Superlist per CBM 64

di Michele De Angelis - Roma

Lavorando con il 64 mi sono spesso rammaricato della poca leggibilità dei listati che, se pure non incide in maniera particolare allorché sono copie di lavoro dei propri programmi, diventa una vera e propria seccatura quando si studia un programma altrui, magari al fine di imparare qualche nuovo "trucchetto".

In questo stato d'animo mi sono imbattuto, tempo fa, in un programma di Valter Di Dio, pubblicato sul numero 17 di MC, che proponeva un "superlist per Apple II" e mi proposi di realizzare qualcosa di simile anche per il 64; il programma che segue ne è il risultato.

Esso serve per avere su stampante listati in Basic; non è un programma originale, poiché segue la falsariga di quello di Valter, ma ho aggiunto di mio la giustificazione a destra dei numeri di riga e l'apertura automatica del file di stampa tramite un comando diretto (si batte '@ + return').

Le principali caratteristiche del superlist per 64, oltre alla giustificazione a destra dei numeri di riga, sono: a) la possibilità di definire il numero di caratteri per riga; b) la definizione a priori del numero di righe per la prima pagina, in modo da poter mettere, eventualmente, una testata od un commento; c) la scelta del numero di righe per le pagine successive al fine di saltare di quattro righe in corrispondenza della perforazione fra le pagine dei moduli continui; d) la possibilità di mettere uno spazio prima e dopo ogni parola chiave; e) andando a capo non si scrive sotto i numeri di riga; f) viene implementata l'indentazione dei cicli FOR-NEXT, cioè ogni qual volta incontra un FOR si sposta di due caratteri a destra rispetto al resto del listato e quando trova un NEXT ritorna di due caratteri a sinistra. A tale proposito non viene riconosciuta una chiusura del tipo NEXT A,B ed in questo caso si chiude un solo ciclo; comunque per ovviare basta sostituire l'istruzione doppia con due semplici.

```
   10  rem  ******************************
   12  rem  *                            *
   14  rem  *        "SuPerlist"         *
   16  rem  *                            *
   18  rem  *             di             *
   20  rem  *                            *
   22  rem  *    "Michele De Angelis"    *
   24  rem  *                            *
   26  rem  ******************************
   28  data 169,11,141,2,3,169,192,141,3,3,96,32,96,165,134,122
   30  data 132,123,160,0,185,0,2,201,64,240,3,76,138,164,32,231
   32  data 255,169,0,32,189,255,169,1,162,4,160,7,32,186,255,32
   34  data 192,255,162,1,32,201,255,165,43,133,249,198,249,165,44,133
   36  data 250,32,120,193,32,125,193,169,80,133,252,32,226,192,32,233
   38  data 192,208,14,32,226,192,32,233,192,208,9,32,17,193,76,235
   40  data 193,32,226,192,32,7,193,32,226,192,32,233,192,170,32,226
   42  data 192,32,233,192,32,149,193,169,32,32,132,193,32,226,192,32
   44  data 233,192,166,212,224,1,240,42,165,9,16,38,41,127,72,201
   46  data 1,208,14,32,112,193,104,32,43,193,230,251,230,251,76,187
   48  data 192,201,2,208,3,32,105,193,32,112,193,104,32,43,193,76
   50  data 187,192,166,251,232,32,138,193,32,132,193,32,226,192,32,233
   52  data 192,208,3,76,75,192,166,212,224,1,240,13,165,9,16,9
   54  data 32,36,193,32,245,192,76,187,192,32,132,193,32,245,192,76
   56  data 187,192,230,249,208,2,230,250,96,160,0,177,249,133,9,32
   58  data 132,230,165,9,96,165,211,201,69,48,11,32,7,193,162,7
   60  data 32,138,193,32,112,193,96,32,215,170,32,125,193,198,252,208
   62  data 245,169,83,133,252,32,215,170,32,215,170,32,215,170,32,215
   64  data 170,76,125,193,41,127,72,32,93,193,104,170,169,157,133,253
   66  data 169,160,133,254,160,0,177,253,201,128,48,3,202,48,9,230
   68  data 253,208,2,230,254,76,52,193,169,32,32,132,193,200,177,253
   70  data 72,32,130,193,104,16,246,169,32,32,132,193,96,201,1,208
   72  data 4,230,251,230,251,201,2,208,6,198,251,240,2,198,251,96
   74  data 166,251,240,4,32,138,193,96,169,1,133,251,96,169,6,133
   76  data 211,96,41,127,32,71,171,230,211,96,72,169,32,32,132,193
   78  data 202,208,250,104,96,201,0,208,36,24,224,10,176,20,72,169
   80  data 32,32,71,171,32,71,171,32,71,171,32,71,171,104,76,227
   82  data 193,96,24,224,100,176,6,72,169,32,76,164,193,24,201,3
   84  data 240,4,144,6,176,10,224,232,176,6,72,169,32,76,167,193
   86  data 24,201,39,240,4,144,6,176,10,224,16,176,6,72,169,32
   88  data 76,170,193,32,205,189,169,0,133,212,96,32,231,255,76,134,227
   90  for K = 49152 to 49648: read a
   92    cK = cK + a: Poke K,a: next
   94  if cK <> 63617 then  Print "■■■■  errore nei data": end
  128  Print "■■■■■"
  130  Print "Quanti caratteri Per riga ?";: Gosub 10000
  132  z =  val (cm$): if z = 0 then 136
  134  Poke 49400,z
  136  Print "■■■Quante righe nella Prima Pagina ?";: Gosub 10000
  138  z =  val (cm$): if z = 0 then 142
  140  Poke 49224,z
  142  Print "■■■Quante righe nelle Pagine successive?";: Gosub 10000
  144  z =  val (cm$): if z = 0 then 148
  146  Poke 49426,z
  148  Print "■■■Vuoi cambiare qualcosa ?(s/n)
  150  Get  c$: if c$ = "" then 150
  152  if c$ = "s" then  Goto  128
  154  sYs 49152: new
10000  cm$ = ""
10002  Print "■■■";
10005  Get z$: if z$ = "" then 10005
10010  z =  asc (z$): if z > 57 then 10005
10015  if z = 13 then  Print "  ": return
10080  if z > 47 then cm$ = cm$ + z$: Print z$;
10090  if z = 20 and z1 then cm$ =  left$ (cm$,z1 - 1): Print z$;
10100  z1 =  len (cm$): if z1 < 2 then 10002
10120  if z1 then  Print "  ": return
10140  Goto  10002
```

