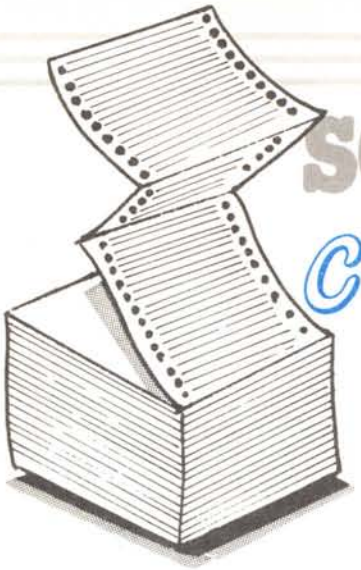

software C-128

a cura di Tommaso Pantuso

Grafici 3D

di Luciano Uzzo - Collegno (TO)

Come me, tanti studenti, hanno dovuto affrontare lo studio delle prime nozioni di trigonometria, (seni, coseni, tangenti, ecc.), per non parlare dei grafici tridimensionali.

Ora, grazie all'ausilio del C-128 anche a chi ha poca dimestichezza con la matematica, questi argomenti possono risultare addirittura divertenti.

Inoltre, vi assicuro che appena viste le immagini tridimensionali che si possono ottenere, ognuno sarà tentato di sviluppare delle proprie equazioni.

Il programma

Linee 10 - 160: PROGRAMMA PRINCIPALE
Linee 170 - 190: LA FUNZIONE
Linee 230 - 480: DRAW GRAFICO
Linee 490 - 510: FERMA L'IMMAGINE
Linee 520 - 600: MENU
Linee 610 - 820: INTESTAZIONE
Linee 830 - 850: MODIFICA FUNZIONE
Linee 860 - 950: SALVA GRAFICO
Linee 960 - 1060: CARICA GRAFICO
Linee 1070 - 1090: RESET

Le linee 10-160 saltano all'intestazione (linee 610-820), richiedono il colore del grafico della funzione, memorizzata nelle linee 170-190, preparano lo schermo ed immagazzinano le coordinate dei punti che comporranno il grafico.

Il loro calcolo richiede un certo tempo, che varia a seconda della complessità della funzione.

```
10 GOSUB620:SCNCLR:GRAPHIC1,1:GRAPHIC0
20 O=320:V=200:SU=1
30 IX=6:IZ=3
40 L=(O/IX/2)
50 P=(V/IZ/3)
60 PRINTTAB(12)"  GRAFICI 3D "
70 PRINTTAB(52)"  BY LUCIANO "
80 PRINTTAB(48)"COLORE GRAFICO (2-16)":INPUTCO
90 IFCO<2ORCO>16THENSCNCLR:GOTO60
100 FAST:TRAP800
110 REM **     CALCOLO...     **
120 IN=25
130 DIMG(L,P)
140 FORA=-P/2TOP/2
150 FORB=-L/2TOL/2
160 X=A*20/L:Z=B*20/P
170 REM **    LA  FUNZIONE    **
180 :
190 Y=INT(SIN(SQR(X*X+Z*Z+0.001))/(SQR(X*X+Z*Z+0.001))*75+15*1.3)/200
200 G(B+L/2,A+P/2)=Y*SU*V
210 NEXTB,A:COLOR1,CO
220 :
230 REM **        DRAW        **
240 FORZ=1TOP
250 XB=IX*Z
260 ZB=V/2+Z*IZ+IN*SU
270 XO=XB+IX
280 ZO=ZB-IZ-G(1,Z)
290 FORX=1TOL
300 XN=XB+X*IX
310 ZN=ZB-X*IZ-G(X,Z)
320 DRAW1,XO,ZO:DRAW1,XN,ZNTOXO,ZO
330 XO=XN:ZO=ZN
340 NEXTX,Z
350 :
360 REM **        DRAW        **
370 FORX=1TOL
380 XB=IX*X+P*IX
390 ZB=V/2-X*IZ+P*IZ+IN*SU
400 ZO=ZB-IZ-G(X,P-1)
410 XO=XB-IX
420 FORZ=0TOP-1
430 XN=XB-Z*IX
440 ZN=ZB-Z*IZ-G(X,P-Z)
450 DRAW1,XO,ZO:DRAW1,XN,ZNTOXO,ZO
460 XO=XN:ZO=ZN
470 NEXTZ,X:SLOW
480 FORI=0TO25:GRAPHIC2,0,I:FORA=0TO20:NEXTA,I
490 REM ** FERMA L'IMMAGINE **
500 COLOR1,2:CHAR1,13,24,"PREMI UN TASTO"
510 GETKEYA$
520 REM **        MENU        **
530 PRINT"(CLR)(DOWN)(DOWN)(DOWN)(DOWN)     MENU"
540 PRINT:PRINT"     (RVS) 1 (OFF) MODIFICA FUNZIONE"
550 PRINT:PRINT"     (RVS) 2 (OFF) SALVA GRAFICO SU DISCO"
560 PRINT:PRINT"     (RVS) 3 (OFF) CARICA GRAFICO DA DISCO"
```

