Dump interpretato

Quando si scrivono o si analizzano dei programmi in linguaggio macchina farebbe spesso comodo poter riconoscere le scritte dal resto dei dati o del programma. Un modo abbastanza macchinoso di fare ciò è di spostare con la MOVE del monitor il pezzo di memoria che si sta esaminando nella zona della pagina video, da $400 a $7FF. Con questo sistema barbaro però non si riesce a capire esattamente né la successione delle scritte (perché la pagina video è mescolata) né il punto di inizio di una determinata parola dato che non esiste un riferimento evidente alle locazioni di memoria.

Ecco perciò il programma di Dump interpretato dove per interpretato si intende che quando il contenuto di una certa locazione è interpretabile come un carattere ASCII questo viene stampato a destra del DUMP. Il Dump, per il resto è identico a quello del Monitor (che si ottiene battendo l'indirizzo iniziale, un punto e l'indirizzo finale). L'unica differenza consiste nel fatto che il Dump parte sempre da una pagina (di memoria) intera.

Premendo un tasto qualsiasi viene effettuato il dump delle otto locazioni successive, premendo la freccia a sinistra vengono interpretate le otto locazioni precedenti e premendo l'Escape si esce dal programma e si torna al Basic o al Monitor.

Un esempio di Dump si trova in *figura 3*. Come vedete è facilissimo riconoscere la tabella dei comandi dell'Applesoft e la locazione iniziale di ciascuna istruzione.

```
**END OF PASS 1
**END OF PASS 2

0300                1           ORG $300
0300                2           OBJ $300
0300                3           ;
0300                4  COUT     EQU $FDED
0300                5  PRBYT    EQU $FDDA
0300                6  PRTAX    EQU $F941
0300                7  CROUT    EQU $FD8E
0300                8  KEYIN    EQU $FD1B
0300                9           ;
0300               10  LOC      EPZ $FE
0300               11           ;
0300 A000          12  START    LDY #0
0302 84FE          13           STY LOC
0304 A5FF          14  LOOP     LDA LOC+1
0306 A6FE          15           LDX LOC
0308 2041F9        16           JSR PRTAX
030B A9AD          17           LDA #"-"
030D 20EDFD        18           JSR COUT
0310 A9A0          19  PBYT     LDA #" "
0312 20EDFD        20           JSR COUT
0315 B1FE          21           LDA (LOC),Y
0317 20DAFD        22           JSR PRBYT
031A C8            23           INY
031B C008          24           CPY #$8
031D D0F1          25           BNE PBYT
031F A9A0          26           LDA #" "
0321 20EDFD        27           JSR COUT
0324 20EDFD        28           JSR COUT
0327               29           ;
0327 A000          30           LDY #0
0329 A200          31           LDX #0
032B B1FE          32  ASCI     LDA (LOC),Y
032D C920          33           CMP #32
032F 9008          34           BLT NOPE
0331 C9A0          35           CMP #$A0
0333 B006          36           BGE OK
0335 C980          37           CMP #$80
0337 9002          38           BLT OK
0339 A9A0          39  NOPE     LDA #" "
033B 0980          40  OK       ORA #%10000000
033D 20EDFD        41           JSR COUT
0340 E6FE          42           INC LOC
0342 E8            43  YRTS     INX
0343 E008          44           CPX #8
0345 D0E4          45           BNE ASCI
0347 20BEFD        46           JSR CROUT
034A A5FE          47           LDA LOC
034C D002          48           BNE KEY
034E E6FF          49           INC LOC+1
0350 201BFD        50  KEY      JSR KEYIN
0353 C988          51           CMP #$88
0355 F005          52           BEQ PREV
0357 C99B          53           CMP #$9B
0359 D0A9          54           BNE LOOP
035B 60            55           RTS
035C A5FE          56  PREV     LDA LOC
035E 38            57           SEC
035F E910          58           SBC #$10
0361 B002          59           BCS PREV1
0363 C6FF          60           DEC LOC+1
0365 85FE          61  PREV1    STA LOC
0367 4C0403        62           JMP LOOP
                   63           END
```

