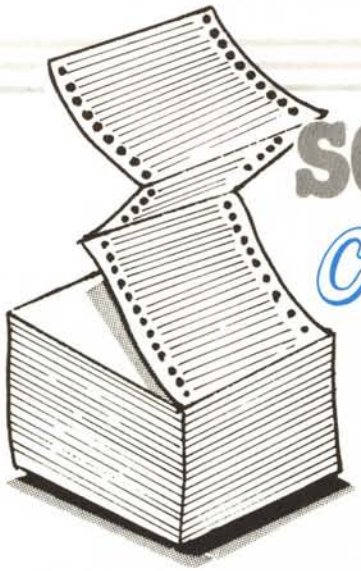

software C-64

a cura di Tommaso Pantuso

File Rescue

di Bert Bunse - Quartu S.E. (CA)

Descrizione programma

«File Rescue» è un'utility per il C64 corredato di disk drive che permette di recuperare dei file dal disco nel caso sia andata alterata o cancellata la directory.

Può infatti capitare che si formatti un disco senza ID per riutilizzarlo (operazione che azzera la BAM e la directory) e ci si accorga a posteriori che vi erano anche uno o più file che non si volevano perdere affatto o addirittura di aver sbagliato disco (ahimé, nessuno è perfetto!). In questo caso si saranno perse completamente tutte le informazioni riguardanti i vari file registrati (soprattutto traccia e settore del loro primo blocco) senza che essi siano tuttavia stati cancellati dal disco.

Può capitare inoltre di aver salvato sul disco un file ricorrendo alla chiocciola in quanto esisteva già un file con lo stesso nome e che successivamente al posto del file salvato ne viene caricato un altro. Questo perché la rispettiva routine del sistema operativo del drive ha messo traccia e settore di un altro file nella corrispondente entrata della directory. Suddetta routine non è affatto perfetta e di solito è meglio non usarla! Comunque anche in questo caso il file in questione non è stato perso perché è già registrato sul disco, solo che non può essere caricato essendo ignoti traccia e settore del suo primo blocco.

Questi sono solo due esempi in cui può essere utile «File Rescue». Esso

Questo programma è disponibile su disco presso la redazione. Vedere l'elenco dei programmi disponibili e le istruzioni per l'acquisto a pag. 223.

```
10 REM ********************************
20 REM *                              *
30 REM *    F I L E    R E S C U E    *
40 REM *                              *
50 REM *           (C) 1987           *
60 REM *                              *
70 REM *          BERT BUNSE          *
80 REM *                              *
90 REM ********************************
100 POKE53280,6:POKE53281,0:GOSUB260
110 CLR:DIMB%,C,L,N,N1,N2,T1,T2,S,U,V,W,X,Y,Z,TR,SE
120 DIMTR$,SE$,NO$,ID$,TY$,X$,Y$,E1$,E2$,E3$,E4$
130 S$="0123456789ABCDEF":CA$="                                        (PUR)"
140 FORX=1TO128:BF$=BF$+CHR$(0):NEXT
150 FORX=1TO18:SR$=SR$+CHR$(160):NEXT
160 DIMBL(35,20),TY$(4)
170 FORX=49152TO49265:READY:POKEX,Y:NEXT
180 FORX=0TO4:READTY$(X):NEXT
190 GOTO520
200 REM ***************
210 REM * SUBROUTINES *
220 REM ***************
230 REM *
240 REM * INTESTAZIONE VIDEO
250 REM *
260 PRINT"(CLR)(PUR)(DOWN)────────────────────────────────────────";
270 PRINTSPC(13)"(WHT)(RVS) FILE  RESCUE "
280 PRINT"(PUR)──────────────────────────────────────";
290 PRINTSPC(10)"(DOWN)(YEL)(C) BERT BUNSE 1987(CYN)"
300 RETURN
310 REM *
320 REM * INPUT S/N/M
330 REM *
340 POKE204,0:POKE198,0
350 GETX$:IFX$<>"N"ANDX$<>"S"ANDX$<>"M"ANDX$<>CHR$(13)GOTO350
360 IFX$=CHR$(13)THENX$="N"
370 IFX$="S"THENPOKE1024+PEEK(214)*40+PEEK(211),147
380 POKE204,1:RETURN
390 REM *
400 REM * LETTURA STATO DISCO
410 REM *
420 POKE211,1:POKE214,20:SYS58640:PRINT"(CYN)STATO DISCO:"
430 INPUT#15,E1$,E2$,E3$,E4$:PRINT"(DOWN)(RGHT)(PUR)"E1$", "E2$","E3$","E4$
440 RETURN
450 REM *****************
460 REM * PRG PRINCIPALE *
470 REM *****************
480 REM *
490 REM * MENU
500 REM *
510 GOSUB260
520 PRINTTAB(6)"(DOWN)(DOWN)(DOWN)RECUPERO FILE ..... (PUR)(RVS) 1 (CYN)"
530 PRINTTAB(6)"(DOWN)MODIFICA DIRECTORY ..... (PUR)(RVS) 2 (CYN)"
540 PRINTTAB(6)"(DOWN)LETTURA DIRECTORY ..... (PUR)(RVS) 3 (CYN)"
550 PRINTTAB(6)"(DOWN)FINE PROGRAMMA ..... (PUR)(RVS) 4 (CYN)"
560 GETX$:IFX$<"1"ORX$>"4"GOTO560
570 ONVAL(X$)GOTO610,1610,2480,3000
580 REM *
590 REM * RECUPERO FILE
600 REM *
610 PRINTTAB(5)"(DOWN)(DOWN)INSERISCI IL DISCO DA ESAMINARE"
620 PRINTTAB(12)"(DOWN)E PREMI 'RETURN'"
630 POKE198,0
640 GETX$:IFX$<>CHR$(13)GOTO640
650 GOSUB260:PRINTTAB(8)"(DOWN)(DOWN)RECUPERO FILE IN CORSO ..."
660 OPEN15,8,15,"M-W"+CHR$(7)+CHR$(28)+CHR$(1)+CHR$(15)
670 OPEN4,8,4,"#1":OPEN5,8,5,"#2"
```