Supermon 128

di Marco Lusini - Arezzo

Questo programma è stato pensato come estensione del già abbastanza completo monitor del C-128, al quale aggiunge numerosi comandi che permettono di gestire facilmente lo Z-80 usando i mnemonici dello Zilog, di accedere alla video ram e tante altre cosine interessanti.

Esso nasce da uno studio del tutto personale della ROM del monitor e dalla lettura dell'articolo apparso sul N. 56 di MC dove si spiegava come poter «accendere» lo Z-80. Poiché i programmatori della Commodore non hanno fornito i mezzi per usare questa e tante altre possibilità del 128, al povero «smanettone» frustrato non resta altro che armarsi di buona volontà e crearseli da solo; così cominciando dal disassembler per MSX di Dario Neddi pubblicato sul numero 45 di MC (unico riferimento in mio possesso per quanto riguardava l'Assembler dello Z-80) il Supermon 128 ha cominciato a prendere forma routine dopo routine.

Dopo aver caricato il programma in memoria è necessario entrare nel monitor digitando MONITOR o F8. A questo punto dare il comando diretto «G 2F00» ed il programma risponderà mostrando i registri dell'8502 come se fossimo appena entrati nel monitor con la differenza che ora sono disponibili tutti i nuovi comandi e banchi di memoria.

I Banchi Speciali

Per prima cosa parliamo dei nuovi banchi di memoria divenuti ora accessibili; infatti, modificando le routine per la gestione della memoria del 128, è possibile usare la RAM dell'8563 (processore video) come se fosse mappata nel secondo banco (N.B. Poiché l'8563 vede al massimo 16K, dopo $4000 le locazioni si ripetono); nel terzo banco troviamo la memoria vista dallo Z-80 compresa quindi la ROM interna del microprocessore e infine nel decimo possiamo accedere alla RAM del disco, previa inizializzazione

```
570 PRINT:PRINT"    (RVS) 4 (OFF) FINE"
580 FORI=25TO0STEP-1:GRAPHIC2,0,I:FORA=0TO20:NEXTA,I:GRAPHIC0
590 GETKEYA$:A=VAL(A$):IFA<1ORA>4THEN590
600 ONAGOTO830,860,960,1070
610 REM **   INTESTAZIONE   **
620 COLOR0,1:COLOR4,1
630 DOUNTILJ=6:READA$:A$="          "+A$:PRINT"(CLR)(WHT)(DOWN)(DOWN)(DOWN)(DOWN
)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)(DOWN)";J=J+1
640 COLOR5,13:PRINT"(UP)"A$:GOSUB770
650 COLOR5,16:PRINT"(UP)"A$:GOSUB770
660 COLOR5,2:PRINT"(UP)"A$:GOSUB780:IFJ=1ORJ=4THENGOSUB780
670 COLOR5,16:PRINT"(UP)"A$:GOSUB770
680 COLOR5,13:PRINT"(UP)"A$:GOSUB770
690 COLOR5,12:PRINT"(UP)"A$:GOSUB770
700 LOOP:SCNCLR:COLOR5,2:GOTO790
710 DATA "    UZZO LUCIANO
720 DATA "      SOFTWARE
730 DATA "      PRESENTA
740 DATA "    GRAFICI 3D
750 DATA "  WRITTEN IN  1986
760 DATA "  BY  UZZO LUCIANO
770 SLEEP.5:RETURN
780 SLEEP2:RETURN
790 RETURN
800 COLOR0,12:COLOR4,14:COLOR5,14
810 SLOW:SCNCLR:GRAPHIC0:PRINT"(DOWN) ERROR:"
820 END
830 REM **  MODIFICA FUNZ.  **
840 SCNCLR:PRINT"MODIFICA LA FUNZIONE E DAI IL RUN"
850 LIST170-199:END
860 REM ** SALVA IL GRAFICO **
870 SCNCLR:PRINT"INSERISCI UN DISCO E PREMI UN TASTO"
880 GETKEYA$
890 CATALOG"GG*"
900 PRINT:PRINT"NOME DEL GRAFICO (OMETTI LE 2 "CHR$(34)"G"CHR$(34)")"
910 OPEN1,0:INPUT#1,VN$:CLOSE1
920 IFLEN(VN$)>16THENPRINT"(UP)(UP)";:GOTO900
930 NN$="GG"+VN$
940 KEY1,"S"+CHR$(34)+NN$+CHR$(34)+",8,1C00,3FFF"+CHR$(13)+"X"+CHR$(13)+"GOTO480
(WHT)"+CHR$(13)
950 PRINT:PRINT"PREMI F1 PER SALVARE(BLK)":MONITOR
960 REM **  CARICA GRAFICO  **
970 SCNCLR:PRINT"INSERISCI UN DISCO E PREMI UN TASTO"
980 GETKEYA$
990 CATALOG"GG*"
1000 PRINT:PRINT"NOME DEL GRAFICO (OMETTI LE 2 "CHR$(34)"G"CHR$(34)")"
1010 OPEN1,0:INPUT#1,VN$:CLOSE1
1020 IFLEN(VN$)>16THENPRINT"(UP)(UP)";:GOTO900
1030 NN$="GG"+VN$
1040 KEY1,"L"+CHR$(34)+NN$+CHR$(34)+",8"+CHR$(13)+"X"+CHR$(13)+"GOTO1060(WHT)"+C
HR$(13)
1050 PRINT:PRINT"PREMI F1 PER CARICARE(BLK)":MONITOR
1060 GOTO480
1070 REM **   RESET SYSTEM   **
1080 COLOR0,12:COLOR4,14:COLOR5,14
1090 SYS16384
```

