Due programmi per chi programma

Questa puntata del software Apple è riservata ai programmatori, sia a quelli che programmano in Basic, sia ai programmatori in linguaggio macchina. Infatti presentiamo due utility abbastanza interessanti: la prima permette di attivare gli attributi video (INVERSE, NORMAL, FLASH) direttamente da tastiera e dentro agli stessi programmi; la seconda serve invece per rilocare (il più possibile automaticamente) un programma in linguaggio macchina di cui non si abbia il sorgente.

Attributi video da tastiera

Questo «miniprogramma» in linguaggio macchina permette di ottenere tutti e tre gli attributi video direttamente dentro una stessa stringa di output, semplicemente battendo da tastiera Control I per INVERSE, Control N per NORMAL e Control F per FLASH. Inoltre, avendo la utility attivata, si possono vedere, già nel listato, le stampe che contengono parole in inverso o in flash, esattamente come verranno sullo schermo.

La utility non fa altro che intercettare il carattere inviato al video e, se questo è uno dei tre carateri di controllo, attivare, tramite il contenuto della locazione 50, la relativa visualizzazione.

Data l'esigua occupazione di memoria (e un po' come esercizio programmatorio) si è deciso di mettere il programma direttamente dentro a quello in Basic.

Come si fa

Il programma base è quello di figura 1, come si vede è lungo solo trenta byte ed inoltre può essere trasferito, e lavorare, in qualsiasi zona libera della memoria. Una volta determinata la posizione in cui si vuole il programma, si deve avvisare il DOS di modificare i puntatori alla routine di stampa in modo che guardino al nostro programmino. Questo si ottiene scrivendo la parte bassa dell'indirizzo nella locazione 54, quella alta nella 55 e poi effettuando una CALL alla routine 1002 che effettua il cosiddetto ricollegamento del DOS. Per chi lavora sotto proDOS il comando è stato semplificato in PR#,A indirizzo (indirizzo che può anche essere scritto in esadecimale se preceduto dal simbolo $).

Per nascondere la routine dentro al programma che la utilizza occorre seguire le seguenti istruzioni:

1) Scrivere una riga come la seguente:

10 ····· REM.....

i puntini dopo il REM sono 30; ATTENZIONE, non mettere lo spazio tra il REM e il primo puntino!

2) Caricare in memoria nella zona 300 il codice oggetto di figura 1 (quindi CALL-151, poi 300:C9 8E D0 ... 4C F0 FD <return>).

```
0300-   C9 8E     CMP   #$8E
0302-   D0 05     BNE   $0309
0304-   A9 FF     LDA   #$FF
0306-   85 32     STA   $32
0308-   60        RTS
0309-   C9 86     CMP   #$86
030B-   D0 05     BNE   $0312
030D-   A9 7F     LDA   #$7F
030F-   85 32     STA   $32
0311-   60        RTS
0312-   C9 89     CMP   #$89
0314-   D0 05     BNE   $031B
0316-   A9 3F     LDA   #$3F
0318-   85 32     STA   $32
031A-   60        RTS
031B-   4C F0 FD  JMP   $FDF0
```

Figura 1: Routine che permette di gestire gli attributi video (Flash ed Inverse) direttamente all'interno delle stringhe di stampa.

3) Battere:

·80B<300.31DM <return>

questo sposta il programma in linguaggio macchina al posto dei puntini dentro la REM. Se a questo punto tornate al Basic ed effettuate un LIST vi ritrovate con la seguente riga...

10 ····· REM - HLIN=SPEED=h DEL 2' - DIM=SPEED=DEL 2' - TEXT=SPEED=? DEL 2'L OVERFLOW h

Che altro non è che il codice del nostro programma, interpretato come se fosse Basic. Per eliminare l'antiestetica riga ci saranno utili quei cinque asterischi messi prima del REM.

