Screen Save & Restore

Il secondo lavoro che vi invio è senz'altro il più originale sia come funzione che come soluzioni utilizzate.

Si tratta in breve di un programma che, in un tempo ragionevole, (<0.4 sec), salva la pagina grafica DHGR permettendo poi di recuperarla.

Una utility di questo tipo è utilissima, in quanto permette, volendo ad esempio aprire delle finestre sulla pagina grafica, di ripristinarne poi facilmente il contenuto.

Naturalmente, dati i tempi ridottissimi, questa operazione non coinvolge i dischi, ma si svolge completamente in RAM.

Siccome ogni pagina DHGR occupa 16 Kb non sarebbe stato possibile, a costo di rinunciare a programmare, compiere il salvataggio nella memoria che abitualmente usiamo.

Ho però pensato che il IIc possiede il famoso Auxbank da 64K: perché non sfruttarlo?

È così possibile salvare, senza particolari problemi, fino a due pagine DHGR, che possono salire a tre utilizzando la seconda language card che, essendo però paginata, complica un po' la vita a chi volesse espandere la routine.

Siccome molte utility in memoria occupano spazio prezioso che è invece disponibile in quantità in Auxbank, ho pensato di piazzare là anche la routine stessa ($C00); nella memoria principale (a $300) risiede solo il link che gestisce le chiamate da programma e si occupa di «saltare il banco».

A proposito del metodo di chiamata ho pensato che le antiestetiche POKE & CALL fossero da scartare; rimaneva il solito &, ma ciò avrebbe precluso l'utilizzo di altri tool.

Ho perciò optato per un'altra soluzione che penso sia veramente una novità. Tutti sappiamo che il IIc non gestisce il registratore a cassette; sappiamo però che il Basic che abbiamo in ROM è sostanzialmente equivalente a quello del IIe. Essendo assenti però le routine I/O cassette del monitor, i vari LOAD, SAVE, STORE, RECALL hanno un effetto diverso: quando uno di questi comandi viene invocato, il Basic esegue un JSR $3F5, ovvero al puntatore dell'ampersand (&).

Ciò è facilmente verificabile; infatti una qualsiasi estensione che sfrutti & è richiamabile anche, ad esempio, con RECALL. Provate con l'Editor di MC!

È possibile però riconoscere cosa ha provocato la chiamata analizzando il contenuto del registro Y del processore, che contiene un valore associato al comando Basic eseguito.

Ho perciò sfruttato i due statement STORE e RECALL per implementare la routine descritta.

Avendo poi la possibilità di salvare due pagine distinte, utilizzando due routine Applesoft ne ho modificato la sintassi in

```
STORE (N)
RECALL (N)
```

dove N è il buffer che si vuole utilizzare (attenzione solo 0 e 1!).

Attenzione: se si lavora in PRODOS, digitando in modo diretto STONE(N) si otterrà SYNTAX ERROR. Questo perché STORE è anche un comando PRODOS. Si può ovviare all'inconveniente premettendo i due punti al comando

```
:STORE (N)
```

Da programma Basic invece non c'è alcun problema.

Rimane da chiarire come è possibile usare anche altre estensioni &, dato che viene sfruttato lo stesso puntatore $3F5.

Per ottenere ciò bisogna prima lanciare il tool che usa l'ampersand e poi la nostra routine STORE&RECALL.

Il primo tool setterà naturalmente il puntatore al proprio entry point, e la nostra routine invece... farà lo stesso.

Solo che, oltre a rilocare se stessa nella RAM ausiliaria e a eseguire i vari setup, si «ricorderà» l'indirizzo dell'altra estensione.

Così, ogni volta che uno dei comandi verrà eseguito, il link controllerà se si tratti di STORE o RECALL e, nel caso, agirà opportunamente; in caso contrario (& QUALCHECOSA) salterà all'altro tool che funzionerà come se nessun controllo fosse stato eseguito.

