteri. Ora, per quanto sappiamo, alla formazione del carattere concorrono 64 pixel e quindi, da un rapido conteggio, ricaviamo che sul teleschermo, tra quelli accesi e quelli spenti, possono essere presenti 64000 punti, come dire 64000 bit.

Ma andiamo oltre. La matrice che contiene ciascun carattere è larga otto pixel ed alta altri otto. Se immaginate ora disposti su una riga tutti e 40 i caratteri che essa può contenere, considerando la prima fila di pixel di ciascun carattere (composta da otto pixel) moltiplicata per il numero di caratteri per riga, cioè 40, ottenete come risultato 320. Se ripetete lo stesso ragionamento immaginando di disporre una schiera di 25 caratteri massimi su di una colonna, otterrete che su una singola fila elementare, dalla parte più alta a quella più in basso dello schermo è possibile disporre di 200 pixel.

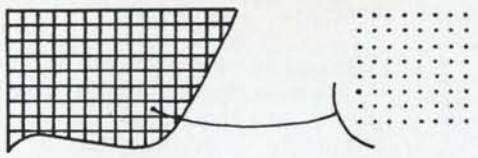

Figura 2 - Ciascun carattere sta dentro una matrice di 8x8 punti. La cella più piccola che contiene il carattere è quindi composta complessivamente da 64 punti, ciascun punto corrispondente ad un bit nella memoria del computer.

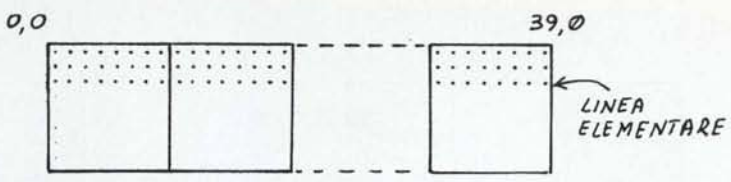

Figura 3 - Se immaginiamo di allineare 40 caratteri su una stessa riga, in quest'ultima saranno contenute 8 linee elementari di 8x40= 320 pixel (accesi o spenti) ciascuna.

In altre parole, con i conteggi fatti, ricaviamo facilmente che l'ampiezza del nostro schermo grafico è di 320 punti in orizzontale per 200 in verticale. È questa l'ampiezza del piano cartesiano su cui graficheremo.

La mappa di bit (Bit Map)

Quando ci mettiamo nel modo bit map, ciascun pixel viene controllato singolarmente modificando opportunamente una locazione di memoria in cui esso è contenuto. Dato che però ogni singola locazione della memoria della macchina è composta da otto bit (1 byte), tramite una sola locazione, trovando il modo opportuno per farlo, potremo controllare ben otto punti sullo schermo accendendoli e spegnendoli singolarmente. Facendo mente locale ai concetti esposti, ricaviamo subito che, essendo lo schermo grafico composto da 64000 punti — ciascuno dei quali controllabile nell'ambito di un gruppo di otto punti contenuti in una singola locazione di memoria — ricaviamo che la memoria necessaria per una mappa di bit completa è 64000/8=8000 locazioni, cioè 8000 byte.

Abbiamo così scoperto un primo fatto importante e cioè che la memoria che conterrà il nostro disegno deve avere un'ampiezza di 8000 byte e da ciò si deduce che tale zona grafica non può essere la memoria di schermo in modo testo, che è composta da soli 1000 byte. In altre parole dovremo definire un'area nella memoria del computer che sarà l'immagine, bit per bit, di ciò che compare sullo schermo grafico.

La definizione di quest'area è per altro molto semplice. Basta modificare un registro del Vic-II ed al resto pensa il sistema operativo. Il registro in questione è racchiuso nella locazione decimale 53272 e in esso bisogna agire su un solo bit, il terzo (cominciando da zero). Quando il bit in questione è posto ad 1, viene predisposta in memoria un'area per la grafica che parte dalla locazione 8192. Dato che (almeno) grazie agli articoli precedenti siete sicuramente pratici di operazioni di And e di Or, non farete fatica ad interpretare le seguenti espressioni che servono per porre ad 1 o a 0 il bit incriminato.

