La parola ai lettori

Dopo tante puntate dedicate all'analisi dell'interprete Basic della Microsoft e proprio nel bel mezzo della serie di puntate riguardanti il calcolo di espressioni, siamo ben lieti di lasciare lo spazio ad un nostro lettore, il quale ci ha inviato una monumentale lettera piena di suggerimenti, programmi e notizie, che presenteremo in più riprese nelle rubriche apposite, sia già affermate che nasciture — è una promessa...

Ecco che dunque lasciamo la parola al nostro lettore Raffaele Gonnella di Roma, che ci presenta un proprio studio riguardante la modifica dell'MBASIC con l'aggiunta di nuove istruzioni, prendendo lo spunto da quanto scritto dal redattore della presente rubrica, però nell'ambito della rubrica «I trucchi del CP/M», all'epoca in cui la presente non era ancora nata.

Ulteriori note sulla modifica dell'MBASIC

Per gli «smanettoni» incalliti, tra i quali credo di potermi inserire a pieno titolo, la serie di articoli di Pierluigi Panunzi sono una vera ghiottoneria. E infatti, quando nell'ormai lontano ottobre 1984 iniziò lo «smanettamento» dell'MBASIC, il mio interesse fu subito altissimo. Ancora di più nel seguito, con l'aggiunta dei nuovi comandi e la spiegazione del funzionamento della jump table.

Però (in informatica c'è sempre un «però» dopo gli elogi) le cose che in teoria erano così belle, nella pratica invece non lo sono state altrettanto. Già quando avevo visto che l'MBASIC di riferimento era quello dell'Osborne incominciai a temere, data la peculiarità di questo sistema. E i miei dubbi erano stati confermati da quanto l'autore precisava sul numero 38.

Ad ogni modo il risultato che si poteva ottenere era troppo allettante per non cimentarsi nell'impresa. Così, armato di tanta pazienza, anche se dopo molto tempo, ho preso il mio interprete Basic per vedere la situazione e, come era scontato, era tutto differente. Comunque sono riuscito a risalire alle differenze tra i due interpreti e poiché credo di non essere l'unico ad avere questa versione di MBASIC spero di aiutare altri lettoti nella stessa situazione.

E veniamo dunque al discorso più tecnico. Prenderò le mosse proprio da quanto scritto sui vari numeri della rivista di cui riporto una mini bibliografia:

Introduzione alla modifica dell'MBASIC - MC 34 pag. 159
Nuovo comando: DSK - MC 35 pag. 160
Nuovi comandi: RDA e STO - MC 36 pag. 192
Ancora sul comando DSK - MC 37 pag. 164
Jump Table MBASIC - MC 38 pag. 163

Per prima cosa va subito detto che la mia versione di MBASIC porta il numero 5.2 rilasciata il 14 luglio 1980. E questo è il primo dato identificativo per orizzontarsi. E adesso, rivista 34 alla mano, andiamo a vedere tutte le differenze.

La tabella di fondo pagina riassume tutte le differenze principali riscontrate fra le due versioni di MBASIC. Passiamo ora a vedere invece le 6 modifiche che venivano proposte a fronte di queste informazioni (cfr. MC 34 pag. 162):

1) come prima cosa viene spostato lo stack pointer in una zona sicura:

MC	Versione 5.2
5D72 21→ B2	5D7F 30→ B2
5D73 61→ 63	5D80 61→ 63

va qui precisato che malgrado i valori di inizializzazione dello SP siano diversi (MC: 6121H; V.5.2: 6130H) il riportarli uguali non ha creato nessun tipo di malfunzionamento (lo SP è stato lasciato uguale per pigrizia).

2) e 3) Queste due modifiche non sono state fatte in quanto, contrariamente a quanto detto sulla rivista, la mia versione 5.2 effettua tranquillamente il recovery.