Figura 2

```
2400- D8 4C 47 99 D8 20 3A 99
2408- 4C FC 97 D8 20 3A 99 DA
2410- 5A 48 20 06 99 29 7F C9
2418- 20 90 56 85 FE 64 FF 18
2420- 06 FE 26 FF 06 FE 26 FF
2428- 06 FE 26 FF 64 FA A9 93
2430- 85 FB 18 A5 FE 65 FA 85
2438- FA A5 FF 65 FB 85 FB A5
2440- 25 0A 0A 0A 85 FC A5 24
2448- D0 03 AD 7B 05 65 20 4A
2450- 85 47 B0 03 8D 55 C0 A2
2458- 00 A5 FC 20 16 99 E6 FC
2460- BA AB B1 FA 45 F9 A4 47
2468- 91 26 E8 E0 08 D0 EA F0
2470- 16 C9 0C D0 06 20 5B FC
2478- 20 A6 97 68 48 09 B0 C9
2480- BA F0 10 C9 BD F0 0C A5
2488- 24 D0 03 AD 7B 05 1A C5
2490- 21 D0 0A A5 25 1A C5 23
2498- D0 03 20 7B 98 8D 54 C0
24A0- 68 7A FA 4C 07 C3 A5 F9
24A8- 85 1F 85 1E A5 22 0A 0A
24B0- 0A 85 1D A5 23 0A 0A 0A
24B8- 85 1C A5 20 4A 85 EC 85
24C0- EE 90 02 E6 EE A5 20 18
24C8- 65 21 1A 4A 85 ED 85 EF
24D0- B0 02 C6 ED A6 1C CA 8A
24D8- 20 16 99 A4 EC A5 1E 8D
24E0- 54 C0 91 26 C8 C4 ED D0
24E8- F9 A4 EE A5 1F 8D 55 C0
24F0- 91 26 C8 C4 EF D0 F9 E4
24F8- 1D D0 DB 60 48 5A DA A5
2500- 25 0A 0A 0A 18 69 07 48
2508- 20 16 99 A5 26 85 EE A5
2510- 27 85 EF 68 3A 20 16 99
2518- A5 24 D0 03 AD 7B 05 18
2520- 65 20 4A AB B0 03 8D 55
2528- C0 B1 26 85 1E B1 EE 85
2530- 1F A2 00 AD 00 C0 30 32
2538- 20 58 98 20 76 98 E8 E0
2540- A0 D0 F0 A2 00 AD 00 C0
2548- 30 20 20 61 98 20 76 98
2550- E8 E0 FF D0 F0 4C 31 98
2558- A5 F9 49 FF 91 26 91 EE
2560- 60 A5 1E 91 26 A5 1F 91
2568- EE 60 20 61 98 FA 7A 68
2570- 8D 54 C0 4C 05 C3 A9 10
2578- 4C AB FC A5 22 1A 0A 0A
2580- 0A 85 1D A5 23 0A 0A 0A
2588- 85 1C A5 20 4A 85 EC 85
2590- EE 90 02 E6 EE A5 20 18
2598- 65 21 1A 4A 85 ED 85 EF
25A0- B0 02 C6 ED A6 1D 8A 20
25A8- 16 99 A5 26 85 2A A5 27
25B0- 85 2B 8A 38 E9 08 20 16
25B8- 99 A4 EC 2C 54 C0 B1 2A
25C0- 91 26 C8 C4 ED D0 F7 A4
25C8- EE 2C 55 C0 B1 2A 91 26
25D0- C8 C4 EF D0 F7 E8 E4 1C
25D8- D0 CC A5 1C 38 E9 08 AA
25E0- 8A 20 16 99 A4 EC 2C 54
25E8- C0 A5 F9 91 26 C8 C4 ED
25F0- D0 F9 A4 EE 8D 55 C0 A5
25F8- F9 91 26 C8 C4 EF D0 F9
2600- E8 E4 1C D0 DB 60 48 A5
2608- 32 0A 90 04 A9 00 B0 02
2610- A9 FF 85 F9 68 60 48 0A
2618- 0A 29 1C 85 27 68 4B 6A
2620- 6A 6A 6A 29 03 05 27 09
2628- 20 85 27 68 6A 29 E0 85
2630- 26 6A 6A 29 18 05 26 85
2638- 26 60 2C 50 C0 2C 57 C0
2640- 2C 58 C0 2C 52 C0 60 48
2648- 5A DA 2C 51 C0 20 00 C3
2650- A9 0B 85 36 A9 04 85 38
2658- A9 97 85 37 85 39 20 3A
2660- 99 20 A6 97 FA 7A 68 4C
2668- 0B
```