Listato del programma in Basic, eseguito con il SUPERLIST.

Come funziona

I primi 11 byte, da &C000 a $C00A, formano una routine a se stante che serve unicamente a spostare i puntatori alla routine di esecuzione dei comandi, facendo in modo che, ogni qual volta si batta un comando in modo diretto, venga prima controllato che si tratti di una '@', in caso affermativo si salta all'inizio del programma, altrimenti si ritorna al sistema operativo [$A48A].

I byte da $C01E $C036 contengono l'istruzione di apertura del file di stampa; in Basic si scriverebbe "open 1,4,7:cmd 1".

Da $C037 incomincia il programma vero e proprio; come prima cosa si ricopiano in $F9 e $FA i puntatori all'inizio del programma Basic che si trovano in $2B e $2C.

Fatto questo, si scorre il programma da listare e si controllano i primi due byte, essi contengono i puntatori alla successiva istruzione e se sono a zero vuol dire che il programma è finito e si deve ritornare al Basic [JMP $E386], altrimenti si leggono i due byte successivi che contengono il numero di riga e la subroutine a $C195 provvede a stamparlo mettendoci davanti tanti spazi quanti sono necessari a portarlo ad una lunghezza di cinque caratteri.

A questo punto bisogna leggere e stampare il prossimo byte e le cose si complicano un pochino poiché esso può contenere un ASCII od una istruzione tokenizzata, cioè trasformata in un solo numero superiore a 128; se il contenuto del byte è inferiore a questo valore esso è certamente un ASCII e tutto va bene, in caso contrario non è detto che sia il codice di un'istruzione poiché il C 64 usa per i caratteri di controllo, quelli grafici e le maiuscole proprio dei valori superiori a 128.

```
,C000 A9 0B     LDA #$0B
,C002 8D 02 03  STA $0302
,C005 A9 C0     LDA #$C0
,C007 8D 03 03  STA $0303
,C00A 60        RTS
,C00B 20 60 A5  JSR $A560
,C00E 86 7A     STX $7A
,C010 84 7B     STY $7B
,C012 A0 00     LDY #$00
,C014 B9 00 02  LDA $0200,
,C017 C9 40     CMP #$40
,C019 F0 03     BEQ $C01E
,C01B 4C 8A A4  JMP $A48A
,C01E 20 E7 FF  JSR $FFE7
,C021 A9 00     LDA #$00
,C023 20 BD FF  JSR $FFBD
,C026 A9 01     LDA #$01
,C028 A2 04     LDX #$04
,C02A A0 07     LDY #$07
,C02C 20 BA FF  JSR $FFBA
,C02F 20 C0 FF  JSR $FFC0
,C032 A2 01     LDX #$01
,C034 20 C9 FF  JSR $FFC9
,C037 A5 2B     LDA $2B
,C039 85 F9     STA $F9
,C03B C6 F9     DEC $F9
,C03D A5 2C     LDA $2C
,C03F 85 FA     STA $FA
,C041 20 78 C1  JSR $C178
,C044 20 7D C1  JSR $C17D