4) Dal monitor battete:

·805:0 20 20 20 20

·ctrl C

5) tornate al Basic e riprovate il LIST. Della riga 10 è rimasto solo il numero di riga (10 appunto). Scrivete ora la chiamata al DOS:

20 POKE 54,11:POKE55,8:CALL 1002

o al proDOS

20 PRINT CHR$(4) «PR#,A$80B»

Provate ora a scrivere un'altra riga in questo modo:

60 PRINT «Questo è <F>FLASH<N> e questo <I>INVERSO<N>»

Dove i caratteri tra parentesi acute vanno battuti con il tasto Control premuto; salvate il programma con un nome qualsiasi, poi date il RUN, se tutto è stato fatto con cura, la scritta deve avere la parola FLASH lampeggiante e quella INVERSO in inverso. Se ora provate a listare il programma vedrete direttamente nel listato le scritte esattamente come appariranno in fase di RUN.

Nota: Avere il programma direttamente dentro a quello Basic permette di caricare la routine insieme al programma stesso senza dover utilizzare file binari o liste di DATA, se però si carica un altro programma con la routine ancora attiva, si rischia di inchiodare il sistema (niente paura, basta il RESET), prima di cambiare programma conviene allora battere PR#0 oppure premere direttamente il tasto RE-

SET. Chi volesse lasciare la routine all'indirizzo $300 (dec. 768) deve cambiare la riga 20 in:

```
20 POKE 54,0:POKE 55,3:CALL 1002
```

oppure, sotto proDOS, battere PR#, A$300 direttamente da tastiera o dentro al programma.

Nel caso che la routine sia allocata fuori dall'area Basic, rimane attiva fino alla pressione del RESET o all'esecuzione di un comando PR#.

Rilocatore di programmi in LM

Quando si scrive un programma in linguaggio macchina occorre sapere in partenza in quale zona della memoria questo dovrà risiedere; ciò perché, alcune delle istruzioni del microprocessore (ad esempio i salti) fanno riferimento ad una precisa locazione di memoria. Anche molte aree dati si trovano in posizioni predeterminate e, se qualcuno le sposta, il programma non è più in grado di trovarle.

Spesso, però, si presenta la necessità di trasferire un programma in una zona di memoria diversa da quella per cui era stato scritto (ad esempio perché si è allungato, oppure perché crea conflitti con le aree variabili del Basic). Trasferire un programma in linguaggio macchina, da una posizione di memoria ad un'altra, si dice «rilocare», ed è il classico «lavoro da certosino». Infatti, con il disassemblato in mano, occorre trovare tutte le istruzioni con indirizzamento assoluto, cioè tutte quelle che fanno riferimento ad una determinata locazione (ad esempio LDA $45FF che legge il contenuto della locazione $45FF).

Le istruzioni ad indirizzamento assoluto, o assoluto indicizzato, si riconoscono dalle altre perché sono lunghe tre byte: il primo byte è il codice del comando, gli altri due rappresentano l'indirizzo scritto «al contrario» (nell'esempio precedente FF 45).

Una volta individuate tutte le istruzioni a tre byte, bisogna ancora vedere quali sono quelle relative al blocco di programma che si sta spostando e quali, invece, sono relative al resto del programma, alla ROM del sistema operativo o ai soft-switch, e che, quindi, non devono essere toccate. Trovate le istruzioni da cambiare, occorre ancora calcolare, in base alla nuova posizione del programma, quale dovrà essere l'indirizzo definitivo e correggere opportunamente i due byte del campo indirizzo di ciascuna di queste istruzioni.

Tutto questo però, spesso, non basta; infatti anche alcune istruzioni a due byte intervengono, indirettamente, nel processo di rilocazione, sono quelle relative alla individuazione di tabelle e punti di entrata particolari. Ad esempio, se dobbiamo informare il sistema operativo che il prossimo INPUT deve venire da una routine del programma, il codice relativo sarà:

```
LDA #PARTE ALTA
STA INPUT ALTO
LDA #PARTE BASSA
STA INPUT BASSO
JMP DOS INPUT
```