serve infatti in tutte le situazioni nelle quali si è «persa» la corretta entrata nella directory di uno o più file senza che essi siano stati cancellati dal disco. Il programma esamina tutto il disco ritrovando traccia e settore dei primi blocchi dei file e ricostruisce il più fedelmente possibile la directory (v. in seguito).

Punto centrale del programma è quindi il ritrovamento di tutti i «primi» blocchi presenti sul disco. Per fare questo è necessario:

1) distinguere blocchi dati dai blocchi vuoti (contenenti cioè solo i caratteri registrati durante la formattazione del disco);

2) trovare tutti i blocchi dati non puntati dai primi 2 byte di altri blocchi e che sono quindi «primi» blocchi.

Per quanto riguarda il primo punto, il problema a prima vista non sembra di difficile soluzione in quanto i blocchi neoformattati hanno di solito un formato tipo «Kaaaa....», ossia il primo byte è il codice ASCII $4B e tutti gli altri sono lo $01. Questo è vero però solo per una formattazione standard, senza ricorrere cioè ad appositi programmi o cartucce velocizzatori, peraltro di larga diffusione. Utilizzando invece tali accorgimenti, il formato può variare abbastanza ampiamente. Ho esaminato 3 cartucce (speed dos, final cartridge e turbo drive) e 15 programmi diversi (fast format, duplicator II, file hackmen, D-sector V3 ecc.) ed i risultati sono i seguenti:

A) esistono due tipi di formattazione principali: in una il blocco viene riempito con l'ASCII-code $01, nell'altra con l'ASCII-code $00;

B) nel primo caso può variare ampiamente il primo byte, nel secondo caso il secondo byte.

A questo punto, per distinguere tra blocchi dati e blocchi vuoti, «File Rescue» usa una routine in L.M. localizzata nel buffer #2 del drive per non dover trasferire tutti i dati dal drive alla memoria del computer, e che considera blocchi vuoti tutti quelli che hanno il codice ASCII $01 nella posizione da 1 a 255 (escluso cioè il primo byte) e il codice ASCII $00 nella posizione 0 e da 2 a 255 (escluso cioè il secondo byte).

Per la soluzione del secondo problema rimando al commento allegato al listato Basic.