Figura 1 - Listato originale del programma sorgente Dump effettuato dal compilatore LISA.

```
*FF:3

*300G
0300- A0 00 84 FE A5 FF A6 FE      ~&~
0308- 20 41 F9 A9 AD 20 ED FD   Ay)- m}
0310- A9 A0 20 ED FD B1 FE 20   )  m}1~
0318- DA FD C8 C0 08 D0 F1 A9   Z}H@ Pq)
0320- A0 20 ED FD 20 ED FD A0     m} m}
0328- 00 A2 00 B1 FE C9 20 90    " 1~I
0330- 08 C9 A0 B0 06 C9 80 90    I 0 I
0338- 02 A9 A0 09 80 20 ED FD    )     m}
0340- E6 FE E8 E0 08 D0 E4 20   f~h` Pd
0348- 8E FD A5 FE D0 02 E6 FF    }%~P
0350- 20 1B FD C9 88 F0 05 C9     }I p I
0358- 9B D0 A9 60 A5 FE 38 E9   P) `%~8i
0360- 10 B0 02 C6 FF 85 FE 4C   0  ~L
0368- 04 03 85 FE 4C 04 03 00     ~L
0370- 00 00 00 00 00 00 00 FF
```

Figura 2 - Dump esadecimale del programma DUMP da caricare in memoria a partire dalla locazione $300. Salvare il tutto su disco con BSAVE DUMP, A$300, L$70.

Come funziona

Per capire come funziona esaminiamo, riga per riga, il listato LISA di figura 1:

Righe 1,2 - Si comunica al compilatore che l'inizio del modulo oggetto è la locazione $300.

Righe 4,8 - Si comunicano al compilatore i nomi e i corrispondenti indirizzi delle routine usate all'interno del programma.

Riga 10 - Si definisce la variabile LOC come il contenuto della locazione $FE; nel corso del programma si farà riferimento anche al contenuto della locazione $FF chiamandolo LOC+1.

Riga 12 - START - la label è di comodo e non verrà richiamata nel corso del programma!

Righe 12,13 - Appoggiandosi al registro Y si azzera la parte bassa dell'indirizzo di partenza ($FE).

Riga 14 - LOOP - Ciclo principale.

Righe 14,16 - La routine del Monitor PRTAX ($F941) stampa sul video alla posizione corrente il contenuto dell'Accumulatore (parte alta) e del registro X (parte bassa) come un numero esadecimale di quattro cifre.

Righe 17,18 - Usando la COUT ($FDED), che stampa il contenuto dell'Accumulatore come carattere ASCII, si stampa un trattino.

Riga 19 - PBYT - primo ciclo per

Come si carica

Passate al monitor con CALL -151, copiate il Dump di *figura 2* e salvate il tutto con BSAVE DUMP, A$300, L$70.

Come si usa

Una volta caricato il programma (BLOAD DUMP) si può effettuare un Dump di memoria sia da Basic che da Monitor. Da Basic basta effettuare una POKE 255, (locazione iniziale / 256) e poi una CALL 768. Dal Monitor (CALL -151) si batta la seguente linea:

*FF:hh N 300G

dove hh è la pagina da cui si vuole che cominci il Dump. Ad esempio: vogliamo effettuare il Dump interpretato della zona di memoria che va da $D000 (.....) a $D1D0 (.....).

Da Basic:

```
POKE 255,208 :REM $D0 = 208
CALL 768
```

Da Monitor:

```
*FF:D0
*300G
```

A questo punto il computer risponde con:

D000 - 6F D8 65 D7 F8 DC 94 D9 oXeWx/Y

e si ferma.

```
]CALL768
D000- 6F D8 65 D7 F8 DC 94 D9  oXeWx\ Y
D008- B1 DB 30 F3 D8 DF E1 DB  1[0sX_a[
D010- 8F F3 98 F3 E4 F1 DD F1   s sdq]q
D018- D4 F1 24 F2 31 F2 40 F2  Tq$r1r@r
D020- D7 F3 E1 F3 E8 F6 FD F6  Wsashv}v.
D028- 68 F7 6E F7 E6 F7 57 FC  hwnwfwW|
D030- 20 F7 26 F7 74 F7 6C F2   w&wtwlr
D038- 6E F2 72 F2 76 F2 7F F2  nrrrvr
D040- 4E F2 6A D9 55 F2 85 F2  NrjYUr r
D048- A5 F2 CA F2 17 F3 BB F3  %rJr s;s
D050- 9E F3 61 F2 45 DA 3D D9   sarEZ=Y
D058- 11 D9 C8 D9 48 D8 F4 03   YHYHXt
D060- 20 D9 6A D9 DB D9 6D D8   YjY[YmX
D068- EB D9 83 E7 C8 D8 AF D8  kY gHX/X
D070- 12 E3 7A E7 D4 DA 95 D8   czgTZ X
D078- A4 D6 69 D6 9F DB 48 D6  $ViV [HV
D080- 90 EB 23 EC AF EB 0A 00   k#l/k
D088- DE E2 12 D4 CD DF FF E2  ^b TMb
D090- 8D EE AE EF 41 E9 09 EF   n.oAi o
D098- EA EF F1 EF 3A F0 9E F0  joqo:p p
D0A0- 64 E7 D6 E6 C5 E3 07 E7  dgVfEc g

D0A8- E5 E6 46 E6 5A E6 86 E6  efFfZf f
D0B0- 91 E6 79 C0 E7 79 A9 E7   fy@gy)g
D0B8- 7B 81 E9 7B 68 EA 7D 96  { i{hj}
D0C0- EE 50 54 DF 46 4E DF 7F  nPT_FN
D0C8- CF EE 7F 97 DE 64 64 DF  O ^dd_
D0D0- 45 4E C4 46 4F D2 4E 45  ENDFORNE
D0D8- 58 D4 44 41 54 C1 49 4E  XTDATAIN
D0E0- 50 55 D4 44 45 CC 44 49  PUTDELDI
D0E8- CD 52 45 41 C4 47 D2 54  MREADGRT
D0F0- 45 58 D4 50 52 A3 49 4E  EXTPR#IN
D0F8- A3 43 41 4C CC 50 4C 4F  #CALLPLO
D100- D4 48 4C 49 CE 56 4C 49  THLINVLI
D108- CE 48 47 52 B2 48 47 D2  NHGR2HGR
D110- 48 43 4F 4C 4F 52 BD 48  HCOLOR=H
D118- 50 4C 4F D4 44 52 41 D7  PLOTDRAW
D120- 58 44 52 41 D7 48 54 41  XDRAWHTA
D128- C2 48 4F 4D C5 52 4F 54  BHOMEROT
D130- BD 53 43 41 4C 45 BD 53  =SCALE=S
D138- 48 4C 4F 41 C4 54 52 41  HLOADTRA
D140- 43 C5 4E 4F 54 52 41 43  CENOTRAC
D148- C5 4E 4F 52 4D 41 CC 49  ENORMALI
```

Figura 3 - Esempio di stampa ottenuta con il programma DUMP. La zona di memoria corrisponde alla Rom dell'Applesoft e in particolare alla tabella delle parole riservate del Basic, chiaramente riconoscibili nella colonna destra del Dump.

stampare otto numeri separati da uno spazio.

Righe19,20 - Si stampa uno spazio.

Riga 21 - Si carica nell'Accumulatore il contenuto della locazione il cui indirizzo si ottiene sommando il valore del registro Y al numero contenuto nelle locazioni $FE e $FF.

Riga 22 - Usando la PRBYT ($FDDA), che stampa il contenuto dell'Accumulatore come un numero esadecimale di due cifre, si stampa il contenuto della locazione appena letta.

Riga 23 - Si incrementa di uno il registro Y in modo che la prossima LDA (LOC), Y legga la locazione successiva.

Righe 24,25 - Si controlla se Y è arrivato ad 8 altrimenti salta a PBYT (riga 19) per ripetere il ciclo.

Righe 26,28 - Finita la parte esadecimale del dump si stampano due spazi per separarla dalla parte ASCII.

Righe 30,31 - Si azzerano tutti e due i registri.

Riga 32 - ASCII - Ciclo ASCII, si rileggono ad uno ad uno gli otto byte nell'Accumulatore.