Figura 3

```
2000- A9 00 85 3C A9 21 85 3D
2008- A9 00 85 3E A9 22 85 3F
2010- A9 0C 85 43 64 42 38 20
2018- 11 C3 A9 00 85 3C A9 22
2020- 85 3D A9 50 85 3E A9 22
2028- 85 3F A9 03 85 43 A9 00
2030- 85 42 A0 00 20 2C FE AD
2038- F7 03 8D 40 03 AD F6 03
2040- 8D 3F 03 A9 03 8D F7 03
2048- A9 00 8D F6 03 60 AD 13
2050- 0F 20 8F 17 8E 03 0A 8C
2058- 04 0A A9 2E 8D 05 0A A9
2060- 20 8D 06 0A 8D 07 0A AD
2068- 13 0F 0A 6D 13 0F AA 52
2070- 9A 10 06 29 7F 18 6D BE
2078- 0F 8D 01 0A 9D B6 0E A5
2080- 9B 10 06 29 7F 18 6D BF
2088- 0F 8D 02 0A 9D B7 0E A0
2090- 00 B1 9C 9D B8 0E AB 18
2098- 69 0B AA 48 B1 9C 9D FF
20A0- 09 CA 8B D0 F7 20 98 1E
20A8- 68 20 C3 14 60 20 A1 1B
20B0- A5 9A 18 69 01 85 B0 A5
20B8- 9B 69 00 85 B1 B1 9A F0
20C0- 06 20 BA 1F 20 C3 14 60
20C8- 8D F4 0F A2 00 A4 96 CB
20D0- 20 15 18 A2 4F AC F4 0F
20D8- BE B4 0E BA 20 BA 1F 98
20E0- 9D FF 09 CA D0 FA 20 98
20E8- 1E AD B4 0E 20 C3 14 60
20F0- 20 0F 20 3E 14 60 23 41
20F8- 57 20 52 49 43 4B 49 45
2100- BA 18 6A 6A 6A 18 69 40
2108- 85 43 18 69 20 85 3F 64
2110- 3E 64 42 A2 C0 A0 28 CA
2118- BA 20 A2 0C 2C 54 C0 88
2120- B1 26 91 3E 2C 55 C0 B1
2128- 26 91 42 C0 00 D0 ED 18
2130- A5 42 69 28 85 42 85 3E
2138- A5 43 90 01 1A 85 43 18
2140- 69 20 85 3F E0 00 D0 CD
2148- 4C 93 0C 8A 18 6A 6A 6A
2150- 18 69 40 85 43 18 69 20
2158- 85 3F 64 3E 64 42 A2 C0
2160- A0 28 CA BA 20 A2 0C 2C
2168- 54 C0 88 B1 3E 91 26 2C
2170- 55 C0 B1 42 91 26 C0 00
2178- D0 ED 18 A5 42 69 28 85
2180- 42 85 3E A5 43 90 01 1A
2188- 85 43 18 69 20 85 3F E0
2190- 00 D0 CD A9 2D 8D ED 03
2198- A9 03 8D EE 03 18 88 4C
21A0- 14 C3 48 0A 0A 29 1C 85
21A8- 27 68 48 6A 6A 6A 6A 29
21B0- 03 05 27 09 20 85 27 68
21B8- 6A 29 E0 85 26 6A 6A 29
21C0- 18 05 26 85 26 60 00 00
21C8- 00 00 00 00 00 00 00 00
21D0- 00 00 00 00 00 00 00 00
21D8- 00 00 00 00 00 00 00 00
21E0- 00 00 00 00 00 00 00 00
21E8- 00 00 00 00 00 00 00 00
21F0- 00 00 00 00 00 00 00 00
21F8- 00 00 00 00 00 00 00 00
2200- C0 4E F0 14 C0 50 D0 36
2208- 20 30 03 A9 00 8D ED 03
2210- A9 0C 8D EE 03 4C 25 03
2218- 20 30 03 A9 4B 8D ED 03
2220- A9 0C 8D EE 03 38 B8 4C
2228- 14 C3 8D 54 C0 4C 8B DE
2230- 20 BB DE 20 FB E6 E0 02
2238- 30 03 4C 99 E1 60 4C 00
2240- BF 13 BA 13 7D 6B 00 D0
2248- C1 13 EB 13 00 34 2A 34
2250- 54 34 7E 34 AB 34 00 D0
2258- BD 34 00 D0 CA 34 D8 34
2260- FC 34 21 35 00 D0 41 35
2268- 69 35 93 35 9E 35 C6 35
2270- EF 35 03 36 2D 36 4B 36
2278- 75 36 9D 36 C4 36 CB 36
2280- F5 36 1E 37 00 D0 39 37
2288- 00 D0 52 37 7B 37 A5 37
2290- CC 37 F6 37 1F 38 00 E9
2298- 00 D0 27 38 51 38 77 38
22A0- 9B 38 C3 38 D4 38 FE 38
22A8- 26 39 42 39 6B 39 92 39
22B0- A5 39 CE 39 F5 39 FB 39
22B8- 21 3A 49 3A 52 3A 79 3A
22C0- 9E 3A CB 3A E1 3A F1 3A
22C8- 17 3B 40 3B 66 3B 8B 3B
22D0- B3 3B DD 3B 04 3C 2E 3C
22D8- 5B 3C 6E 3C 95 3C BA 3C
22E0- CF 3C F6 3C 20 3D 48 3D
22E8- 72 3D 92 3D BC 3D E2 3D
22F0- EB 3D 0F 3E AB B1 D5 B1
22F8- FC B1 C2 47 00 D0 23 3E
2300- 00
```

