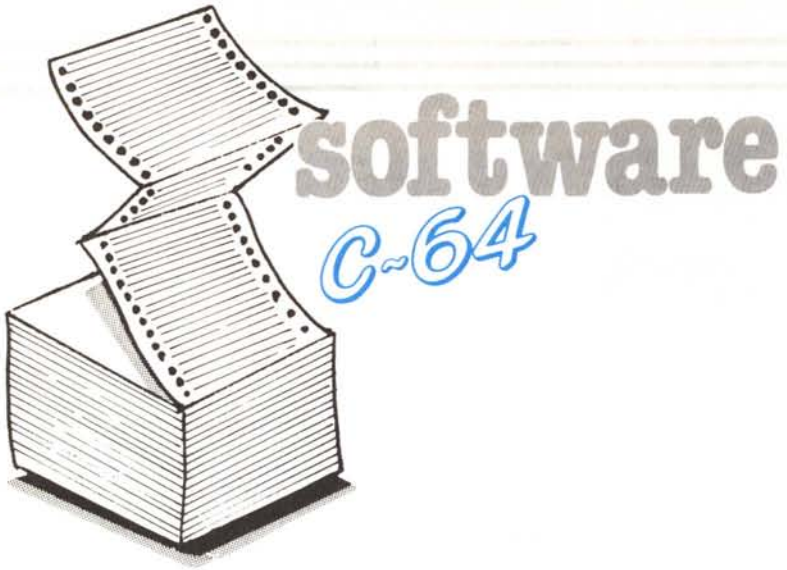

a cura di Tommaso Pantuso

Effetto telecamera

di Paolo Buratti - Trento

Il nome nasce dal fatto che si ha questa sensazione quando questa tecnica viene utilizzata per un video game: il protagonista rimane sempre al centro dello schermo mentre il fondale si muove nella direzione opposta a quella desiderata. Sarebbe tuttavia più preciso definirla una finestra grafica aperta su uno schermo virtuale di dimensioni maggiori. In gergo tecnico prende il nome di «scrolling fine». Comunque la si voglia definire questa è, tra le varie potenzialità del C64, una delle più difficili da realizzare, complice anche la scarsità di informazioni offerte dai vari manuali. Ma andiamo per ordine .

La potenzialità più nota del C64 è quella degli sprite ma ve ne è un'altra molto interessante anche se per molti aspetti sconosciuta: la possibilità di inquadrare lo schermo visibile in otto posizioni diverse sia sull'asse delle x che su quello delle y, tramite dei registri di scorrimento. Questo cosa significa? Facciamo un esempio: quando, scrivendo qualcosa sullo schermo in modo diretto arriviamo all'ultima linea, tutto il contenuto di questo sale verso l'alto di uno spazio carattere (scroll). Questo movimento verso l'alto avviene di scatto: una lettera, che prima era sull'ultima linea, dopo lo scroll si ritroverà una linea sopra e quindi otto pixel più in alto. Se questo movimento a scatti di otto pixel è accettabile quando sullo schermo ci sono solo lettere o cifre, diventa molto fastidioso quando i vari caratteri formano un disegno di insieme (cosa che capita sempre nei video giochi). Grazie alla possibilità di inquadramento sopracitata noi possiamo far si che ogni carattere si sposti di un singolo pixel alla volta fino al limite della posizione finale. A questo punto sarà nostro compito, con un apposito programma, fare lo scroll effettivo di un intero spazio carattere. Questo, ma non solo questo è quello che fa «effetto telecamera» che, come vedremo, può essere utilizzato sia da Basic che da linguaggio macchina senza alcuna interferenza particolare (è gestito dalle interruzioni) ed è completamente programmabile. Si ha piena possibilità di scelta sia per le dimensioni dello schermo virtuale, sia per la sua locazione nella memoria. La sottoprocedura, oltre a gestire lo scorrimento (mantenendo fissa la zona alta dello schermo per conservarci i dati desiderati: punteggio, record ecc.), controlla e aggiorna (solo se lo si desidera) la posizione reale degli sprite in riferimento a quella relativa allo schermo virtuale.

Per modificare l'inquadramento dello schermo si utilizzano i tre bit meno significativi della locazione 53270, per le X, e della 53265, per le Y. Per evitare che anche i bordi seguano lo spostamento è necessario selezionare il «modo» grafico di 38 colonne per 24 righe, azzerando il bit 3 (valore assoluto 8) della locazione 53270 (38 colonne) e lo stesso bit della locazione 53265 per le 24 colonne.