Per attivare il bit 3 dovrete eseguire

```
POKE 53272, PEEK (53272) OR 8
```

mentre per disattivarlo basterà dare l'istruzione

```
POKE 53272, PEEK (53272) AND 247
```

Una volta creata l'opportuna area grafica, dovremo comunicare al sistema che vogliamo passare nel modo Bit Map e ciò può essere fatto agendo sul bit 5 di un altro registro del Vic-II, quello posto nella locazione 53265. Le operazioni per l'attivazione e la disattivazione del Bit Map sono rispettivamente le seguenti:

```
POKE 53265, PEEK (53265)
OR 32 (attivazione)
```

e

```
POKE 53265, PEEK (53265)
AND 223 (disattivazione)
```

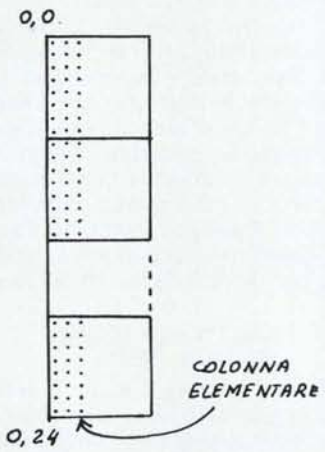

Figura 4 - Ripetendo lo stesso ragionamento fatto nella didascalia della figura 3, questa volta per un'immaginaria colonna di caratteri, ricaviamo che una colonna elementare dello schermo è composta da 200 pixel.

Il colore dei pixel

Prima di andare avanti nella descrizione, dobbiamo soffermarci su un altro fattore importante: il colore. In modo Bit Map normale (non Multicolor, che per il momento non consideriamo) è possibile la scelta di due colori rispettivamente per i pixel accesi e per quelli spenti. In pratica è come se definissimo un colore per lo sfondo (l'insieme dei pixel spenti) ed un colore per il disegno che vogliamo tracciare, cioè per l'insieme dei pixel accesi. Un modo pratico per controllare il colore è quello di agire su un'area di 1000 byte nel modo seguente. Nell'area grafica, consideriamo ciascun gruppo di 8x8=64 bit che ha le stesse dimensioni della matrice relativa ad un carattere. Mettiamo in corrispondenza ogni gruppo di quest'area con una locazione dell'area da 1000 byte succitata. Avremo quindi 64000/64=1000 gruppi in corrispondenza con altrettante locazioni.

Se dividiamo ora ciascuna di queste locazioni in due parti otteniamo, per

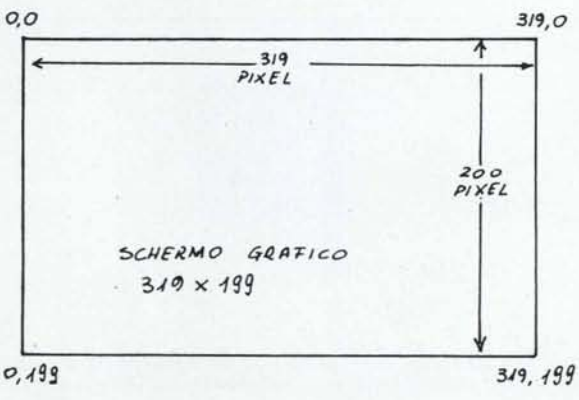

Figura 5 - Potendo accendere e spegnere a piacere uno qualunque dei pixel dello schermo descritto finora avremmo a disposizione un piano grafico da 320x200= 64000 punti.