Dal menu principale del programma si può accedere alle seguenti opzioni:

1) Recupero File

Dapprima vengono cercati tutti i «primi» blocchi (tempo di ricerca 3' - 5') e successivamente ne vengono indicati sul video la traccia ed il settore. Qualora si fossero «persi» solo uno o pochi file (come nel secondo esempio sopra) conviene prenderne nota e passare all'opzione «Modifica Directory». Con essa si potranno confrontare traccia e settore dei primi blocchi dei

```
680 PRINT#15,"B-P";5;0
690 FORX=1TO60:READY:PRINT#5,CHR$(Y);:NEXT
700 FORX=1TO4
710 READT1,T2,S
720 FORY=T1TOT2
730 FORZ=0TOS
740 PRINT#15,"U1";4;0;Y;Z:PRINT#15,"U3"
750 PRINT#15,"B-P";5;80
760 GET#5,X$:IFX$=""GOTO810
770 GET#4,TR$,SE$:TR=ASC(TR$+CHR$(0)):SE=ASC(SE$+CHR$(0))
780 IFBL(Y,Z)<>1THENBL(Y,Z)=2
790 IFTR=0GOTO810
800 BL(TR,SE)=1
810 NEXTZ:NEXTY:NEXTX
820 PRINT"(CLR)"TAB(4)"(DOWN)(RVS) PROBABILI 'PRIMI' BLOCCHI SONO: (DOWN)"
830 N=-1
840 FORY=1TO35:FORZ=0TO20
850 IFBL(Y,Z)=2THENPRINT"(PUR) "Y;Z,:N=N+1
860 NEXTZ:NEXTY
870 PRINT:PRINTTAB(5)"(DOWN)(CYN)SCRIVO UNA NUOVA DIRECTORY ? N(LEFT)";:GOSUB340
880 IFX$="N"ORX$="M"GOTO1560
890 REM *
900 REM * SCRITTURA DIRECTORY
910 REM *
920 PRINT:PRINTTAB(7)"(DOWN)NOME DISCO:"TAB(30)"ID:(DOWN)(PUR)"
930 OPEN1,0:POKE647,4
940 POKE211,7:INPUT#1,NO$:IFLEN(NO$)>16GOTO940
950 POKE211,30:INPUT#1,ID$:IFLEN(ID$)<>2GOTO950
960 CLOSE1
970 GOSUB260:PRINTTAB(10)"(DOWN)(DOWN)ATTENDERE, PREGO ..."
980 PRINT#15,"M-W"CHR$(1)CHR$(1)CHR$(1)CHR$(65)
990 PRINT#4,BF$BF$;
1000 FORX=2TO18:PRINT#15,"U2";4;0;18;X:NEXT
1010 PRINT#15,"B-P";4;0
1020 PRINT#4,CHR$(18)CHR$(1)CHR$(65);
1030 PRINT#15,"B-P";4;144
1040 PRINT#4,LEFT$(NO$+SR$,18)ID$CHR$(160)"2A"LEFT$(SR$,4);
1050 PRINT#15,"U2";4;0;18;0
1060 GOSUB420:IFE1$<>"00"GOTO1550
1070 DIMBN(N,1):N=-1
1080 FORY=1TO35:FORZ=0TO20
1090 IFBL(Y,Z)=2THENN=N+1:BN(N,0)=Y:BN(N,1)=Z
1100 NEXTZ:NEXTY
1110 GOSUB260:PRINTTAB(7)"(DOWN)(DOWN)LETTURA FILE ....:"
1120 PRINTTAB(7)"(DOWN)TRACCIA .....:"
1130 PRINTTAB(7)"(DOWN)SETTORE .....:"
1140 PRINTTAB(7)"(DOWN)(DOWN)('RETURN' = PROSSIMO FILE)(PUR)"
1150 N1=INT(N/8):N2=N-(N1*8):N=-1:Z=7:POKE650,128
1160 FORX=0TON1
1170 PRINT#4,BF$BF$;
1180 PRINT#15,"B-P";4;0
1190 IFX=N1THENPRINT#4,CHR$(0)CHR$(255);:Z=N2:READV:GOTO1210
1200 READV:PRINT#4,CHR$(18)CHR$(V);
1210 FORY=0TOZ
1220 L=0:C=130:N=N+1
1230 POKE211,25:POKE214,8:SYS58640:PRINTBN(N,0)"(LEFT) -"BN(N,1)"(LEFT)  "
1240 PRINT#15,"U1";5;0;BN(N,0);BN(N,1)
1250 GET#5,TR$,SE$,X$,Y$
1260 U=ASC(X$+CHR$(0))+ASC(Y$+CHR$(0))*256
1270 TR=ASC(TR$+CHR$(0)):SE=ASC(SE$+CHR$(0))
1280 POKE211,25:POKE214,10:SYS58640:PRINTTR"(LEFT)  "
1290 POKE211,25:POKE214,12:SYS58640:PRINTSE"(LEFT)  "
1300 GETX$:IFX$=CHR$(13)THENC=0:GOTO1340
1310 L=L+1:IFTR=0GOTO1340
1320 PRINT#15,"U1";5;0;TR;SE
1330 GET#5,TR$,SE$:GOTO1270
```