Dopo la visualizzazione del grafico si passa ad un menu composto da 4 opzioni:

1 MODIFICA FUNZIONE
2 SALVA GRAFICO (su disco)
3 CARICA GRAFICO (da disco)
4 RESET

con il comando U. Ad esempio: per scrivere una «®» in alto a sinistra nello schermo a 80 col. basterà digitare:

>20000 00 <ret>

per disassemblare la ROM dello Z-80:

P 30000 <ret>

e per esaminare il buffer 0 del disco:

" <ret>
M A0300 <ret>

I Nuovi Comandi

B-lock Sintassi: B

Mostra la traccia e il settore dell'ultimo blocco del disco coinvolto in operazioni di I/O.

Es: B
TRACK: +18 SECTOR: +01.

E-xecute Sintassi: E <indirizzo>

Inizia l'esecuzione di una routine nella memoria del disco all'indirizzo specificato. Deve essere stato precedentemente impartito il comando U.

Es: E EAA0

Esegue la routine di reset del 1541.

I-n Sintassi: I <indirizzo> <traccia> <settore>

Carica all'indirizzo il blocco del disco specificato dai 2 byte seguenti:

Es: I 1000 +18 +01

Carica in 1000-1100 il primo blocco della directory.

K-all Sintassi: K <indirizzo>

Esegue un programma in modo Z-80; se si desidera un ritorno al monitor il programma deve finire con RET. Al rientro sono mostrati i principali registri dello Z-80.

Es: K 0000

Resetta completamente il C-128.

O-ut Sintassi: O <indirizzo> <traccia> <settore>

Memorizza su disco il blocco indicato dall'indirizzo, traccia e settore. Usatelo con attenzione perché non chiede conferma.

Es: O 1000 +18 +01

Salva il primo blocco della directory.

P-rint Sintassi: P [<indirizzo 1>] [<indirizzo 2>]

Disassembla una routine in Assembler Z-80.

Es: P FFE0
! FFE0 F3 DI
.....

Mostra la routine che permette il passaggio dallo Z-80 all'8502.

Q-uit disk Sintassi: Q

Chiude i file aperti da U escludendo la memoria del disco.

Es: Q

U-se disk Sintassi: U

Inizializza il disco per poter usare il banco 10.

Es: U
00, OK, 00, 00

W-rite Z-80 Sintassi: W <indirizzo> <mnemonico> <operando>

Permette di scrivere routine in Assembler per lo Z-80. Gli operandi in esadecimale devono essere necessariamente preceduti da $; le istruzioni RST nn non necessitano di «$»

Es: W 01800 LD A, $00
W 01802 RST 08
.....

Z-80 Sintassi: Z

Mostra i registri principali dello Z-80.

Es: Z
Z-80
SP AF BC DE HL IX IY
,4000 0000 0000 0000 0000 0000 0000

, Sintassi: , <registri>

Serve a modificare il valore dei registri che lo Z-80 userà in seguito all'istruzione K.

Es: , 6000

Posiziona lo Stack Pointer a 6000.

!Sintassi:! <indirizzo> <byte>

Consente di modificare un listato Z-80 agendo direttamente sui codici delle istruzioni.