4) Ecco le locazioni per l'aggancio

locazioni MC	locazioni 5.2	funzione svolta
5D71H	5D75H	jump iniziale
5DEEH	5DF0H	lunghezza record
5E00H	5E02H	max RAM a disposizione
5E06H	5E08H	max numero file
5E1CH	5E1EH	rout. controllo '/'
5D6FH-5EC5H	5D7CH-5ECBH	rout. per spazio file
5EECH (079AH-0AB1H)	5EF2H (07BD-0AD4)	gestione spazio file
61AEH	61BCH	START OF BASIC

della routine al programma esistente:

MC	Versione 5.2
5E70 C9→ DB	5E72 EC→ E4
5E71 0C→ 5F	5E73 0C→ 5F

5) Ecco la modifica per il nuovo inizio del Basic:

MC	Versione 5.2
5EC6 6F→ 00	5ECC 7C→ 00
5EC7 5D→ 61	5ECD 5D→ 61

6) Ecco infine il nuovo comando aggiunto:

```
5FE4    FE  50      CP 'P'
5FE6    C2  EC  0C  JP NZ,0CECH
5FE9    CD  32  13  CALL 1332H
5FEC    CD  07  44  CALL 4407H
5FEF    3A          DB ':'
5FF0    CD  7F  1C  CALL 1C7FH
5FF3    E5          PUSH HL
5FF4    2A  CC  5E  LD HL,(5ECCH)
5FF7    19          ADD HL,DE
5FF8    22  CC  5E  LD (5ECCH),HL
5FFB    E1          POP HL
5FFC    F1          POP AF
5FFD    C3  98  5E  JP 5E98H
```

Bisogna notare che questa routine va ad occupare gli ultimi byte del prompt iniziale per cui occorre modificare anche le seguenti locazioni:

5FE1→ 0D
5FE2→ 0A
5FE3→ 00

In questo modo le ultime lettere del prompt verranno tagliate. È anche possibile modificare il prompt nelle ultime locazioni. Va tenuto presente che il prompt stesso parte dalla locazione 5F85H.

Vediamo ora tutto il resto (cioè quanto detto nelle riviste 35-38). Per far questo occorre partire dal fondo, cioè dall'ottima spiegazione sul funzionamento della jump table del Basic. Prendendo le mosse da questa infatti non solo ho effettuato le modifiche proposte dall'autore ma, seguento la strada tracciata, ne ho apportate delle altre. In particolare ho aggiunto una nuova istruzione (ROW) e ho modificato il comando SAVE (come in seguito vedremo). Inoltre ho raccolto tutte le modifiche in un unico programma in Assembler a partire dalla locazione 6000H, in modo da ottenere un tutt'uno più organico e più facilmente modificabile e gestibile. Infine ho corretto qualche inesattezza rendendo le modifiche più universali. Prima di vedere il listato di questo programma vediamo come ho impostato il lavoro e le aggiunte fatte.

Parleremo prima di quest'ultime. Infatti una volta scoperta la strada la fantasia si scatena e allora ho fatto due modifiche:

1) ho aggiunto una nuova istruzione ROW(0) la cui sintassi è identica alla POS(0) e che ritorna la riga su cui è posizionato il cursore. Non so se tutti i lettori potranno implementare questa istruzione (copiata dalla CRSLIN del BASICA IBM) in quanto richiede una approfondita conoscenza del proprio sistema. In ogni caso, se non fosse possibile sarà sufficiente ignorare quanto detto a suo riguardo.

Questa nuova istruzione mi ha costretto ad ulteriormente modificare l'ormai irriconoscibile istruzione RANDOMIZE che tagliando qui e lì si è ridotta ad una striminzita RAN.

2) Ho eliminato una delle noie più grosse del CP/M. Infatti alzi la mano chi non si è mai beccato il «simpaticissimo» messaggio **«Bdos err on x: R/O»** durante il salvataggio di un programma importantissimo perché non era stato dato il reset software. Credo nessuno potrà farlo. In verità mi sono chiesto perché i cervelloni della Microsoft non l'avessero fatta già loro questa modifica e mi è sorto il dubbio che ci fosse qualche problema interno alla logica dell'interprete: invece no. Con la modifica implementata, quando si impartisce il comando SAVE avviene un reset automatico e poi il Basic continua a lavorare normalmente.

Ed ecco una per una le modifiche da apportare all'MBASIC.

Per aggiungere l'istruzione DSK e modificare la DELETE in DEL:

1DD→ 00
1DE→ 60
2AA→ CC
2AB→ AA
2AC→ 53
2AD→ CB
2AE→ 20

(cfr. MC 35 pag. 160)

Per aggiungere le istruzioni RDA e ROW e modificare la RANDOMIZE in RAN:

1DF→ 03
1E0→ 60 (per la RDA; cfr. MC 36 pag. 192)
1E1→ OC
1E2→ 60 (per la ROW)
3E3→ 41
3E4→ CE
3E5→ BB
3E6→ 4F
3E7→ D7
3E8→ 22
3E9→ 44
3EA→ C1
3EB→ 21 (per la RANDOMIZE)

Come appare chiaro l'istruzione ROW prende il token 22 (cfr. MC 38 pag. 164).