Ovvero, si scrive in due locazioni in pagina zero il nuovo indirizzo e poi si avverte il DOS di andare a leggere la modifica. Le due istruzioni LDA #NN sono istruzioni a due soli byte, e quindi non sembrerebbero da rilocare, ma i valori Parte Alta e Parte Bassa sono relativi al nuovo punto di entrata del programma e vanno, perciò, modificati ugualmente.

```
0800- A9 FF 85 31 A0 00 20 ED
0808- 08 20 F9 08 85 EB 84 EC
0810- A0 13 20 ED 08 20 F9 08
0818- 85 ED 84 EE 38 A5 ED E5
0820- EB 85 CE A5 EE E5 EC 85
0828- CF 90 10 A5 CE 49 FF 69
0830- 00 85 CE A5 CF 49 FF 69
0838- 00 85 CF A0 26 20 ED 08
0840- 20 F9 08 85 FB 84 FC A0
0848- 39 20 ED 08 20 F9 08 85
0850- FD 84 FE A0 4C 20 ED 08
0858- 20 F9 08 38 E5 EB 85 F9
0860- 98 E5 EC 85 FA 18 A5 F9
0868- 69 00 85 F9 A5 FA 69 20
0870- 85 FA A0 5F 20 ED 08 20
0878- 0C FD C9 CD 66 FF A9 20
0880- 85 3B A9 00 85 3A AD 00
0888- C0 C9 83 F0 5C 20 D0 F8
0890- A0 02 C4 2F D0 42 38 88
0898- B1 3A E5 FB C8 B1 3A E5
08A0- FC 90 35 A5 FD 88 F1 3A
08A8- A5 FE C8 F1 3A 90 29 24
08B0- FF 10 07 20 0C FD C9 CE
08B8- F0 1E 18 A0 01 B1 3A 65
08C0- CE 91 3A C8 B1 3A 65 CF
08C8- 91 3A 20 80 FE 20 D0 F8
08D0- 20 84 FE A9 05 20 C9 FC
08D8- 20 53 F9 85 3A 84 3B C4
08E0- FA 90 A3 A5 F9 C5 3A B0
08E8- 9D 2C 10 C0 60 A9 8D 20
08F0- ED FD C8 B9 05 09 D0 F7
08F8- 60 20 6F FD A0 00 20 A7
0900- FF A5 3E A4 3F 60 C9 CE
0908- C9 DA C9 CF A0 D0 D2 CF
0910- C7 D2 C1 CD CD C1 BA A0
0918- 00 C4 C5 D3 D4 C9 CE C1
0920- DA C9 CF CE C5 A0 A0 A0
0928- A0 BA A0 00 C9 CE C9 DA
0930- C9 CF A0 C1 D2 C5 C1 A0
0938- D2 C9 CC AE BA A0 00 C6
0940- C9 CE C5 A0 A0 A0 C1 D2
0948- C5 C1 A0 D2 C9 CC AE BA
0950- A0 00 C6 C9 CE C5 A0 D0
0958- D2 CF C7 D2 C1 CD CD C1
0960- A0 A0 BA A0 00 C1 AD C1
0968- D5 D4 CF A0 CD AD CD C1
0970- CE D5 C1 CC A0 BA A0 00
```

Figura 2: Modulo oggetto del programma rilocatore. Va copiato in memoria e salvato con BSAVE RILOCATORE,A$800,L$178.

Un secondo problema è rappresentato dalle tabelle di caratteri ASCII (tipicamente le scritte) che il disassembler Apple tenta di interpretare come istruzioni (e spesso ci riesce), talvolta questa interpretazione produce un codice a tre byte (spesso però l'indirizzo non è tra quelli da rilocare) apparentemente da modificare.

Questi ultimi due casi non possono essere eseguiti da un programma automatico e andranno perciò controllati manualmente; se il secondo non crea problemi (al massimo qualche testo incomprensibile, ma facilmente individuabile) il primo porta al blocco del programma (e a volte alla distruzione dello stesso); per cui, dopo aver effettuato la rilocazione, si deve controllare attentamente il programma alla ricerca di simili occorrenze.