Es: ! 01800 0120D0

Descrizione tecnica

Il Supermon fa largo uso delle routine del monitor del 128, quindi converrà elencarne le principali con un breve commento. Ciò che segue è frutto delle mie fatiche e non deriva da nessun testo o rivista.

Nelle seguenti note indicherò con Ind. 1 il primo indirizzo che viene memorizzato in $60-$61-$62 dalle routine che necessitano un indirizzo, e con Ind. 2 il secondo indirizzo memorizzato in $66-$67-$68; in entrambi i casi i primi due byte indicano l'indirizzo e il terzo il banco a cui si riferisce.

(032E) Vettore per esecuzione comandi; normalmente punta a B006 ma modificandolo si possono aggiungere comandi propri.
B050 Esegue il comando R.
B0B8 Loop principale di accettazione comandi e di salto alle relative routine.
B0BC Entry Point per gli errori: stampa ? e torna al loop principale.
B0E3 Esegue il comando X.
B0FC Tabella con gli indirizzi delle routine dei comandi.
B11A Legge un byte puntato da (Ind.2), Y.
B12A Scrive un byte puntato da (Ind.2), Y.
B152 Esegue il comando M.
B1AB Esegue il comando >.
B1D6 Esegue il comando G.
B1DF Esegue il comando J.
B231 Esegue il comando C.
B234 Esegue il comando T.
B337 Esegue i comandi S, L, V.
B3DB Esegue il comando F.
B406 Esegue il comando A e il comando.
B599 Esegue il comando D.
B5D4 Disassembla 1 linea a partire da Ind.1.
B7A7 Accetta un byte o indirizzo da tastiera; è usata da quasi tutti i comandi per ricevere i propri parametri. Ritorna con Carry settato se non ha trovato alcun byte valido, altrimenti ritorna il dato in Ind.1.
B892 Stampa Ind.2 in esadecimale.
B8C2 Stampa il byte nell'accumulatore in forma esadecimale.
B8D2 Trasforma l'acc. nei due caratteri Ascii corrispondenti alla sua forma esadecimale e li immette nell'acc. e nel reg. X.
B901 Trasferisce Ind.1 in Ind.2.
B90E Fa la differenza tra Ind.2 e Ind.1 e salva il risultato in $60. (usata dai comandi con due indirizzi).
B922 Decrementa di 1 Ind.1.
B924 Sottrae l'acc. da Ind.1.
B950 Incrementa di 1 Ind.2
B952 Aggiunge l'acc. a Ind.2
B960 Decrementa di 1 Ind.2
BA90 Esegue il comando ®.

Segue ora l'elenco delle routine principali che compongono Supermon, anch'esse con un breve commento; come potrete notare Supermon è locato in una posizione un po' strana, cioè da $02000 a $02B00 e da $0E000 a $0FD00; ciò è dovuto al fatto che esso era nato come una semplice routine facilmente rilocabile e non aveva nessuna pretesa di diventare quello che è ora. Comunque la sua posizione ha il vantaggio di lasciare spazio per brevi programmi in Basic ($1C00-$1FFF) e per lunghi programmi in l.m. ($2B00-$DFFF), non interferendo allo stesso tempo con quei programmi che usano lo spazio per le versioni straniere ($1400-$17FF), i buffer dell'RS-232 ecc.

N.B. Tutte le routine che accedono allo Z-80, disabilitano prima il modo FAST, altrimenti si inchioda tutto quanto.

225D Esegue il comando p.
229B Prepara l'indirizzo per il banco speciale 2.
22AE Legge un byte in Z-80 mode: indirizzo in $2166-$2167 e dato in $2166.
22B9 Scrive un byte in Z-80 mode: indirizzo in $2166-$2167 e dato in $21BE.
22C3 Esegue la routine 22AB.
22C6 Esegue la routine 22B9.
22E6 Prepara l'indirizzo per il banco speciale 3.
22F5 Nuova routine INDFET.
2351 Nuova routine INDSTA.
23A8 Nuova routine INDCMP.
23E1 Codice Z-80 usato da K per aggiornare i registri e chiamare la subroutine.
2410 Esegue il comando K.
2449 Esegue il comando Z.
2492 Esegue il comando !.
2518 Esegue il comando U.
2573 Esegue il comando E.
259B Esegue il comando,.
25B6 Esegue il comando W.
27B8 Prende i parametri per I e O e apre i file necessari.
2805 Manda il comando di I/O per un blocco.
2828 Trasforma un byte in due cifre decimali, che sono immesse nell'acc. e nel reg. X.
284B Esegue il comando I. Termina con un salto all'interno del comando M.
287A Esegue il comando O.
28C3 Esegue il comando B.
28FA Esegue il comando Q.
2A00 Riconosce e esegue i comandi; se non trova il comando prosegue a $B006.
2F00 Inizializza il tutto.
E000 Tabella degli opcode e relativi mnemonici senza prefisso.
E975 Tabella degli opcode con prefisso $CB.
F313 Tabella degli opcode con prefisso $DD.
F519 Tabella degli opcode con prefisso $ED.
F725 Tabella degli opcode con prefisso $FD.
F92B Tabella degli opcode con prefisso $DDCB.
FAEF Tabella degli opcode con prefisso $FDCB.