Per chi non può implementare l'istruzione ROW:

— modificare le locazioni 1DF e 1E0 come indicato;

— NON modificare le locazioni 1E1 e 1E2;

— per le locazioni da 3E3 in poi procedere come di seguito:

3E3-3E6 NON MODIFICARE
3E7→ CD
3E8→ BB
3E9→ 44
3EA→ C1
3EB→ 21

Effettuando questa modifica l'istruzione RANDOMIZE diventa RANDOM.

Per aggiungere l'istruzione STO e modificare l'istruzione SYSTEM in SYS:

13D→ 06
13E→ 60
41C→ D3
41D→ BD
41E→ 54
41F→ CF
420→ 9C

(cfr. MC 36 pag. 193)

Per modificare l'istruzione SAVE:

19B→ 09
19C→ 60

Come si vede tutte le locazioni inferiori a 200H sono uguali a quelle della versione 5.21 mentre ci sono piccole differenze per le altre. Unica eccezione l'istruzione SYSTEM che ha la stessa posizione in entrambe le versioni.

I valori immessi nelle locazioni inferiori a 200H saranno chiari durante la spiegazione del listato.

Ultima tabella prima di spiegare il listato è quella che riepiloga le corrispondenze tra tutte le varie routine utilizzate nelle nuove istruzioni:

MC	Versione 5.2
1305	1332
1C55	1C7F
205C	2081
2069	208E
22C2	22CB
22CC	22E0
29CD	29DB
43C7	4407
5D41	5D4E

E, infine vediamo questo listato. Il programma usa una sua jump table che inizia a 6000H e per questo le locazioni inferiori a 200H dell'interprete venivano modificate con quei valori. Inoltre non credo sia necessario dilungarmi sulla flessibilità di una jump table, che consente di effettuare le modifiche alle varie routine senza essere costretti a modificare gli indirizzamenti assoluti all'interno dell'interprete.

Il programma così come è scritto è pronto per essere assemblato con il MACRO80 della Microsoft ma non credo che ci saranno grosse modifiche per qualsiasi altro assemblatore. Per chi non dispone di assemblatore sarà sufficiente copiare il codice oggetto riportato a lato del listato.

Ed ora andiamo con i commenti:

istr. 1-2

indicano che il programma verrà assemblato a partire dalla locazione 100H

istr. 3

viene inizializzata la variabile BDOS uguale a 5 per le call al CP/M

istr. 4
viene inizializzata una variabile utilizzata nella istruzione ROW che più giù vedremo
istr. 5
viene impostato lo «sfasamento» in modo da ottenere gli indirizzamenti desiderati (per noi a partire da 6000H)
istr. 6
si avverte l'assemblatore che il programma è scritto in Z80
istr. 7-13
jump table delle nuove istruzioni
istr. 14-87
programma dell'istruzione DSK (cfr. MC 37 pag. 164). Il programma è quasi identico a quello di Pierluigi Panunzi tranne per qualche particolare. Più precisamente:
— usando l'istruzione DSK come era nella prima versione il drive di default veniva modificato con quello scelto nell'istruzione;
— non veniva effettuato il reset software per cui cambiando il disco per il meccanismo interno del CP/M il valore tornato non era esatto
— non veniva considerato il «bloccaggio» del disco, ovvero se la minima allocazione di spazio era di 1, 2, 4, 8 oppure 16 K.