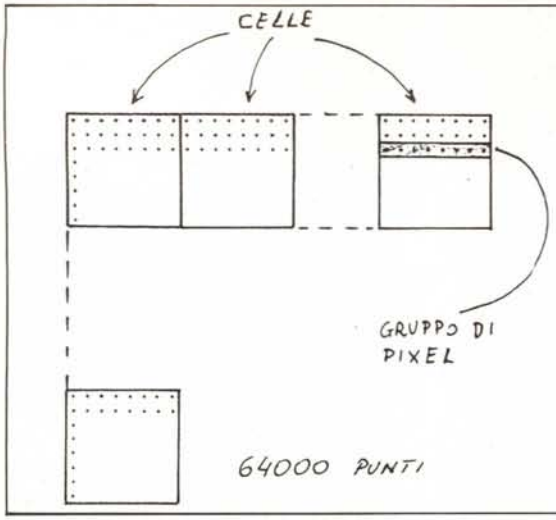

Figura 6
Allo schermo grafico corrisponde un'area di memoria di 8000 byte. Infatti in ciascuna cella sono associati 8 gruppi da 8 pixel ciascuno; se ad ogni gruppo facciamo corrispondere un byte della memoria del computer ci accorgiamo presto che serviranno: 8 bit per gruppo x 8 gruppi x 40 celle per riga x 25 celle per colonna = 64000 bit.

ognuna di esse, due sottoinsiemi (nibble) di 4 bit ciascuno. In ciascun nibble possiamo porre una configurazione che va da 0000 a 1111, cioè da 0 a 15 decimale. Possiamo poi far sì che il numero contenuto in un nibble rappresenti il codice per il colore dello sfondo (bit spenti) dell'area elementare di 64 bit della mappa di bit mentre, il numero nel nibble adiacente, rappresenti quello per il colore del disegno (bit accesi). Agendo globalmente sull'area da 1000 byte (quindi su tutti i gruppi da 64 bit) otterremo di colorare estesamente lo sfondo e il disegno stesso.

Questo procedimento è molto più semplice di quanto possa sembrare a parole anche perché, una volta stabilito il numero da inserire nei 1000 byte interessati, al resto pensa il sistema operativo. Facciamo un esempio che chiarisca meglio le idee.

Supponiamo di voler colorare lo sfondo della prima area, cioè tutti i pixel spenti (tutti i bit di valore zero) con il colore celeste. Per far ciò dovremo porre nel primo nibble, quello di destra, il codice 3, che in binario è 0011 mentre, per avere un disegno nero, cioè avere in nero tutti i pixel accesi, porremo nel nibble di sinistra il codice 0, che in binario è 0000. In definitiva, affiancando i due nibble, otterremo il numero binario

00000011

che in decimale è 3. Se ora memorizziamo questo numero in tutte e mille le locazioni dell'area interessata avremo un solo colore per il fondo ed un solo colore per il disegno.

Il sistema operativo definisce già l'area da destinare al controllo del colore. Tale area è la memoria video. D'altronde perché sprecare memoria visto che quest'area in modo grafico non verrebbe utilizzata? Come ricorderete, all'accensione della macchina, il video è posto a partire dalla locazione 1024 in poi per 1000 byte. Per definire quindi il colore di tutta l'area grafica, per quanto detto, basterà eseguire:

```
FOR I = 1024 TO 2023
POKE I, 3 : NEXT
```

Per determinare il numero decimale da inserire nell'area del colore, una volta determinati i due codici da inserire nei due nibble, basta applicare la formula che segue.

Se n è il codice del colore del fondo ed m quello del disegno, basterà calcolare:

16 x m + n

per ricavare il valore desiderato. Nel nostro caso essendo m = 0 ed n = 3 otterremo

16 x 0 + 3 = 3

Facciamo un esempio.

A questo punto è il caso di riepilogare quanto detto facendo un esempio che mostri come illuminare un certo numero di punti nella mappa di bit.

La prima cosa da fare è definire l'area da 8000 byte, la nostra mappa di bit, che per semplicità faremo partire dalla locazione 8192. Per far ciò basterà la seguente istruzione:

```
10 POKE 53272, PEEK (53272) OR 8
```

Poi dovremo comunicare al sistema che vogliamo abilitare il Bit Map Mode. Ciò viene ottenuto da:

```
20 POKE 53265, PEEK (53265) OR 32
```

Dovremo ancora definire il colore del disegno e del fondo. Con le stesse scelte fatte precedentemente, utilizzando cioè il pattern che dà come risultato 3, scriveremo:

```
30 FOR I = 1024 TO 2023
40 POKE I,3 : NEXT
```

Se intanto provate a far girare il programmino che risulta da quest'insieme di istruzioni, vedrete lo schermo riempirsi di strani caratteri. Ciò è dovuto al fatto che nella mappa grafica sono contenuti dei numeri a caso, cioè essa è «sporca» e quindi va pulita.