Istruzioni per l'uso

Anche assemblare questa routine non è una operazione intuitiva, visto che il codice è diviso in tre moduli.

Digitare allora i listati di figura 3.

Salvare quindi il tutto con

BSAVE SCREEN.SYS, A$2000, L$300

Una volta dato il BRUN i comandi saranno disponibili e l'unica zona occupata sarà a $300 (una settantina di byte).

La routine funziona senza problemi con DOS e PRODOS (non usare il disco virtuale /RAM!!!).

Chi volesse cimentarsi in modifiche potrebbe sistemare il BASIC.link tra il BASIC.SYSTEM e i buffer con lo stesso sistema usato per il text.generator, oppure preparare una versione di quest'ultimo che lavori in AUXRAM (non dovrebbe essere particolarmente difficoltoso).

In questo modo, oltre a potere usare l'orologio in contemporanea, si potrebbero avere in memoria le due utility occupando non più di 255 byte di spazio utente!

Tutte le routine descritte fanno uso di codici 65C02 e, sfruttando prerogative del IIc, girano solo su questa macchina; penso però che il TEXT.GEN possa funzionare senza problemi anche sul IIe enhanced.

Mouse facile

di Stefano Riva
Cinisello Balsamo (MI)

Il programma permette di gestire il mouse II da programmi Basic molto più agevolmente e velocemente che tramite le solite operazioni (PRINT CHR$(4) «PR£4»: PRINT CHR$(1): PRINT CHR$(4) «PR£0» per accendere il mouse, PRINT CHR$(4) «IN£4»: INPUT «»; X,Y,S: PRINT CHR$(4) «IN£0» per leggerlo ed infine PRINT CHR$(4) «PR£4»: PRINT CHR$(0): PRINT CHR$(4) «PR£0» per spegnerlo).

Inoltre, il programma esplora automaticamente tutti gli slot del computer per scoprire in quale sia inserito il mouse, cosicché non è necessario che l'utente gli fornisca il numero di slot in cui è alloggiata la periferica.

Utilizzo

Per utilizzare il programma, dopo averlo inserito occorre eseguire CALL 768 per inizializzarlo (se è già residente su disco, è sufficiente utilizzare BRUN invece che BLOAD per caricarlo). Dopo aver fatto questo, sono a disposizione i seguenti comandi:

CALL 918 = Accende il mouse.
CALL 949 = Lo spegne.
USR(1) = Posizione orizzontale del mouse (0 - 1023).
USR(2) = Posizione verticale (0 - 1023).
USR(3) = Stato del pulsante (1 premuto, 0 rilasciato).

L'aspetto più interessante del programma è l'impiego per le letture della funzione USR del Basic, la quale permette una grande semplicità di utilizzo, al posto di astruse PEEK(n) × PEEK(n+1) * 256 che sarebbero necessarie se essa non fosse utilizzata. L'impiego del mouse risulta in tal modo semplice come per joystick e paddle (funzione PDL).