Per realizzare questa sottoprocedura ho dovuto superare parecchie difficoltà tecniche, vediamone alcune. Consideriamo di volere effettuare uno spostamento rallentato a destra di uno spazio carattere. All'inizio sarà sufficiente impostare a 0 la posizione x dello schermo, per poi incrementarlo di 1 per ogni ulteriore spostamento di un pixel verso destra. Il problema nasce quando, arrivati al settimo bit di spostamento, dobbiamo effettuare lo scroll di un intero spazio carattere a destra seguito immediatamente da una nuova impostazione a 0 dell'inquadramento x. Il fatto è che anche il linguaggio macchina non riesce ad eseguire lo scroll in modo abbastanza rapido da non far notare lo sfarfallio che avviene quando l'inquadramento è ancora sul 7 e gran parte dello schermo è già stata spostata più a destra. Per risolvere questo problema ho dovuto utilizzare un'altra possibilità grafica del C64 (in realtà, comune a quasi tutti i computer) quella che permette all'utente di scegliere la pagina video da far vedere al VIC, tramite una semplice modifica dei quattro bit più significativi (nybble alto) della locazione 53272. Grazie a questa tecnica tutto diventa molto semplice: prima si eseguirà lo scroll su una pagina non inquadrata dal VIC, dopodiché basterà impostare a 0 (continuando nell'esempio precedente) l'inquadramento x e «mandare in onda», tramite la locazione 53272, la pagina con lo scroll effettuato.

Visto che entrambe queste operazioni sono semplicemente due impostazioni verranno eseguite in un tempo tanto rapido da rendere assolutamente invisibile lo sfarfallio.

Un altro problema, molto più complesso, nasce dalla necessità di avere una parte dello schermo fissa, per conservare scritte che siano leggibili. Il punto è che l'inquadramento riguarda tutto lo schermo, se quindi è facile far si che il programma di scroll sposti solo la parte bassa dello schermo, non è possibile «spiegare» al VIC di lasciare ferma la parte alta. Ovviamente pure in questo caso c'è la soluzione, anche se un po' più complessa: le interruzioni video. Le interruzioni (IRQ) funzionano grosso modo così: il microprocessore sta eseguendo un programma principale (supponiamo quello di un programma Basic, tramite l'interprete) quando succede un determinato evento. A questo punto, se l'interruzione per quel tipo di evento è abilitata tramite la maschera di interrupt posta in 53274 ($D01A), il m.p. interrompe l'esecuzione del programma Basic, salva i registri e l'indirizzo della prossima istruzione ch avrebbe dovuto eseguire, e salta all'indirizzo contenuto nei due byte 788-789 ($0314-$0315). Quando le istruzioni che partono da quest'indirizzo saranno state eseguite, tornerà al programma principale, recuperando i valori precedentemente salvati nello stack.

Sul C64 gli «eventi» che possono causare un'interruzione sono i seguenti: triggerato per penna ottica (bit 3), collisione fra due sprite (bit 2), contatto tra sprite e fondo (bit 1) e comparazione quadro (bit 0) che è proprio quella che ci interessa. In

È disponibile, presso la redazione, il disco con i programmi pubblicati in questa rubrica. Le istruzioni per l'acquisto e l'elenco degli altri programmi disponibili sono a pag. 249.

realtà vi sarebbero altre sorgenti di interruzioni: interruzione timer A o B, porta seriale ed altre ancora ma queste sono una cosa leggermente diversa dalle precedenti, si tratta infatti di NMI (Interruzioni Non Mascherabili) che per ora non ci interessano. Per spiegare a cosa servono le interruzioni video, originate dalla coincidenza di valori tra il comparatore di quadro e la nostra precedente impostazione del medesimo, è necessario fare un passo indietro e considerare il modo in cui il VIC manda le immagini al monitor. Nella maggioranza dei computer ogni disegno, carattere o sprite viene depositato in una RAM video che forma la schermata che verrà mandata in uscita sul monitor. Questo come ben sapete non avviene per il C64. Il VIC costruisce l'immagine «al volo» e la manda al monitor senza prima organizzarla in qualche zona della memoria. Difatti la ram che parte dalla locazione 1024 è solo un vettore che contiene i puntatori dei vari caratteri e non certamente uno schermo ad alta risoluzione tradizionale. Il fatto che il VIC costruisca l'immagine direttamente in uscita, linea per linea potrebbe essere un vantaggio se si riuscisse a modificare razionalmente e rapidamente i vari registri da cui preleva le informazioni necessarie; questo è possibile appunto tramite le interruzioni video. Il comparatore di quadro, di cui sopra, è formato dalla locazione 53266 (SD012) e ha il proprio nell'MSB (Most Significant Byte) della 53265 (SD011). Quando il VIC costruisce l'immagine video parte dall'alto e scende verso il basso. Nel comparatore di quadro ci sarà il numero corrispondente alla linea di pixel che sta eseguendo in quel momento. Se si vuole generare un'interruzione video sarà sufficiente inserire nel comparatore di quadro il numero di linea in cui si desidera che questa avvenga e abilitare l'interruzione attivando il bit 1 della maschera SD01A. Perché questo abbia un senso è necessario che ci siano due punti di interruzione in modo da mantenere dati diversi per due fasce di schermo. Un possibile esempio potrebbe essere questo: un programma imposta l'interruzione video alla 100esima linea, quando questa avviene, il programma, a cui verrà dato il controllo, imposterà a nero il colore di fondo e preparerà la prossima interruzione alla linea 0 modificando anche l'indirizzo di IRQ. Quando si arriverà alla linea 0 verrà passato il comando a un programma che imposterà a bianco il colore di fondo e preparerà la prossima IRQ alla linea 100 con l'indirizzo precedente. In questo modo le due routine si alterneranno nel soddisfare le due fonti di interruzione, una alla linea 0 e l'altra a quella 100. L'effetto finale sarà di avere stabilmente la parte alta dello schermo col fondo bianco e la parte bassa nero. Ho usato l'esempio del colore di fondo ma ovviamente è possibile qualunque modifica con questa tecnica. Si può avere la parte alta in modo testo e la parte bassa in alta risoluzione oppure, in alto otto animazioni e in basso altre otto animazioni e così via. È quindi anche possibile risolvere il nostro problema iniziale: mantenere la parte alta dello schermo sempre con lo stesso inquadramento e la parte bassa mobile.