A queste inesattezze si è fatto fronte nella routine. Alle istruzioni 16-25 non si perde il puntamento sul drive di default. Alle istruzioni 21-22 si effettua il reset software mentre alle istruzioni 34-35 e 66-87 si considera l'eventuale «bloccaggio» (in inglese bloksize) del disco esaminato per far tornare un esatto valore.
istr. 88-95
routine dell'istruzione RDA in tutto identica a quella presentata su MC 36 pag. 192
istr. 96-110
routine dell'istruzione STO. Anche questa routine è identica a quella presentata su MC (n. 36 pag. 193) tranne per un particolare: l'istr. 97 dove viene incrementato il registro DE. Questo incremento sulla routine originale non c'era, ma senza il primo valore dell'istruzione STO veniva inspiegabilmente saltato. Non so se questa sia stata una dimenticanza oppure un diverso funzionamento delle routine nei due interpreti, fatto sta che questa modifica è necessaria per un corretto funzionamento dell'istruzione nella versione 5.2
istr. 111-125
routine della modifica del comando SAVE. Il suo funzionamento è abba-

```
 1    0000'                          ASEG
 2                                   ORG     100H
 3    0005                           BDOS    EQU     5
 4    0048                           PVIDEO  EQU     48H
 5                                   .PHASE  6000H
 6                                   .Z80
 7                              ;****** NUOVE ISTRUZIONI MBASIC
 8    6000   C3 600F                 JP      DSK
 9    6003   C3 607D                 JP      RDA
10    6006   C3 608A                 JP      STO
11    6009   C3 60A1                 JP      SAVE
12    600C   C3 [illegible]           JP      ROW
13                              ;
14    600F   CD 208E           DSK:  CALL    208EH
15    6012   E6 0F                   AND     0FH
16                              ;
17    6014   32 6025                 LD      (DRIVE),A
18    6017   0E 19                   LD      C,19H
19    6019   CD 0005                 CALL    BDOS
20    601C   32 6073                 LD      (OLDRIV),A
21    601F   0E 0D                   LD      C,0DH
22    6021   CD 0005                 CALL    BDOS
23                              ;
24    6024   1E                      DB      1EH
25    6025   00               DRIVE: DB      0
26    6026   0E 0E                   LD      C,0EH
27    6028   CD 0005                 CALL    BDOS
28                              ;
29    602B   0E 1F                   LD      C,1FH
30    602D   CD 0005                 CALL    BDOS
31    6030   11 0002                 LD      DE,2
32    6033   19                      ADD     HL,DE
33    6034   7E                      LD      A,(HL)
34    6035   D6 02                   SUB     2
35    6037   F5                      PUSH    AF
36    6038   13                      INC     DE
37    6039   19                      ADD     HL,DE
38    603A   5E                      LD      E,(HL)
39    603B   23                      INC     HL
40    603C   56                      LD      D,(HL)
41                              ;
42    603D   13                      INC     DE
43                              ;
44    603E   0E 1B                   LD      C,1BH
45    6040   CD 0005                 CALL    BDOS
46    6043   DD 21 0000              LD      IX,0
47    6047   06 08            ALFA: LD      B,8
48    6049   7E                      LD      A,(HL)
49    604A   F5                      PUSH    AF
50    604B   F1               BETA:  POP     AF
51    604C   07                      RLCA
52                              ;
53    604D   38 02                   JR      C,GAMMA
54                              ;
55    604F   DD 23                   INC     IX
56    6051   F5               GAMMA: PUSH    AF
57    6052   1B                      DEC     DE
58    6053   7A                      LD      A,D
59    6054   B3                      OR      E
60    6055   28 06                   JR      Z,DELTA
61    6057   10 F2                   DJNZ    BETA
62    6059   23                      INC     HL
63    605A   F1                      POP     AF
64    605B   18 EA                   JR      ALFA
65    605D   F1               DELTA: POP     AF
66    605E   F1                      POP     AF
67    605F   DD E5                   PUSH    IX
68    6061   E1                      POP     HL
69    6062   47                      LD      B,A
70    6063   DD 21 6072              LD      IX,RADDOPPIA
71    6067   DD 2B            DECR:  DEC     IX
72    6069   10 FC                   DJNZ    DECR
73    606B   DD E9                   JP      (IX)
```

```
 74    606D   29                ADD     HL,HL
 75    606E   29                ADD     HL,HL
 76    606F   29                ADD     HL,HL
 77    6070   29                ADD     HL,HL
 78                             ;
 79    6071   E5                PUSH    HL
 80    6072                     RADDOPPIA:
 81    6072   1E                DB      1EH
 82    6073   00                OLDRIV: DB      0
 83    6074   0E 0E             LD      C,0EH
 84    6076   CD 0005           CALL    BDOS
 85    6079   E1                POP     HL
 86    607A   C3 29DB           JP      29DBH
 87                             ;
 88    607D   CD 22E0           RDA:    CALL    22E0H
 89    6080   CD 5D4E           CALL    5D4EH
 90    6083   7E                LD      A,(HL)
 91    6084   23                INC     HL
 92    6085   66                LD      H,(HL)
 93    6086   6F                LD      L,A
 94    6087   C3 29DB           JP      29DBH
 95                             ;
 96    608A   CD 22CB           STO:    CALL    22CBH
 97    608D   13                INC     DE
 98    608E   D5                PUSH    DE
 99    608F   CD 5D4E           CALL    5D4EH
100    6092   CD 2081           L1:     CALL    2081H
101    6095   D1                L2:     POP     DE
102    6096   12                LD      (DE),A
103    6097   13                INC     DE
104    6098   D5                PUSH    DE
105    6099   2B                DEC     HL
106    609A   CD 1332           CALL    1332H
107    609D   20 F3             JR      NZ,L1
108    609F   D1                POP     DE
109    60A0   C9                RET
110                            ;
111    60A1   08                SAVE:   EX      AF,AF'
112    60A2   D9                EXX

113    60A3   0E 19             LD      C,19H
114    60A5   CD 0005           CALL    BDOS
115    60A8   F5                PUSH    AF
116    60A9   0E 0D             LD      C,0DH
117    60AB   CD 0005           CALL    BDOS
118    60AE   F1                POP     AF
119    60AF   5F                LD      E,A
120    60B0   0E 0E             LD      C,0EH
121    60B2   CD 0005           CALL    BDOS
122    60B5   D9                EXX
123    60B6   08                EX      AF,AF'
124    60B7   C3 539C           JP      539CH
125                            ;
126    60BA   AF                ROW:    XOR     A
127    60BB   2A 0048           LD      HL,(PVIDEO)
128    60BE   11 0050           LD      DE,80
129    60C1   CB 7D             R1:     BIT     7,L
130    60C3   28 05             JR      Z,R2
131    60C5   ED 52             R3:     SBC     HL,DE
132    60C7   3C                INC     A
133    60C8   18 F7             JR      R1
134    60CA   47                R2:     LD      B,A
135    60CB   AF                XOR     A
136    60CC   BC                CP      H
137    60CD   78                LD      A,B
138    60CE   20 F5             JR      NZ,R3
139    60D0   3C                INC     A
140    60D1   26 00             LD      H,0
141    60D3   6F                LD      L,A
142    60D4   C3 29DB           JP      29DBH
143                            ;
144                                    .DEPHASE
145                                    END
```