,C047 A9 50     LDA #$50
,C049 85 FC     STA $FC
,C04B 20 E2 C0  JSR $C0E2
,C04E 20 E9 C0  JSR $C0E9
,C051 D0 0E     BNE $C061
,C053 20 E2 C0  JSR $C0E2
,C056 20 E9 C0  JSR $C0E9
,C059 D0 09     BNE $C064
,C05B 20 11 C1  JSR $C111
,C05E 4C EB C1  JMP $C1EB
,C061 20 E2 C0  JSR $C0E2
,C064 20 07 C1  JSR $C107
,C067 20 E2 C0  JSR $C0E2
,C06A 20 E9 C0  JSR $C0E9
,C06D AA        TAX
,C06E 20 E2 C0  JSR $C0E2
,C071 20 E9 C0  JSR $C0E9
,C074 20 95 C1  JSR $C195
,C077 A9 20     LDA #$20
,C079 20 84 C1  JSR $C184
,C07C 20 E2 C0  JSR $C0E2
,C07F 20 E9 C0  JSR $C0E9
,C082 A6 D4     LDX $D4
,C084 E0 01     CPX #$01
,C086 F0 2A     BEQ $C0B2
,C088 A5 09     LDA $09
,C08A 10 26     BPL $C0B2
,C08C 29 7F     AND #$7F
,C08E 48        PHA
,C08F C9 01     CMP #$01
,C091 D0 0E     BNE $C0A1
,C093 20 70 C1  JSR $C170
,C096 68        PLA
,C097 20 2B C1  JSR $C12B
,C09A E6 FB     INC $FB
,C09C E6 FB     INC $FB
,C09E 4C BB C0  JMP $C0BB
,C0A1 C9 02     CMP #$02
,C0A3 D0 03     BNE $C0A8
,C0A5 20 69 C1  JSR $C169
,C0A8 20 70 C1  JSR $C170
,C0AB 68        PLA
,C0AC 20 2B C1  JSR $C12B
,C0AF 4C BB C0  JMP $C0BB
,C0B2 A6 FB     LDX $FB
,C0B4 E8        INX
,C0B5 20 8A C1  JSR $C18A
,C0B8 20 84 C1  JSR $C184
,C0BB 20 E2 C0  JSR $C0E2
,C0BE 20 E9 C0  JSR $C0E9
,C0C1 D0 03     BNE $C0C6
,C0C3 4C 4B C0  JMP $C04B
,C0C6 A6 D4     LDX $D4
,C0C8 E0 01     CPX #$01
,C0CA F0 0D     BEQ $C0D9
,C0CC A5 09     LDA $09
,C0CE 10 09     BPL $C0D9
,C0D0 20 24 C1  JSR $C124
,C0D3 20 F5 C0  JSR $C0F5
,C0D6 4C BB C0  JMP $C0BB
,C0D9 20 84 C1  JSR $C184
,C0DC 20 F5 C0  JSR $C0F5
,C0DF 4C BB C0  JMP $C0BB
,C0E2 E6 F9     INC $F9
,C0E4 D0 02     BNE $C0E8
,C0E6 E6 FA     INC $FA
,C0E8 60        RTS
,C0E9 A0 00     LDY #$00
,C0EB B1 F9     LDA ($F9),Y
,C0ED 85 09     STA $09
,C0EF 20 84 E6  JSR $E684
,C0F2 A5 09     LDA $09
,C0F4 60        RTS
,C0F5 A5 D3     LDA $D3
,C0F7 C9 40     CMP #$40
,C0F9 30 0B     BMI $C106
,C0FB 20 07 C1  JSR $C107
,C0FE A2 07     LDX #$07
,C100 20 8A C1  JSR $C18A
,C103 20 70 C1  JSR $C170
,C106 60        RTS
,C107 20 D7 AA  JSR $AAD7
,C10A 20 7D C1  JSR $C17D
,C10D C6 FC     DEC $FC
,C10F D0 F5     BNE $C106
,C111 A9 50     LDA #$50
,C113 85 FC     STA $FC
,C115 20 D7 AA  JSR $AAD7
,C118 20 D7 AA  JSR $AAD7
,C11B 20 D7 AA  JSR $AAD7
,C11E 20 D7 AA  JSR $AAD7
,C121 4C 7D C1  JMP $C17D
,C124 29 7F     AND #$7F
,C126 48        PHA
,C127 20 5D C1  JSR $C15D
,C12A 68        PLA
,C12B AA        TAX
,C12C A9 9D     LDA #$9D