(continua a pagina 220)

Nota

I codici di controllo nei listati sono riportati in forma «esplicita», in conseguenza dell'impiego della stampante Star NL-10 e relativa interfaccia per Commodore. Ovviamente, nella digitazione del programma è necessario usare i consueti tasti che corrispondono alle indicazioni fra parentesi: ad esempio cursore destro per (RGHT), CTRL-3 per (RED) eccetera.

(CLR)	=	(YEL)	=		
(HOME)	=	(RVS)	=		
(DOWN)	=	(OFF)	=		
(UP)	=	(ORNG)	=		
(RGHT)	=	(BRN)	=		
(LEFT)	=	(LRED)	=		
(BLK)	=	(GRY1)	=		
(WHT)	=	(GRY2)	=		
(RED)	=	(LGRN)	=		
(CYN)	=	(LBLU)	=		
(PUR)	=	(GRY3)	=		
(GRN)	=	(SWLC)	=		
(BLU)	=				

vari file, apportando le opportune modifiche. Nel caso sia stata cancellata tutta la directory conviene farla ricostruire dal programma rispondendo «si» alla domanda «Scrivo una nuova directory?». Sarà scritta una directory con un'entrata per ciascun «primo» blocco.

Prima di poter completare tale entrata il programma legge ciascun file dal primo all'ultimo blocco per poterne conoscere la lunghezza. Durante tale lettura vengono indicati traccia e settore dei vari blocchi letti in quanto possono capitare dei dischi che contengono una concatenazione di blocchi la quale manda in un loop infinito. In questo caso basta premere RETURN per uscirne il che può essere usato anche qualora si voglia evitare a priori l'entrata nella directory di un determinato file.

Non essendo possibile la ricostruzione automatica del nome originale di ciascun file, ad ognuno sarà assegnato un nome formato da un numero di ordine crescente e dalla locazione di memoria a partire dalla quale il file verrebbe caricato con un LOAD rilocato. Questo per facilitare l'individuazione del primo file nel caso di programmi consistenti in più parti; esso verrà infatti caricato per lo più a partire dall'inizio dell'area Basic ($0801-2048). Il tipo di file sarà messo per default a PRG.

L'ultima operazione consiste in un VALIDATE per ricostruire la BAM.

2) Modifica Directory

Mediante questa opzione possono essere cambiati nome ed ID del disco nonché tutti i parametri riguardanti i vari file quali nome, tipo, traccia e settore del primo blocco e lunghezza. In questo modo si potrà ricostruire il più fedelmente possibile la directory andata alterata o perduta cambiando soprattutto nome e tipo in una directory ricostruita dal programma.

Per quanto riguarda il tipo di file, esso può essere PRG, SEQ, USR, REL, DEL ed eventualmente protetto da una SCRATCH aggiungendo < (p.es.: PRG<). DEL* sta per un file che è stato cancellato con una SCRATCH e che può essere fatto ricomparire mettendovi una sigla diversa. Viceversa DEL* sarà usato per cancellare un file. Togliendo o mettendo DEL* bisogna successivamente attualizzare la BAM.

N.B. Si può tornare in ogni momento al menu principale premendo M.