Il rilocatore

Il programma rilocatore funziona sfruttando le capacità del disassembler interno dell'Apple. Una alla volta disassembla le istruzioni, poi guarda se sono a tre byte e se l'indirizzo è di quelli da modificare; se sì, lo modifica e riscrive l'istruzione corretta, altrimenti prosegue la ricerca. Nel caso si sappia in anticipo che nel programma si possono trovare delle scritte è possibile fare in modo che il rilocatore si arresti prima di ciascuna istruzione da modificare chiedendo il permesso all'operatore. Inoltre, per permettere la rilocazione anche di singole parti di programma, l'area di indirizzi da modificare può essere definita a piacere (si può ad esempio spostare solo una subroutine e modificare nel programma originale tutte le chiamate a quella specifica subroutine).

Per caricare la routine rilocatrice si può utilizzare un assemblatore (copiando il listato sorgente di figura 3) oppure immetterla direttamente dal monitor con il solito CALL-151 e poi l'indirizzo iniziale, i due punti e i dati della figura 2 separati dallo spazio. Terminato l'inserimento si salva il tutto battendo:

BSAVE RILOCATORE, A$800,L$178.

Naturalmente chi vuole, ora, lo può rilocare altrove!

Nota: Alla riga 160/161 compare una istruzione di attesa (DELAY), questa è relativa alla gestione del registratore a cassetta (ritarda ACC*.1664 secondi) e non esiste nel IIc. Si può sostituire con un LDA #$87 e JSR COUT, naturalmente abbassando il volume!

1

```
0800             1  ;
0800             2  ; PROGRAMMA
0800             3  ;
0800             4  ; RILOCATORE
0800             5  ;
0800             6  MODE    EPZ $31
0800             7  NUML    EPZ $3E
0800             8  NUMH    EPZ $3F
0800             9  PCL     EPZ $3A
0800            10  PCH     EPZ $3B
0800            11  LEN     EPZ $2F
0800            12  AUTO    EPZ $FF
0800            13  STARTL  EPZ $EB
0800            14  STARTH  EPZ $EC
0800            15  DSTL    EPZ $ED
0800            16  DSTH    EPZ $EE
0800            17  DIFFL   EPZ $CE
0800            18  DIFFH   EPZ $CF
0800            19  ENDL    EPZ $F9
0800            20  ENDH    EPZ $FA
0800            21  INRL    EPZ $FB
0800            22  INRH    EPZ $FC
0800            23  FINL    EPZ $FD
0800            24  FINH    EPZ $FE
0800            25  ;
0800            26  WORK    EPZ $20
0800            27  ;
0800            28  MSG1    EPZ 0
0800            29  MSG2    EPZ 19
0800            30  MSG3    EPZ 38
0800            31  MSG4    EPZ 57
0800            32  MSG5    EPZ 76
0800            33  MSG6    EPZ 95
0800            34  ;
0800            35  COUT    EQU $FDED
0800            36  GETLN   EQU $FD6F
0800            37  GETNUM  EQU $FFA7
0800            38  ISTDSP  EQU $F8D0
0800            39  PCADJ   EQU $F953
0800            40  SETINV  EQU $FE80
0800            41  SETNRM  EQU $FE84
0800            42  KEYIN   EQU $FD0C
0800            43  BELL    EQU $FBDD
0800            44  DELAY   EQU $FCC9
0800            45  ;
0800            46  KSTR    EQU $C010
0800            47  KEY     EQU $C000
0800            48  ;
0800            49  ;
0800 A9FF       50  INIT    LDA #$FF      prepara il Monitor
0802 8531       51          STA MODE      all'uso di GETNUM.
0804 A000       52  START   LDY #MSG1     stampa il primo
0806 20ED08     53          JSR DSP       messaggio, e
0809 20F908     54          JSR INPUT     riceve la risposta
080C 85EB       55          STA STARTL    a sedici bit in
080E 84EC       56          STY STARTH    Start.
0810            57  ;
0810 A013       58          LDY #MSG2     Stampa il secondo
0812 20ED08     59          JSR DSP       messaggio, e
0815 20F908     60          JSR INPUT     riceve la relativa
```