,C12E 85 FD     STA $FD
,C130 A9 A0     LDA #$A0
,C132 85 FE     STA $FE
,C134 A0 00     LDY #$00
,C136 B1 FD     LDA ($FD),Y
,C138 C9 80     CMP #$80
,C13A 30 03     BMI $C13F
,C13C CA        DEX
,C13D 30 09     BMI $C148
,C13F E6 FD     INC $FD
,C141 D0 02     BNE $C145
,C143 E6 FE     INC $FE
,C145 4C 34 C1  JMP $C134
,C148 A9 20     LDA #$20
,C14A 20 84 C1  JSR $C184
,C14D C8        INY
,C14E B1 FD     LDA ($FD),Y
,C150 48        PHA
,C151 20 82 C1  JSR $C182
,C154 68        PLA
,C155 10 F6     BPL $C14D
,C157 A9 20     LDA #$20
,C159 20 84 C1  JSR $C184
,C15C 60        RTS
,C15D C9 01     CMP #$01
,C15F D0 04     BNE $C165
,C161 E6 FB     INC $FB
,C163 E6 FB     INC $FB
,C165 C9 02     CMP #$02
,C167 D0 06     BNE $C16F
,C169 C6 FB     DEC $FB
,C16B F0 02     BEQ $C16F
,C16D C6 FB     DEC $FB
,C16F 60        RTS
,C170 A6 FB     LDX $FB
,C172 F0 04     BEQ $C178
,C174 20 8A C1  JSR $C18A
,C177 60        RTS
,C178 A9 01     LDA #$01
,C17A 85 FB     STA $FB
,C17C 60        RTS
,C17D A9 06     LDA #$06
,C17F 85 D3     STA $D3
,C181 60        RTS
,C182 29 7F     AND #$7F
,C184 20 47 AB  JSR $AB47
,C187 E6 D3     INC $D3
,C189 60        RTS
,C18A 48        PHA
,C18B A9 20     LDA #$20
,C18D 20 84 C1  JSR $C184
,C190 CA        DEX
,C191 D0 FA     BNE $C18D
,C193 68        PLA
,C194 60        RTS
,C195 C9 00     CMP #$00
,C197 D0 24     BNE $C1BD
,C199 18        CLC
,C19A E0 0A     CPX #$0A
,C19C B0 14     BCS $C1B2
,C19E 48        PHA
,C19F A9 20     LDA #$20
,C1A1 20 47 AB  JSR $AB47
,C1A4 20 47 AB  JSR $AB47
,C1A7 20 47 AB  JSR $AB47
,C1AA 20 47 AB  JSR $AB47
,C1AD 68        PLA
,C1AE 4C E3 C1  JMP $C1E3
,C1B1 60        RTS
,C1B2 18        CLC
,C1B3 E0 64     CPX #$64
,C1B5 B0 06     BCS $C1BD
,C1B7 48        PHA
,C1B8 A9 20     LDA #$20
,C1BA 4C A4 C1  JMP $C1A4
,C1BD 18        CLC
,C1BE C9 03     CMP #$03
,C1C0 F0 04     BEQ $C1C6
,C1C2 90 06     BCC $C1CA
,C1C4 B0 0A     BCS $C1D0
,C1C6 E0 E8     CPX #$E8
,C1C8 B0 06     BCS $C1D0
,C1CA 48        PHA
,C1CB A9 20     LDA #$20
,C1CD 4C A7 C1  JMP $C1A7
,C1D0 18        CLC
,C1D1 C9 27     CMP #$27
,C1D3 F0 04     BEQ $C1D9
,C1D5 90 06     BCC $C1DD
,C1D7 B0 0A     BCS $C1E3
,C1D9 E0 10     CPX #$10
,C1DB B0 06     BCS $C1E3
,C1DD 48        PHA
,C1DE A9 20     LDA #$20
,C1E0 4C AA C1  JMP $C1AA
,C1E3 20 CD BD  JSR $BDCD
,C1E6 A9 00     LDA #$00
,C1E8 85 D4     STA $D4
,C1EA 60        RTS
,C1EB 20 E7 FF  JSR $FFE7
,C1EE 4C 86 E3  JMP $E386
```