Grazie a questa tecnica ho potuto anche risolvere un altro problema minore: la scomparsa, nella parte alta, degli sprite. Se difatti a destra, a sinistra e in basso questi possono scomparire dolcemente dietro il bordo è chiaro che questo non può accadere in alto, nel nuovo bordo corrispondente al punto di frizione tra la parte alta fissa e quella bassa mobile. Potevo quindi scegliere tra il far comparire lo sprite quando ancora è sull'intestazione oppure visualizzarlo di colpo solo quando è completamente nella parte bassa, in entrambi i casi l'effetto sarebbe stato fastidioso. Grazie alle interruzioni video ho invece potuto simulare un bordo, come quelli originali, dietro cui lo sprite scomparisse pixel per pixel. Questo è stato possibile impostando nella parte alta i puntatori di tutti gli sprite ad una zona di memoria azzerata, recuperando i loro valori originari nella parte bassa.

Modalità d'uso

Contrariamente alle apparenze, l'utilizzo di questa sottoprocedura è semplicissimo sotto ogni aspetto.

La procedura è compresa tra $8000 e $94DB e lascia quindi libera per il Basic

```
1 REM ************************************************************************
2 REM *                                                                      *
3 REM *                  EFFETTO TELECAMERA BY BURATTI PAOLO                 *
4 REM *                                FOR                                   *
5 REM *                           MC MICROCOMPUTER                           *
6 REM ************************************************************************
10 FORJ=0TO1150
20 READ CODE
30 POKE36955+J,CODE
40 NEXT
45 PRINT"OK! CARICATI CODICI LM IN MEMORIA"
50 END
1000 REM CODICI OGGETTO L.M.
1001 DATA0,0,255,1,255,1,0,0,0,7,0,7,0,0,1,255,1,255,0,0,255,255,7,0,7,0,255
1002 DATA255,255,255,41,0,31,0,255,255,0,32,240,128,240,136,141,73,144,169
1003 DATA0,141,58,144,141,59,144,141,62,144,173,4,144,141,61,144,162,7,78,73
1004 DATA144,144,19,173,58,144,24,109,61,144,141,58,144,173,59,144,109,62,144
1005 DATA141,59,144,14,61,144,46,62,144,202,16,223,96,169,4,141,136,2,76,71
1006 DATA254,120,169,127,141,13,220,169,1,141,26,208,169,90,141,18,208,169
1007 DATA27,141,17,208,169,66,162,145,141,20,3,142,21,3,169,0,141,70,144,141
1008 DATA56,144,141,6,144,141,60,144,141,69,144,173,4,144,56,233,39,141,121
1009 DATA144,173,5,144,56,233,19,141,123,144,32,47,147,169,7,141,75,144,141
1010 DATA76,144,173,2,221,9,3,141,2,221,173,0,221,41,252,9,1,141,0,221,169
1011 DATA136,141,136,2,169,147,32,210,255,32,155,148,169,128,141,136,2,169
1012 DATA189,162,144,141,24,3,142,25,3,88,96,173,25,208,141,25,208,173,57,144
1013 DATA240,8,32,149,147,169,0,141,57,144,173,17,208,41,48,141,17,208,173
1014 DATA22,208,41,16,141,22,208,169,135,162,145,141,20,3,142,21,3,173,24,208
1015 DATA41,15,141,24,208,169,91,141,18,208,169,0,141,33,208,32,177,148,76
1016 DATA49,234,173,25,208,141,25,208,173,17,208,9,32,141,17,208,173,22,208
1017 DATA72,9,16,141,22,208,104,41,16,141,90,144,173,24,208,41,15,141,69,144
1018 DATA169,2,141,24,208,173,18,208,201,96,144,249,32,22,147,174,56,144,169
1019 DATA2,29,127,144,141,24,208,173,69,144,29,127,144,72,173,17,208,41,95
1020 DATA170,173,22,208,41,239,13,90,144,168,173,18,208,201,103,144,249,169