La pulizia della mappa si ottiene semplicemente memorizzando uno zero in ciascuno dei suoi 8000 byte. Ciò si può fare con le seguenti linee (che potrete aggiungere dopo aver riportato il sistema nelle condizioni di default con la pressione dei tasti Run/Stop-Restore):

```
50 FOR N = 8192 TO 8192 + 8000
60 POKE N,0 : NEXT
```

A questo punto avremo uno schermo pulito sul quale potranno essere «plottati» i punti che fanno parte del nostro disegno.

Per esempio, se vogliamo disegnare un trattino lungo otto punti in una certa posizione della pagina grafica, inseriremo nel byte interessato il pattern 11111111 (8 pixel accesi) che corrisponde al decimale 255. Provate ad aggiungere la linea

```
70 POKE 10000,255
```

e fate girare il programma completo. Vedrete comparire un trattino in un certo punto dello schermo. Ma perché il trattino compare proprio in quel punto? Come fare per individuare i singoli punti del piano cartesiano accendendoli in modo coerente? Esiste un semplice algoritmo per calcolare la posizione di ciascun punto?

Se avrete pazienza di aspettare fino alla prossima volta, esamineremo insieme questi aspetti conclusivi del problema fornendo anche qualche routine in L/M che velocizzi alcune sequenze del procedimento. MC

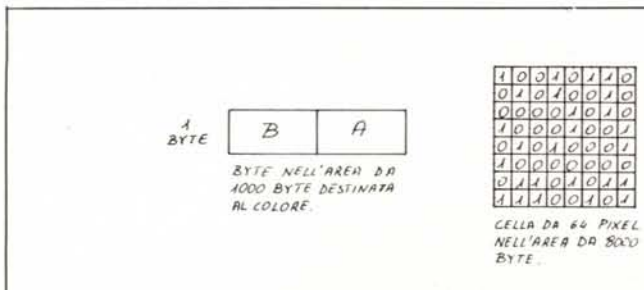

Figura 7 - Un numero contenuto in A associa un colore (da 0 a 15) a tutti i pixel posti a zero nella corrispondente cella sullo schermo grafico. Un numero contenuto nel nibble B associa il colore ai pixel posti ad 1.

SHARP

PC-7000

Il piacere di scegliere

SHARP: alta tecnologia e tradizionale affidabilità nel personal computer e nell'office automation

PC-7000:
Il nuovo personal computer con schermo antiriflesso a cristalli illuminati ed inclinazione regolabile, che consente una perfetta visibilità in qualsiasi condizione di luce.

È costruito a misura d'uomo: compatto, comodo, trasportabile. Per l'eccezionale rapporto prezzo/prestazioni è l'ideale per le applicazioni professionali e gestionali.

La compatibilità con gli standard di mercato assicura una larga reperibilità di software collaudato.

CONFIGURAZIONE BASE
CPU: 8086 (7,37 MHz)
Memoria RAM: 320KB standard (espandibile a 704KB)
Floppy Disk: 2x320/360KB
Schermo: 80 caratteri x 25 righe, 640x200 pixels
Tastiera: conforme a PC/AT IBM**
Porte I/O: 1 seriale e 1 parallela, standard
Software: S.O. *MS-DOS, compatibile con PC IBM e PC XT*****

OPZIONI
Microprocessore aritmetico: 8087
Interfaccia per monitor a colori, PC compatibile
Unità di espansione:
- Hard disk 10MB (3,5")
- 3 schede PC hardware compatibili
Stampante: NLQ, silenziosa, trasportabile con l'unità principale.

* MS-DOS is a trademark of Microsoft Corp.
** IBM is trademark of Int. Business Machine
*** PC-7000 esegue i programmi più diffusi tra quelli scritti per IBM DOS

Distribuito da:

MELCHIONI COMPUTERTIME®

Viale Europa, 49 - 20093 COLOGNO MONZESE (MI)
Tel. (02) 2538621 - TLX METIME I 310352 - FAX (02) 2541420