Disassemblato del programma SUPERLIST

Allora, poiché tutti gli ASCII debbono essere racchiusi fra virgolette a meno che non siano variabili (e non si può usare un carattere maiuscolo, grafico o di controllo come variabile), andiamo a controllare che non siano state aperte le virgolette; questo ce lo dice il flag posto in $D4: se esso è a 1 vuol dire che il numero superiore a 128 è un codice ASCII, altrimenti è un'istruzione.

Nel primo caso viene stampato direttamente [$C184], se è un'istruzione controlliamo che essa non sia un FOR o un NEXT (in tal caso bisogna aggiornare il margine sinistro tramite il contatore posto in $FB), quindi si va alla routine in $C12B che ricerca la stringa corrispondente all'istruzione nella tabella del Basic posta a partire da $A09D e la stampa mettendo subito prima e subito dopo uno spazio per isolarla.

Prima di ogni ASCII e dopo ogni parola chiave viene fatto, tramite il contatore del numero di caratteri per riga posto in $D3, il controllo del superamento del margine, in caso affermativo si va a capo e si controlla l'arrivo al bordo inferiore della pagina (contatore in $FC).

Da quanto detto ne consegue che, poiché il controllo del margine viene fatto DOPO le parole chiave, il carattere più a destra può trovarsi stampato fino ad otto posizioni oltre il margine fissato e bisogna tener conto di ciò nel decidere il numero di caratteri per riga.

Esso è contenuto in $C0F8 ed il programma di lancio in Basic provvede a 'pokare' in quella locazione il numero prescelto, come fa per il numero di righe per la prima pagina (in $C048) e per quello delle pagine successive (in $C112).

Come funziona

Coloro che preferiscono lavorare in Assembler debbono solo ricopiare il dissassemblato e porre, nelle locazioni sopra indicate, i valori da loro prescelti per i caratteri e le righe se sono diversi da quelli da me usati, altrimenti il programma in Basic risolve tutti i problemi.

Una volta digitato, e dopo averlo salvato, dato il RUN si deve attendere qualche secondo per il trasferimento dei dati in memoria, dopo di che se c'è qualcosa che non va il programma termina avvisando "errore nei data" e può essere listato per le correzioni, altrimenti saranno poste le domande sulla scelta del formato di stampa.