1021 DATA2,141,33,208,104,141,24,208,142,17,208,140,22,208,174,56,144,32,191
1022 DATA148,162,66,160,145,142,20,3,140,21,3,169,0,141,18,208,173,60,144,208
1023 DATA13,173,7,144,208,3,141,6,144,173,6,144,208,37,173,70,144,208,6,104
1024 DATA168,104,170,104,64,162,6,104,202,208,252,162,5,189,63,144,72,202,16
1025 DATA249,169,0,141,70,144,141,60,144,76,30,146,174,6,144,202,206,7,144
1026 DATA173,75,144,141,71,144,173,76,144,141,72,144,224,2,176,34,173,72,144
1027 DATA221,99,144,240,37,173,71,144,24,125,91,144,141,75,144,173,72,144,24
1028 DATA125,95,144,141,76,144,238,57,144,76,30,146,173,71,144,221,99,144,240
1029 DATA3,76,93,146,173,2,144,221,119,144,208,11,169,0,141,7,144,141,6,144
1030 DATA76,30,146,173,3,144,221,123,144,240,237,169,1,141,60,144,173,2,144
1031 DATA24,125,103,144,141,2,144,173,3,144,24,125,107,144,141,3,144,189,111
1032 DATA144,48,3,141,71,144,189,115,144,48,3,141,72,144,104,141,63,144,104
1033 DATA141,64,144,104,141,65,144,104,141,66,144,104,141,67,144,104,141,68
1034 DATA144,169,146,72,169,237,72,169,48,72,72,72,72,76,30,146,32,45,147,173
1035 DATA20,3,201,66,208,249,238,57,144,120,173,89,144,141,56,144,173,71,144
1036 DATA141,75,144,173,72,144,141,76,144,169,1,141,70,144,88,76,19,147,173
1037 DATA17,208,41,48,13,76,144,141,17,208,173,22,208,41,240,13,75,144,141
1038 DATA22,208,96,56,36,24,173,56,144,144,2,73,1,141,89,144,10,170,189,129
1039 DATA144,133,251,189,130,144,133,252,173,3,144,32,133,144,173,58,144,24
1040 DATA109,2,144,133,253,173,59,144,105,0,133,254,165,253,24,109,0,144,133
1041 DATA253,165,254,109,1,144,133,254,162,255,232,160,255,200,177,253,145
1042 DATA251,192,39,208,247,165,253,24,109,4,144,133,253,144,2,230,254,165
1043 DATA251,24,105,40,133,251,144,2,230,252,224,18,208,217,96,173,2,144,141
1044 DATA77,144,173,3,144,141,81,144,169,0,141,78,144,141,82,144,162,3,14,77
1045 DATA144,46,78,144,14,81,144,46,82,144,202,208,241,173,77,144,24,105,88
1046 DATA141,79,144,173,78,144,105,1,141,80,144,173,81,144,24,105,24,141,81
1047 DATA144,144,3,238,82,144,173,81,144,24,105,192,141,83,144,173,82,144,105
1048 DATA0,141,84,144,169,1,141,74,144,162,255,232,189,40,144,208,3,76,130
1049 DATA148,189,8,144,56,237,77,144,141,85,144,189,24,144,237,78,144,144,16
1050 DATA141,86,144,173,79,144,56,253,8,144,173,80,144,253,24,144,144,110,189
1051 DATA16,144,56,237,81,144,141,87,144,189,32,144,237,82,144,144,92,141,88
1052 DATA144,173,83,144,56,253,16,144,173,84,144,253,32,144,144,74,138,10,168
1053 DATA173,85,144,24,109,75,144,153,0,208,173,86,144,105,0,8,173,74,144,40
1054 DATA208,11,73,255,45,16,208,141,16,208,76,108,148,13,16,208,141,16,208
1055 DATA173,87,144,24,105,69,24,109,76,144,153,1,208,173,74,144,13,21,208
1056 DATA141,21,208,224,7,240,20,14,74,144,76,241,147,173,74,144,73,255,45
1057 DATA21,208,141,21,208,76,130,148,96,169,0,162,255,232,157,179,133,157
1058 DATA179,134,157,240,134,157,187,135,224,255,208,239,96,162,7,169,29,157
1059 DATA248,131,157,248,139,202,16,245,96,138,208,12,160,7,185,48,144,153
1060 DATA248,131,136,16,247,96,160,7,185,48,144,153,248,139,136,16,247,96
```

tutta la memoria fino a 32768 ($8000). Chi programma in linguaggio macchina potrà anche utilizzare i soliti 4K posti a partire da $C000 (49152 decimale).