Ancora due parole sul come creare banchi di memoria a proprio piacimento: per fare ciò è necessario modificare le routine INDFET, INDSTA, INDCMP in pagina 2 per farle puntare a routine scritte appositamente da noi, le quali controlleranno se il banco interessato è uno dei banchi speciali o no: se non lo è, si proseguirà con la normale routine, altrimenti si calcolerà l'indirizzo interesssato e si preleverà il byte nel modo più opportuno.

Ad esempio supponiamo di voler intercettare le operazioni di memoria nel banco 2, (cui corrisponde il valore $BF per il CR della MMU) per farle eseguire nella VIDEO RAM: bisogna modificare le routine nel modo indicato nel riquadro sottostante.

Utilizzando questa tenica non dovrebbe essere difficile per i fortunati possessori delle espansioni RAM per il C-128 scrivere delle routine che possano accedervi facilmente anche da Basic con delle semplici POKE e PEEK in un banco speciale. MC

```
      ORIGINALE                          MODIFICATA

INDFET 02A2 LDA #FF00              02A2 JMP MIAFET
       02A5 STX #FF00              02A5 STX #FF00
       02A8 TAX                    02A8 TAX
       02A9 LDA (#66),Y            02A9 LDA (#5E),Y
       02AB STX #FF00              02AB STX #FF00
       02AE RTS                    02AE RTS

INDSTA 02AF PHA                    02AF PHA
       02B0 LDA #FF00              02B0 JMP MIASTA
       02B3 STX #FF00              02B3 STX #FF00
       02B6 TAX                    02B6 TAX
       02B7 PLA                    02B7 PLA
       02B8 STA (#66),Y            02B8 STA (#6E),Y
       02BA STX #FF00              02BA STX #FF00
       02BD RTS                    02BD RTS

INDCMP 02BE PHA                    02BE PHA
       02BF LDA #FF00              02BF JMP MIACMP
       02C2 STX #FF00              02C2 STX #FF00
       02C5 TAX                    02C5 TAX
       02C6 PLA                    02C6 PLA
       02C7 CMP (#66),Y            02C7 CMP (#6E),Y
       02C9 STX #FF00              02C9 STX #FF00
       02CC RTS                    02CC RTS

    le nuove routine saranno:

SETADDR  CLC              ;calcola l'indirizzo tenendo conto di Y
         TYA
         ADC #00,X
         PHA              ;salva il byte low dell' indirizzo
         LDA #00
         ADC #01,X        ;calcola il byte hi.
         LDX #12
         JSR #CDCC        ; byte hi --> registro 12 dell' 8563
         PLA
         INX
         JMP #CDCC        ; byte low --> registro 13 dell' 8563

MIAFET   CPX #BF          ;il dato e' nel banco 2 ?
         BEQ VDFET        ; SI!
         LDA #FF00        ; NO!
         JMP #02A5        ; continua la indfet normale.
VDFET    LDX #02AA        ; prende in X il puntatore all' indirizzo
CMPENTRY JSR SETADDR      ; prepara l'indirizzo
         JMP #CDD8        ; riceve un byte di memoria dall' 8563.

MIASTA   CPX #BF          ; banco 2 ?
         BEQ VDSTA        ; SI!
         LDA #FF00        ; NO!
         JMP #02B3        ; continua indsta
VDSTA    LDX #02B3        ; prende il puntatore
         JSR SETADDR      ; prepara indirizzo.
         PLA              ; recupera dato per sta
         JMP #CDCA        ; scrive un byte nella memoria dell'8563.

MIACMP   CPX #6F
         BEQ VDCMP
         LDA #FF00
         JMP #02C2
VDCMP    LDX #02C8        ; prende il puntatore
         JSR CMPENTRY     ; legge il byte nella ram dell' 8563
         STA TEMP         ; lo memorizza
         PLA              ; recupera il valore da cofrontare
         CMP TEMP         ; confronta.
         RTS
```