La routine di input è ottimizzata e accetta solo due caratteri numerici passando automaticamente alla prossima domanda non appena viene battuto il secondo; logicamente se non si vuol cambiare nulla basta dare semplicemente il return.

Dopo questa fase il programma è pronto per essere utilizzato e, allorché avrete in memoria un programma in Basic, vi basterà battere '@ + RETURN' (accertatevi che la stampante sia accesa!) per avere un listato finalmente chiaro e leggibile.

```
    1   rem  *** questo  programma ***
    2   rem  *** serve per provare ***
   10   rem  *** il   list  editor ***
   20   print "s"
   30   input "quanti caratteri per riga";
        cr
   60   input "quante righe per pagina (50
        )";l:l =  val (l$)
   70   if l then l = 50
   90   rem
  100   for a = 1 to 100
  120     print "prova"
  130     for b = 1 to 10 step 2
  140       for c = 1 to 2
  150         poke 53281,0: rem  prova di
              a capo dentro il ciclo
              for-next
  170         g =  peek (43): rem  prova d
              i indentazione di righe
               che iniziano per varia
              bile
  180         a = f * 3
  190         if f then 180
  200       next c
  999       print "prova"
 1000     next b
 9999     print "prova"
10000   next a
20000   end
```

```
1 rem *** questo  programma ***
2 rem *** serve per provare ***
10 rem *** il   list  editor ***
20 print"s"
30 input"quanti caratteri per riga";cr
60 input"quante righe per pagina (50)";l
:l=val(l$)
70 iflthenl=50
90 rem
100 fora=1to100
120 print"prova"
130 forb=1to10step2
140 forc=1to2
150 poke53281,0:rem prova di a capo dent
ro il ciclo for-next
170 g=peek(43):rem prova di indentazione
 di righe che iniziano per variabile
180 a=f*3
190 iffthen180
200 nextc
999 print"prova"
1000 nextb
9999 print"prova"
10000 nexta
20000 end
```

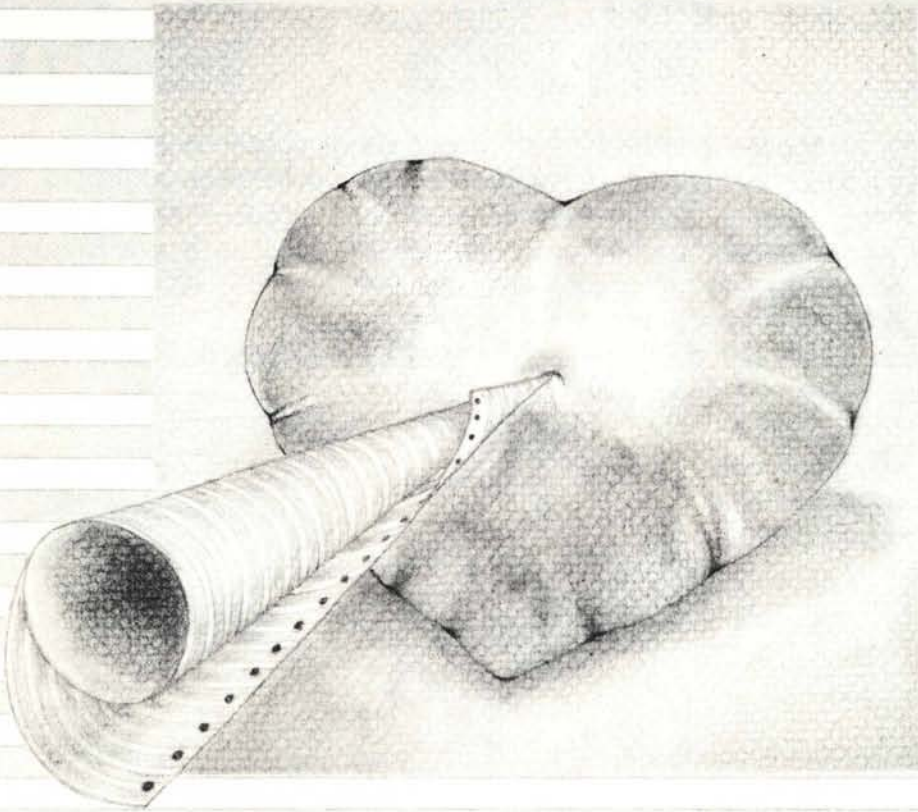
Due programmi listati rispettivamente con il SUPERLIST e senza di esso.