Righe 33,34 - Se il contenuto è minore di 32 (il più piccolo carattere ASCII stampabile) si salta a NOPE (riga 39);

Righe 35,38 - se non è compreso tra $80 e $A0 (in questa zona ci sono i caratteri di controllo) si salta ad OK;

Riga 39 - NOPE - altrimenti si sostituisce il carattere in Accumulatore con lo spazio.

Riga 40 - OK - Viene effettuato l'OR dell'Accumulatore con il numero binario 10000000, ovvero si forza ad uno il bit più significativo;

Riga 41 - poi si stampa, sempre con la COUT, il contenuto dell'Accumulatore sotto forma di carattere.

Righe 42,45 - Si incrementa LOC di uno per otto volte; il registro X serve da contatore del ciclo, finché è minore di otto si salta ad ASCII.

Riga 46 - La riga è finita; si effettua perciò un a-capo chiamando l'apposita routine del monitor (COUT = $FD8E).

Righe 47,49 - Se LOC è uguale a zero allora è venuto il momento di incrementare anche LOC + 1.

Riga 50 - KEY - Usando la routine del monitor KEYIN ($FD1B), che preleva un carattere dalla tastiera e lo deposita nell'Accumulatore, si attende la pressione di un tasto.

Righe 51,52 - Se il tasto premuto (il cui codice ASCII si trova nell'Accumulatore) era la freccia a sinistra si salta a PREV (riga 56);

Righe 53,54 - se non è un escape allora si continua cosi saltando a LOOP (riga 14);

Riga 55 - altrimenti si ritorna al chiamante (RETURN).

Riga 56 - PREV - Freccia a sinistra e quindi dump all'indietro!

Righe 56,58 - Si sottrae $10 (16) al contenuto di LOC.

Righe 59,60 - Se c'è stato prestito si decrementa anche LOC + 1.

Riga 61 - PREV 1 - Si rimette in LOC il valore aggiornato

Riga 62 - e si riprende il ciclo.

Riga 63 - Si comunica al compilatore che il programma sorgente è finito.

Joystick per Apple: alcune precisazioni

Alcuni lettori ci hanno chiesto come mai i valori dei componenti dello schema di figura 6 non coincidono con la descrizione fatta nel testo. Il motivo è da ricercarsi nel fatto che la descrizione è relativa più che altro alla figura 5, cioè allo schema semplificato, in questo caso infatti i deviatori inseriscono delle resistenze tra i +5.1 volt e un ingresso Game Control (GC). Se però si prelevano i +5.1 volt dall'uscita di un integrato allo stato Alto in realtà ci ritroviamo con +4.2 +4.4 volt, i diodi inoltre provocano un'ulteriore caduta di tensione pari a 0.6 volt e arriviamo così a circa 3.8 volt contro i 5.1 "ufficiali". In queste condizioni, impiegando i valori di resistenza usati per lo schema semplificato ci si ritroverebbe con le temporizzazioni leggermente allungate e quindi con la funzione PDL impossibilitata a scendere al di sotto di 80. Soprattutto la taratura dei trimmer diventa piuttosto critica, potendo usare in pratica solo un quarto di giro.

Ecco perché i trimmer sono stati ridotti a 25 kohm e le resistenze sono diminuite anche loro leggermente; questi valori del resto non sono assolutamente critici e possono essere modificati anche del 30% in più o in meno senza pregiudicare minimamente il funzionamento dell'interfaccia. Unico valore un po' più critico è quello delle resistenze da 1kohm che possono al massimo variare del 10% in più o del 20% in meno, senza scendere al di sotto di 860 ohm.

Consigliamo inoltre di mantenere la lunghezza del cavo di collegamento tra l'interfaccia e il computer entro limiti ragionevoli e non superare possibilmente il metro. Se proprio è necessario un cavo più lungo si può verificare una forte instabilità del valore letto dalla funzione PDL; in questo caso usare un cavo schermato e aggiungere un condensatore da 5 µF tra il positivo e il negativo dell'alimentazione saldato direttamente sulle piste dell'interfaccia.

Nella figura 6 inoltre è stato scritto due volte GC2; il primo in alto dopo GC1 e che fa capo a R16 ed R17 è da intendersi GC0.

I cannon a nove poli sono facilmente reperibili presso la catena dei negozi Melchioni.