3

```
0884 853A      119          STA PCL       punta la Work.area.
0886 AD00C0    120  LOOP    LDA KEY       ** loop principale.
0889 C983      121          CMP #$83      CTRL C = interrompe
088B F05C      122          BEQ RTS1      il lavoro.
088D 20D0F8    123          JSR ISTDSP    disassembla da PC,
0890 A002      124          LDY #$2       se è lunga
0892 C42F      125          CPY LEN       3 byte
0894 D042      126          BNE CONT1     allora va
0896 38        127          SEC           rilocata,
0897 88        128          DEY           controlla se il
0898 B13A      129          LDA (PCL),Y   campo indirizzo
089A E5FB      130          SBC INRL      dell'istruzione
089C C8        131          INY           è compreso
089D B13A      132          LDA (PCL),Y   nell'area rilocabile.
089F E5FC      133          SBC INRH
08A1 9035      134          BCC CONT1
08A3 A5FD      135          LDA FINL
08A5 88        136          DEY
08A6 F13A      137          SBC (PCL),Y
08A8 A5FE      138          LDA FINH
08AA C8        139          INY
08AB F13A      140          SBC (PCL),Y   altrimenti non va
08AD 9029      141          BCC CONT1     rilocata, se sì
08AF 24FF      142          BIT AUTO      devo aspettare la
08B1 1007      143          BPL GOON      conferma?
08B3 200CFD    144          JSR KEYIN     Si, leggi la tastiera:
08B6 C9CE      145          CMP #"N"      se è "N"
08B8 F01E      146          BEQ CONT1     non rilocare!
08BA 18        147  GOON    CLC           ** riloca **
08BB A001      148          LDY #$1       somma
08BD B13A      149          LDA (PCL),Y   all'indirizzo
08BF 65CE      150          ADC DIFFL     dell'istruzione
08C1 913A      151          STA (PCL),Y   la distanza
08C3 C8        152          INY           di rilocazione
08C4 B13A      153          LDA (PCL),Y   e rimette
08C6 65CF      154          ADC DIFFH     a posto il campo
08C8 913A      155          STA (PCL),Y   indirizzo.
08CA           156  ;
08CA 2080FE    157          JSR SETINV    Ristampa in inverso
08CD 20D0F8    158          JSR ISTDSP    la nuova istruzione,
08D0 2084FE    159          JSR SETNRM    Normal.
08D3 A905      160  WAIT    LDA #$5       aspetta un secondo
08D5 20C9FC    161          JSR DELAY     per leggerla,
08D8           162  ;
08D8 2053F9    163  CONT1   JSR PCADJ     poi incrementa
08DB 853A      164          STA PCL       il program counter
08DD 843B      165          STY PCH       e
08DF C4FA      166          CPY ENDH      vede se il
08E1 90A3      167          BCC LOOP      programma
08E3 A5F9      168          LDA ENDL      è finito,
08E5 C53A      169          CMP PCL       altrimenti
08E7 B09D      170          BCS LOOP      continua.
08E9           171  ;
08E9 2C10C0    172  RTS1    BIT KSTR      pulisce lo strobe
08EC 60        173          RTS           ed esce.
08ED           174  ;
08ED           175  ; SUBROUTINES
08ED           176  ;
08ED A98D      177  DSP     LDA #$8D      Stampa un CR e
```