Locazioni in pagina zero usate dal programma Superlist

$D3 (#211) Contatore di caratteri per riga
$D4 (#212) Flag di virgolette
$F9 (#249) Puntatore inizio del Basic Lo-B.
$FA (#250) Puntatore inizio del Basic Hi-B.
$FB (#251) Contatore d'indentazione per cicli FOR NEXT
$FC (#252) Contatore del numero di righe per pagina
$FD (#253) Puntatore inizio tabella parole chiave del Basic Lo-B
$FE (#254) Puntatore inizio tabella parole chiave del Basic Hi-B
$09 (# 9) Loc. di comodo usata a volte per salvare un dato

Routine del sistema operativo e del Kernal

$FFE7 Chiude tutti i canali e i file
$FFBD Imposta il nome del file
$FFBA Imposta gli indirizzi primario, secondario e logico del file
$FFC0 Apre un file logico
$FFC9 Apre il canale di output
$A560 Riceve in input una linea
$E684 Test di virgolette
$AAD7 Effettua un carriage return
$AB47 Stampa il contenuto dell'accumulatore
$BDCD Stampa il numero di linea
$E386 Basic warm restart



Colpitele al cuore

MANNESMANN
TALLY

le stampanti che colpiscono al cuore le vostre esigenze

MT 80 PLUS/PC MT 85 MT 86

silverstar
componenti e periferiche

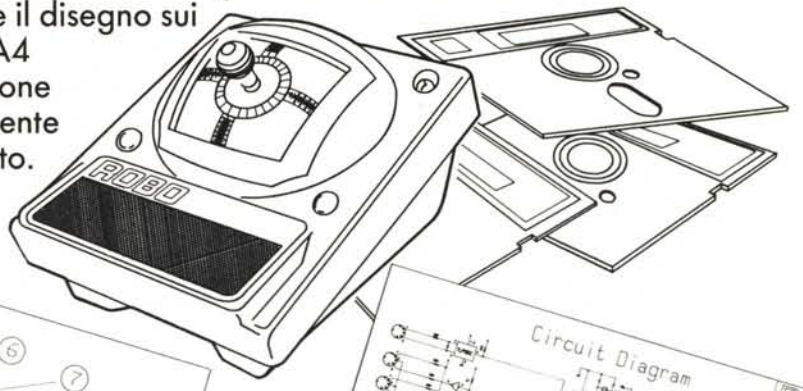
Sede: 20146 Milano - Via dei Gracchi, 20
Tel. (02) 4996 (12 linee) - Telex 332187
40122 Bologna - Via del Porto, 30
Tel. (051) 522231

00198 Roma - Via Paisiello, 30
Tel. (06) 8448841 (5 linee) - Telex 610511
10139 Torino - P.za Adriano, 9
Tel. (011) 443275/6 - 442321 - Telex 220181

SISTEMI DI DISEGNO CON COMPUTER

ROBO 500 ROBO 1000 ROBO 1500

Oggi ben tre soluzioni differenti Vi consentono di disegnare con il personal computer APPLE*. I sistemi offerti dalla Robo Vi permettono di realizzare una workstation completa, economica e su misura per le Vostre esigenze. Schemi elettrici, elettronici, circuiti stampati, impiantistica generale o particolare, diagrammi, architettura, meccanica, ingegneria, costruzioni: tutto ciò può essere disegnato con il sistema più adatto scelto tra il Robo 500, 1000 o 1500 in maniera semplice ed efficace anche senza conoscere il computer.
Ogni sistema acquisisce infatti dati da librerie già esistenti o costruite dall'utente e tramite apposito software può plottare il disegno sui plotter più diffusi dal formato A4 al formato A0 con una definizione illimitata e determinata unicamente dalla qualità del plotter utilizzato.



ADW Studio

*Sistema richiesto Apple][+, IIe 64K, doppio disco, monitor