Programma dimostrativo in Basic

```
10  TEXT : HOME : CALL 768: CALL 918
20  PRINT "IL MOUSE E' NELLO SLOT " PEEK (7) - 192"; MUOVILO!": VTAB 10:
    HTAB 10: PRINT "X" SPC( 9)"Y" SPC( 9)"S"
30  VTAB 12: HTAB 10: PRINT  USR (1)"  ": VTAB 12: HTAB 20: PRINT  USR (
    2)"  ": VTAB 12: HTAB 30: PRINT  USR (3)
40  GOTO 30
```

Funzionamento

Nella parte tra $300 ed $32F, il programma esplora gli slot e scrive nella locazione 7 il numero dello slot + $C0 (questo valore sarà in seguito sfruttato dalle altre routine per accedere al firmware del mouse); perciò, se dopo avere inizializzato il programma la locazione 7 viene cambiata per qualsiasi motivo, occorre rifare CALL 768. Se il mouse non è in nessuno slot, viene generato il messaggio d'errore Applesoft «OUT OF DATA ERROR» (sarebbe stato più adatto «I/O ERROR» o «DEVICE NOT CONNECTED» del ProDOS, ma ho preferito non toccare il sistema operativo).

Fra $330 ed $338 viene inizializzata l'istruzione USR, scrivendo l'indirizzo a cui dovrà passare il controllo ($339) nelle locazioni $B ed $C, quindi il controllo ritorna al Basic.

Da $339 a $395 il programma gestisce l'istruzione USR, agendo direttamente sul suo buffer (che parte da $9D) per l'input ed utilizzando la routine delle ROM del Basic a $E2F2 (che scrive nel buffer il valore contenuto nell'accumulatore come byte alto e nel registro Y come byte basso). Se l'argomento della USR non è fra 1 e 3, viene generato il messaggio «ILLEGAL QUANTITY ERROR».

Tra $396 ed $3B4 è situata la routine che accende il mouse e, infine, tra $3B5 ed $3C3 quella che lo spegne.

Listato assemblato

```
*300.3C3

0300- A9 C1 85 07 A0 00 A9 0C
0308- 85 06 B1 06 C9 20 D0 0D
0310- A9 FB 85 06 B1 06 C9 D6
0318- D0 03 4C 30 03 A5 07 C9
0320- C7 D0 03 4C 2B 03 E6 07
0328- 4C 06 03 A2 2A 4C 12 D4
0330- A9 39 85 0B A9 03 85 0C
0338- 60 A9 82 C5 9D D0 3F A9
0340- 80 C5 9E F0 09 A9 C0 C5
0348- 9E F0 1C 4C 99 E1 20 5A
0350- 03 AD FC 05 AC FC 04 4C
0358- F2 E2 A9 14 85 06 A0 00
0360- B1 06 85 06 6C 06 00 20
0368- 5A 03 AD 7C 07 0A B0 07
0370- A0 00 A9 00 4C F2 E2 A0
0378- 01 A9 00 4C F2 E2 A9 81
0380- C5 9D D0 C7 A9 80 C5 9E
0388- D0 C1 20 5A 03 AD 7C 05
0390- AC 7C 04 4C F2 E2 20 A8
0398- 03 A9 12 85 06 A0 00 B1
03A0- 06 85 06 A9 01 6C 06 00
03A8- A9 19 85 06 A0 00 B1 06
03B0- 85 06 6C 06 00 A9 12 85
03B8- 06 A0 00 B1 06 85 06 A9
03C0- 00 6C 06 00
```

Compatibilità

Il programma è stato scritto e verificato sul mio Apple IIc, ma dovrebbe essere utilizzabile su qualsiasi Apple II, dato che durante la stesura ho posto la massima cura nell'evitare istruzioni specifiche del microprocessore 65C02 e, come suddetto, il programma può lavorare su ogni slot; non solo sul numero 4. Per quanto riguarda il sistema operativo, non ci sono problemi: il programma funziona sia in DOS 3.3 che in ProDOS.

Routine del Firmware del mouse utilizzate

Initms =	Inizializza il mouse.
Setms =	Accende o spegne il mouse a seconda del contenuto dell'accumulatore (1 o 0).
Readms =	Legge i dati del mouse (posizione e stato del pulsante) e li trasferisce nei «buchi di schermo».

N.D.R.

Esiste un modo ancora più semplice di utilizzare il mouse: a patto di accontentarsi di un range tra 0 e 255 si può leggere il mouse tramite le funzioni PDL(0) e PDL(1); ovviamente dopo averlo attivato con PR#4; PRINT CHR$(1): PR#0. MC