Gli indirizzi da utilizzare per controllare «effetto telecamera» partono dalla locazione 36864 che noi d'ora in poi, per questioni di semplicità, definiremo: BASE.

BASE + 0 e BASE + 1 dovranno contenere l'indirizzo nel solito formato basso alto, dove si intende allocare lo schermo virtuale. Se ad esempio volessimo piazzarlo in 49152, dovremmo inserire in BASE + 0, zero, e in BASE + 1, centonovantadue.

Difatti: 192*256 + 0 = 49152.

Per decidere il numero di righe e colonne, che desideriamo per lo schermo virtuale, dovremo impostare in BASE + 4, il numero delle colonne e in BASE + 5, il numero delle righe. I limiti minimi sono rispettivamente: 40 per le colonne e 20 per le righe. Il limite massimo invece è, per entrambe le coordinate, posto a 254.

La procedura si incarica di muovere la finestra grafica sullo schermo virtuale a seconda delle impostazioni fatte nel registro di direzione posto in BASE + 6, e in quello di quantità (di pixel) posto in BASE + 7.

Le direzioni sono contraddistinte dai seguenti numeri: 1 in alto, 2 in basso, 3 a sinistra e 4 a destra. Se quindi noi vogliamo che la finestra si sposti di, supponiamo, 40 pixel a sinistra dovremo prima (IMPORTANTE!!) impostare il numero di pixel (BASE + 7 = 40) e poi la direzione: BASE + 6 = 3. A questo punto ci penseranno le interruzioni a fare il resto. Il programma principale potrà contemporaneamente svolgere altre funzioni mentre «effetto telecamera» farà scorrere lo schermo. Quando la finestra arriva a un limite dello schermo virtuale un eventuale ordine di scorrimento in quella direzione verrà ignorato dalla procedura. Per decidere l'impostazione iniziale della finestra bisognerà agire su BASE + 2, per la coordinata X, e su BASE + 3, per quella Y.

Se ad esempio impostiamo entrambi i registri a zero avremo, sullo schermo reale, la parte in alto a sinistra dello schermo virtuale. Questi valori possono anche essere modificati in un secondo tempo. Prima sarà però necessario verificare che in quel momento la procedura non stia effettuando uno scorrimento. Per rendersene conto sarà sufficiente testare il registro di quantità, BASE + 7. Quando questo avrà valore zero significherà che lo scorrimento è terminato.

Se si desidera che sia «effetto telecamera» ad aggiornare le coordinate delle animazioni sullo schermo reale, sarà necessario utilizzare i seguenti vettori:

BASE + 8 - Byte basso coordinata X.
BASE + 16 - Byte basso coordinata Y.
BASE + 24 - Byte alto coordinata X.
BASE + 32 - Byte alto coordinata Y.
BASE + 40 - registro di attivazione delle animazioni controllate da «effetto telecamera».

Se il valore è posto a zero l'animazione corrispondente sarà ignorata dalla procedura, in caso contrario «effetto telecamera» si incaricherà di aggiornarne le coordinate reali.

Forse, però, è meglio spiegare cosa si intende con «aggiornare le coordinate reali...». Dopo avere definito lo schermo virtuale questo dovrà essere considerato a tutti gli effetti, dal programmatore, come l'unico schermo esistente. Ogni modifica che si voglia effettuare sarà eseguita direttamente sullo schermo virtuale. Questo modo di procedere non dovrebbe essere troppo complicato, visto che, il tutto, si riduce a sostituire, nella solita formuletta [1024 + X + 40*Y], l'origine dello schermo virtuale a quella in 1024, e il numero delle colonne a 40. Se ad esempio abbiamo definito uno schermo virtuale di 80 colonne per 40 righe posizionato a partire da 49152, per modificare una singola locazione di «schermo» dovremo utilizzare la nuova formula 49152 + X + 80*Y. I problemi nascono con le animazioni. Supponiamo, continuando nell'esempio precedente, di volere piazzare un'animazione in mezzo a questo schermo virtuale. La sua posizione sull'asse delle X sarebbe quindi grosso modo di 24 + 40*8 = 344, ma sarebbe un errore limitarsi a considerare questo numero come quello reale, bisognerà anche considerare quale punto, in quel momento, è inquadrato dello schermo virtuale. Se ad esempio la finestra è posizionata a (0,0) la coordinata reale sarebbe proprio 344, ma se solo la finestra è in (8,0), ecco che la posizione sul monitor sarebbe sbagliata di 64 pixel troppo a destra. Facendo controllare alla procedura questo aggiornamento ogni problema svanisce. Sarà sufficiente considerare come registri di posizione i nuovi vettori su elencati e al resto ci penserà «effetto telecamera».