stanza semplice. La prima cosa da fare, per sicurezza, è quella di salvare tutti i registri (istr. 111-112). Alle istr. 113-115 si prende il drive di default che viene memorizzato nello stack. Le istr. 116-117 effettuano il reset. Le istr. 118-121 riprendono il drive di default dallo stack e lo selezionano di nuovo. Infine (istr. 122-123) si ripescano i valori originari dei registri e quindi (istr. 124) si torna alla vecchia rout. di SAVE. Il valore dell'istr. 124 era quello che si trovava alle locazioni 19B e 19C dell'interprete. La logica di reset software è stata copiata dall'istruzione Basic RESET

istr. 126-143

viene qui implementata l'istruzione ROW(0) che funziona come la POS(0), ritornando il valore della riga su cui è posizionato il cursore. Per implementare questa funzione occorre però conoscere dove il proprio sistema va a memorizzare il puntamento video. Inoltre (questo per i sistemi con il video gestito «memory mapped») occorre sapere se questo valore è in valore assoluto (da 1 fino a 1920 o 2000) oppure se in locazione di memoria video. Per me è stato facile in quanto ho implementato da me il software di base e quindi il puntatore video che mi interessava è memorizzato alla locazione 48H (ecco perché veniva inizializzata la costante PVIDEO all'istr. 4). Una volta a conoscenza di questa informazione non resta che manipolarla. La routine presentata memorizza il valore del puntatore video in HL e poi sottrae 80 (la lunghezza di una riga) fino a che non si arriva a zero. Il valore ottenuto viene poi tornato al Basic tramite il JP 29DBH. Per puntamenti «memory mapped» si può utilizzare la stessa routine preceduta da una sottrazione del valore di HL con l'indirizzo iniziale della memoria video.

Chi non volesse avventurarsi in questa modifica, dopo aver effettuato quanto detto in precedenza per non implementare questa istruzione, deve ancora fare le seguenti cose:

— modificare l'istr. 12 nel seguente modo

12 600C C3 00 00 JP 0000

— copiare il resto del programma fino all'istr. 125, tralasciando quindi la routine della ROW

istr. 144-145

chiudono il programma con la fine dello «sfasamento» e l'END.

MC