2

```
0818 85ED      61          STA DSTL       risposta in
081A 84EE      62          STY DSTH       Dest.
081C           63  ;
081C 38        64          SEC
081D A5ED      65          LDA DSTL       Calcola
081F E5EB      66          SBC STARTL     la distanza
0821 85CE      67          STA DIFFL      tra la
0823 A5EE      68          LDA DSTH       locazione
0825 E5EC      69          SBC STARTH     iniziale e
0827 85CF      70          STA DIFFH      quella
0829 9010      71          BCC NOCOMP     finale; se
082B A5CE      72          LDA DIFFL      è positiva, OK.
082D 49FF      73          EOR #$FF       altrimenti
082F 6900      74          ADC #$0        effettua il
0831 85CE      75          STA DIFFL      complemento
0833 A5CF      76          LDA DIFFH      a due della
0835 49FF      77          EOR #$FF       distanza, che,
0837 6900      78          ADC #$0        così, andrà
0839 85CF      79          STA DIFFH      sempre sommata.
083B           80  ;
083B A026      81  NOCOMP LDY #MSG3      Terzo
083D 20ED08    82          JSR DSP        messaggio e
0840 20F908    83          JSR INPUT      relativa risposta
0843 85FB      84          STA INRL       di inizio area
0845 84FC      85          STY INRH       rilocabile.
0847           86  ;
0847 A039      87          LDY #MSG4      Adesso
0849 20ED08    88          JSR DSP        chiede la
084C 20F908    89          JSR INPUT      fine dell'area
084F 85FD      90          STA FINL       rilocabile.
0851 84FE      91          STY FINH
0853           92  ;
0853 A04C      93          LDY #MSG5      Chiede
0855 20ED08    94          JSR DSP        l'indirizzo
0858 20F908    95          JSR INPUT      finale del
085B 38        96          SEC            programma e
085C E5EB      97          SBC STARTL     lo usa per
085E 85F9      98          STA ENDL       calcolarne la
0860 98        99          TYA            lunghezza.
0861 E5EC     100          SBC STARTH     che,
0863 85FA     101          STA ENDH       sommata
0865 18       102          CLC            all'indirizzo
0866 A5F9     103          LDA ENDL       iniziale della
0868 6900     104          ADC #$0        Work.area, darà
086A 85F9     105          STA ENDL       l'indirizzo
086C A5FA     106          LDA ENDH       finale del
086E 6920     107          ADC #WORK      programma
0870 85FA     108          STA ENDH       rilocato.
0872          109  ;
0872 A05F     110          LDY #MSG6      Ultimo messaggio;
0874 20ED08   111          JSR DSP        automatico o
0877 200CFD   112          JSR KEYIN      manuale, il Carry
087A C9CD     113          CMP #"M"       finisce nel bit 7
087C 66FF     114          ROR AUTO       di Auto.
087E          115  ;
087E A920     116  GO      LDA #WORK      Inizio operazione
0880 853B     117          STA PCH        di rilocazione, il
0882 A900     118          LDA #$0        program counter
```

4

```
08EF 20EDFD   178  NEXT    JSR COUT       poi il testo
08F2 C8       179          INY            a partire dal
08F3 B90509   180          LDA TESTI,Y    punto Y e fino a
08F6 D0F7     181          BNE NEXT       che non trova
08F8 60       182          RTS            uno zero.
08F9          183  ;
08F9          184  ;
08F9 206FFD   185  INPUT   JSR GETLN      Accetta una linea
08FC A000     186          LDY #$0        da tastiera e la
08FE 20A7FF   187          JSR GETNUM     mette nel buffer,
0901 A53E     188          LDA NUML       poi la trasforma in
0903 A43F     189          LDY NUMH       un numero esadecimale
0905 60       190  TESTI   RTS            che mette in A ed Y.
0906          191  ;
0906 C9CEC9   192          ASC "INIZIO PROGRAMMA: "
0909 DAC9CF
090C A0D0D2
090F CFC7D2
0912 C1CDCD
0915 C1BAA0
0918 00       193          HEX 00
0919 C4C5D3   194          ASC "DESTINAZIONE    : "
091C D4C9CE
091F C1DAC9
0922 CFCEC5
0925 A0A0A0
0928 A0BAA0
092B 00       195          HEX 00
092C C9CEC9   196          ASC "INIZIO AREA RIL.: "
092F DAC9CF
0932 A0C1D2
0935 C5C1A0
0938 D2C9CC
093B AEBAA0
093E 00       197          HEX 00
093F C6C9CE   198          ASC "FINE   AREA RIL.: "
0942 C5A0A0
0945 A0C1D2
0948 C5C1A0
094B D2C9CC
094E AEBAA0
0951 00       199          HEX 00
0952 C6C9CE   200          ASC "FINE PROGRAMMA  : "
0955 C5A0D0
0958 D2CFC7
095B D2C1CD
095E CDC1A0
0961 A0BAA0
0964 00       201          HEX 00
0965 C1ADC1   202          ASC "A-AUTO M-MANUAL : "
0968 D5D4CF
096B A0CDAD
096E CDC1CE
0971 D5C1CC
0974 A0BAA0
0977 00       203          HEX 00
```

Figura 3: Sorgente LISA del programma rilocatore.