I puntatori per le immagini degli sprite che partivano da 2040 ora sono posizionati a partire da BASE + 48.

Per avere un aggiornamento dell'immagine dello schermo virtuale sarà sufficiente effettuare una SYS 37679.

Per avere un aggiornamento delle coordinate degli sprite basterà, dopo averne modificato i valori, impostare BASE + 57 a uno.

Per attivare «effetto telecamera» effettuare una SYS 37061.

Per poter avere due pagine video vicine ho dovuto modificare il banco video. Questo significa che adesso i nuovi set di caratteri e le conseguenti impostazioni nel comando: POKE 53272, (PEEK (53272) AND 240) OR A avranno i seguenti valori:

A = 4 - Caratteri standard, maiuscole/grafici.
A = 6 - Caratteri standard, maiuscole/minuscole.
A = 8 - I dati per i nuovi caratteri dovranno essere inseriti a partire da 40960. (Questa

Demo Effetto telecamera

```
1 FORJ=0TO3:READKX(J),KY(J):NEXT:REM LEGGE INCREMENTI 4 DIREZIONI
2 DATA0,1,-1,0,0,-1,1,0:REM PER DISEGNARE LABIRINTO CASUALE
3 PRINT"(CYN)(CLR)":REM COLORE CARATTERI VERDE
5 POKE53280,0:POKE37245,0:POKE37348,2:REM COLORE PARTE ALTA E BORDO NERO
8 FORJ=0TO63:READA:POKE35840+J,A:NEXT:REM DATI PER SPRITE (35840=64*48+32768)
10 BS=36864:REM INIZIO INDIRIZZI DI PROCEDURA TELECAMERA
11 FORJ=0TO7:POKEBS+48+J,48:NEXT:REM IMMAGINE DI MONGOLFIERA AD OGNI SPRITES
20 POKEBS+2,0:POKEBS+3,0:REM IMPOSTA A (0,0) IL PUNTO IN ALTO A SINISTRA
30 POKEBS+4,80:POKEBS+5,50:REM 80 COLONNE PER 50 RIGHE
40 POKEBS,0:POKEBS+1,192:REM BASE GRANDE SCHERMO IN 49152 (192*256+0=49152)
41 FORJ=0TO7
43 READCX,CY:REM LEGGE COORDINATE X Y DI DI POSIZIONE CARATTERE
45 CX=CX*8+24:CY=CY*8+50:REM TRASFORMA COORDINATE CARATTERE IN PIXEL
47 POKEBS+8+J,CXAND255:POKEBS+24+J,CX/256:REM IMPOSTA COORDINATA X
49 POKEBS+16+J,CYAND255:POKEBS+32+J,CY/256:REM IMPOSTA COORDINATA Y
50 POKEBS+40+J,1:POKE53287+J,1
52 NEXT
55 FORJ=0TO4000:POKE49152+J,209:NEXT:REM RIEMPIE SCHERMO DI CARATTERI PIENI
56 FORX=0TO79:POKE49152+X,102:POKE49152+49*80+X,102:NEXT:REM CORNICE ASSE X
57 FORY=0TO49:POKE49152+Y*80,102:POKE49152+79+Y*80,102:NEXT:REM CORNICE ASSEY
58 GOSUB500:REM DISEGNA LABIRINTO CASUALE
60 SYS37061:PRINT"(CLR)(WHT)(DOWN)":REM ATTIVA PROCEDURA E CANCELLA NUOVO SCHER
MO IN $8000
61 PRINT
62 PRINT"(RGHT)      ***** EFFETTO TELECAMERA *****"
63 PRINT
64 PRINT"(RGHT)              BY BURATTI PAOLO"
69 BI=2
70 IFPEEK(BS+6)>0THEN70:REM SE NON ANCORA|TERMINATO SCORRIMENTO ATTENDERE
80 X=PEEK(56320):IFX=127THEN80:REM ATTENDE COMANDI DA JOYSTICK
82 K=0
90 IF(XAND1)=0THENK=1:REM IN ALTO
92 IF(XAND2)=0THENK=2:REM IN BASSO
94 IF(XAND4)=0THENK=3:REM A SINISTRA
96 IF(XAND8)=0THENK=4:REM A DESTRA
98 IF((XAND16)=0)AND(BI=2)THENBI=80:GOTO100
99 IF(XAND16)=0THENBI=2
100 POKEBS+7,BI:REM IMPOSTA NUMERO DI PIXEL DI SCORRIMENTO
110 POKEBS+6,K:REM IMPOSTA DIREZIONE DI SCORRIMENTO
120 GOTO70
500 I=0:REM COSTRUISCE LABIRINTO CASUALE
510 X=40:Y=25:FC=8
520 C=INT(RND(1)*4):LL=INT(RND(1)*20)+1:Z=0
521 IFC=FCTHEN520
522 IFABS(C-FC)=2THEN520
525 IFI=60THENRETURN
530 FX=X+KX(C):FY=Y+KY(C)
540 IFFX=79ORFX=0ORFY=49ORFY=0THENFC=C:GOTO520
550 IFZ=LLTHENI=I+1:FC=C:GOTO520
560 Z=Z+1:X=FX:Y=FY:POKE49152+X+Y*80,32:GOTO525
570 REM IMMAGINE ANIMAZIONE MONGOLFIERA COMMODORE
580 DATA0,127,0,1,255,192,3,255,224,3,231,224
590 DATA7,217,240,7,220,240,7,217,240,3,231,224
600 DATA3,255,224,3,255,224,2,255,160,1,127,64
610 DATA1,62,64,0,156,128,0,156,128,0,73,0,0,73,0
650 DATA0,62,0,0,62,0,0,62,0,0,28,0,0
660 REM COORDINATE DI ANIMAZIONI.CARATTERE SOTTO ANIMAZIONE IN ALTO A SINISTRA
670 DATA5,40,21,33,75,32,20,8,66,29,41,4,25,24,66,43
```

zona è «sotto» alla ROM del Basic ma non ha alcuna importanza visto che il VIC vede solo la RAM).
A = 10 - Nuovo set a partire da 43008.
A = 12 - Nuovo set a partire da 45056.
A = 14 - Nuovo set a partire da 47104.

Grazie alla modifica del banco video tutta la memoria fino a 32768 è REALMENTE disponibile per il Basic a differenza di quanto accade col banco video standard quando, dovendo piazzare caratteri e animazioni nella parte bassa della memoria, per il Basic restano solo pochi K. A proposito, ovviamente, essendo cambiato il banco video anche i dati per gli sprite sono diversi. I puntatori, come già detto, sono a partire da BASE + 48. Per ottenere la locazione corrispondente al puntatore oltre a moltiplicare, come sempre per 64, sarà anche necessario sommare il valore del nuovo banco e cioè 32768.

Per gli sprite sono disponibili le stesse locazioni dei set carattere e cioè da 40960 fino a 49152, escluso.

La parte superiore fissa può essere scritta anche con delle PRINT avendo però l'accortezza di non stampare caratteri che modifichino tutto lo schermo (clear per esempio). Oppure con delle operazioni di scrittura direttamente a partire da 32768, zona di inizio della memoria video. Per modificare il colore di fondo della parte alta bisogna utilizzare la locazione posta a BASE + 381 (= 37245) mentre per la parte bassa si deve utilizzare quella posta a BASE + 484 (= 37348).

Il programma dimostrativo non è niente di eccezionale ma dovrebbe essere sufficiente per colmare le immancabili lacune presenti in questa spiegazione. Per utilizzarlo è sufficiente inserire il joystick in porta 2 (porta «A», secondo certi manuali). Le direzioni servono a fare scorrere lo schermo mentre «fire» fa cambiare il numero di pixel di scorrimento (da 2 a 80). Quando lo state usando prestate attenzione a cosa accade agli sprite quando si spostano verso l'alto!

Calendario perpetuo

di Marco Averone - Potenza

Dopo quella pubblicata per C128, non poteva mancare la versione per Commodore 64 del programma calendario perpetuo di cui diamo alcune note descrittive. Il programma serve per generare calendari a partire da qualsiasi anno.

Per motivi di lentezza di calcolo (il programma è in Basic) ho definito diversi anni base da cui si può partire per generare il calendario; per modificarli basta intervenire sulle linee 10-18 del listato.

Il programma, dopo aver visualizzato la presentazione e le istruzioni, chiede l'anno di cui si vuole il calendario e dopo poco visualizza il mese di gennaio dell'anno richiesto. Per vedere i mesi successivi basta premere un tasto qualsiasi, arrivati a dicembre il programma visualizza il gennaio dell'anno successivo. Per cambiare l'anno basta premere il tasto F1.

Il programma è di facile uso poiché le istruzioni sono presenti al suo interno e basta rispondere alle domande che esso pone.

MC

```
10 REM ********************************
11 REM *          CALENDARIO          *
12 REM ********************************
13 REM *     PER CAMBIARE ANNO BASE    *
14 REM *    RIMUOVERE LE REM VICINE    *
15 REM *     ALL'ANNO DESIDERATO ED    *
16 REM *   ALLE LINEE DEL PRECEDENTE   *
17 REM *           ANNO BASE.          *
18 REM ********************************
20 REMYY=1800
21 REMCC=2
30 REMYY=1900
31 REMCC=1
40 REMY=1950
41 REMC=0
50 YY=1582
60 CC=2
70 GOTO 1000
80 PRINTCHR$(147);CHR$(30):PRINT"●UESTO PROGRAMMA GENERA UN CALENDARIO A"
90 PRINT"PARTIRE DA QUALSIASI ANNO DOPO IL";YY;"."
100 PRINT"┐ER PARTIRE CON UN ANNO BASE DIVERSO"
110 PRINT"VEDI NEL LISTATO LE LINEE 10-18."
120 PRINT:PRINT"┐ER CAMBIARE L'ANNO MOSTRATO PREMI IL"
130 PRINT"TASTO ";CHR$(18);CHR$(5);"-1";CHR$(30);CHR$(146);".":PRINT
140 PRINT"↘L CALENDARIO INIZIA SEMPRE DA GENNAIO"
150 PRINT"E PER VEDERE I MESI SEGUENTI PREMERE"
160 PRINT"QUALSIASI TASTO. ♠RRIVATI A DICEMBRE SI"
170 PRINT"PASSA AL GENNAIO SUCCESSIVO.":PRINT:PRINT
180 INPUT"●UALE ANNO VUOI";A$:Y=VAL(A$)
200 DA=365:IFY/4=INT(Y/4)THENDA=366
210 H=0:DB=DA
220 IFY>YYTHEN270
230 RESTORE
240 K=0
250 PRINTCHR$(147);CHR$(30);CHR$(14)
255 PRINT"↗ON E' POSSIBILE OTTENERE ANNI":PRINT"PRECEDENTI IL";YY;"."
260 GOTO180
270 Z=Y-YY:C=CC
280 FORI=1TOZ
290 C=C+1:YX=YY+I
300 IFYX/4=INT(YX/4)THENC=C+1
310 IFC>7THENC=C-7
320 NEXT
330 IFY/4=INT(Y/4)THENC=C-1:IFC<0THENC=7+C
340 READM$,M:S=LEN(M$)
350 IFM=3AND(Y/4=INT(Y/4))THENM=2
360 IFC>7ORC=7THENC=C-7
370 B=C
390 PRINTCHR$(147);CHR$(142):PRINTCHR$(18);TAB(8);Y;CHR$(157);" ";CHR$(146)
400 PRINTTAB(1);H;TAB(16);DB
410 PRINTTAB(10-(S/2))"↗";:FORI=1TOS:PRINT"─";:NEXTI:PRINT"↘"
420 PRINTTAB(10-(S/2))"│";M$;"│"
430 PRINTTAB(10-(S/2))"↘";:FORI=1TOS:PRINT"─";:NEXTI:PRINT"↗"
440 PRINT:PRINT" ";CHR$(18);"D ";CHR$(146);" L  M  M  G  V  S ":PRINT
450 PRINTSPC(3*B);
460 FORI=1TO31-M
470 PRINTI;:IFI>9THENPRINTCHR$(157);
480 IFPOS(0)>19THENPRINT:PRINT
490 NEXTI:PRINT
500 GETX$:IFX$=""THEN500
510 IFX$=CHR$(133)THEN230
520 C=B+3-M
530 IFM$<>"DICEMBRE"THEN590
540 RESTORE
550 Y=Y+1
560 H=0:DB=365:IFY/4=INT(Y/4)THENDB=366
570 DA=DB
580 GOTO340
590 DB=DB-(31-M):H=DA-DB
600 GOTO340
610 END
900 DATA GENNAIO,0,FEBBRAIO,3,MARZO,0,APRILE,1,MAGGIO,0,GIUGNO,1,LUGLIO,0
910 DATA AGOSTO,0,SETTEMBRE,1,OTTOBRE,0,NOVEMBRE,1,DICEMBRE,0
1000 POKE53281,0:POKE53280,0:POKE646,5:PRINTCHR$(14);CHR$(147):PRINT
1010 PRINTTAB(12);"↘─ ↘ICROCOMPUTER":PRINT:PRINTTAB(16);"PRESENTA":PRINT
1020 PRINTTAB(9);"┌──────────┐":PRINTTAB(8);"┌──────────────┐"
1030 PRINTTAB(8);"│└┘":POKE646,10:PRINT"─ALENDARIO  PERPETUO":CHR$(30);"└┘"
1040 PRINTTAB(8);"└──────────┘┘":PRINTTAB(9);"└──────────┘"
1050 :PRINT:PRINTTAB(19);"BY":PRINT:PRINT:POKE646,14:PRINTTAB(13);
1060 PRINT"↘ARCO  ♠VERONE":FORI=1TO6:PRINT:NEXTI:POKE646,5:PRINTTAB(13);
1070 PRINT"┐REMI UN TASTO"
1080 GETA$:IFA$=""THEN1080
1090 GOTO80
```