3) Lettura Directory

Serve per una veloce lettura della

(segue da pagina 219)

```
1340 B%=0:X$=""
1350 FORW=3TO0STEP-1
1360 U=U-B%*16↑(W+1):B%=U/16↑W:X$=X$+MID$(S$,B%+1,1)
1370 NEXTW
1380 NO$=MID$(STR$(N+1)+" - $"+X$+SR$,2,16)
1390 PRINT#15,"B-P";4;Y*32+2
1400 PRINT#4,CHR$(C)CHR$(BN(N,0))CHR$(BN(N,1))NO$;
1410 PRINT#15,"B-P";4;Y*32+30
1420 PRINT#4,CHR$(L);
1430 NEXTY
1440 IFN<143GOTO1500
1450 PRINT#15,"B-P";4;0
1460 PRINT#4,CHR$(0)CHR$(255);
1470 PRINT#15,"U2";4;0;18;18
1480 POKE211,1:POKE214,22:SYS58640:PRINT"TROPPI FILES !"
1490 FORX=1TO2500:NEXT:GOTO1520
1500 READV:PRINT#15,"U2";4;0;18;V
1510 NEXTX
1520 GOSUB260:PRINTTAB(12)"(DOWN)(DOWN)SCRITTURA BAM ..."
1530 PRINT#15,"V"
1540 GOSUB420
1550 FORX=1TO2500:NEXT
1560 PRINT#15,"M-W"CHR$(7)CHR$(28)CHR$(1)CHR$(58):POKE650,0
1570 CLOSE4:CLOSE5:CLOSE15:GOTO100
1580 REM *
1590 REM * MODIFICA DIRECTORY
1600 REM *
1610 OPEN15,8,15,"I":OPEN5,8,5,"#"
1620 PRINT"(CLR)"TAB(10)"(DOWN)(RVS) MODIFICA DIRECTORY: "
1630 PRINT#15,"M-E"CHR$(183)CHR$(199)
1640 PRINT#15,"M-R"CHR$(177)CHR$(2)CHR$(25)
1650 INPUT#15,NO$
1660 PRINTTAB(8)"(DOWN)(DOWN)NOME DISCO:"TAB(27)"ID:(DOWN)(PUR)"
1670 PRINTTAB(8)NO$
1680 PRINTTAB(14)"(DOWN)(DOWN)(CYN)LO CAMBI ? N(LEFT)";:GOSUB340
1690 IFX$="N"GOTO1810
1700 IFX$="M"GOTO2440
1710 POKE211,0:POKE214,6:SYS58640:PRINTCA$
1720 OPEN1,0:POKE647,4
1730 POKE211,8:POKE214,6:SYS58640:INPUT#1,NO$:IFLEN(NO$)>16GOTO1730
1740 POKE211,27:INPUT#1,ID$:IFLEN(ID$)<>2GOTO1740
1750 CLOSE1
1760 PRINT#15,"U1";5;0;18;0
1770 PRINT#15,"B-P";5;144
1780 PRINT#5,LEFT$(NO$+SR$,18)ID$CHR$(160)"2A"LEFT$(SR$,4);
1790 PRINT#15,"U2";5;0;18;0
1800 GOSUB420:IFE1$<>"00"GOTO2430
1810 POKE650,128:S=1:N=0
1820 POKE211,1:POKE214,4:SYS58640
1830 PRINT"(CYN)LUN.: NOME 1. FILE:"TAB(24)"TIPO:  TR.: SE.:":PRINTCA$
1840 PRINTTAB(14)"(DOWN)(DOWN)(DOWN)(DOWN)(CYN)(MENU - 'M')"
1850 PRINTTAB(4)"(DOWN)(TIPO - PRG SEQ REL USR DEL < *)"
1860 PRINT#15,"U1";5;0;18;S;V-S
1870 GET#5,X$,Y$:T1=ASC(X$+CHR$(0)):S=ASC(Y$+CHR$(0))
1880 FORX=0TO7:N=N+1
1890 PRINT#15,"B-P";5;X*32+2
1900 GET#5,X$,TR$,SE$:Z=ASC(X$+CHR$(0)):TR=ASC(TR$+CHR$(0)):SE=ASC(SE$+CHR$(0))
1910 TY$=TY$(ZAND15)
1920 IFZAND128GOTO1940
1930 TY$=TY$+"*"
1940 IFZAND64THENTY$=TY$+"<"
1950 NO$=""
1960 FORY=1TO16
1970 GET#5,X$:IFX$=CHR$(160)GOTO2010
1980 NO$=NO$+X$
1990 NEXTY
2000 IFNO$=""GOTO2380
2010 PRINT#15,"B-P";5;X*32+30
2020 GET#5,X$:L=ASC(X$+CHR$(0))
2030 POKE211,11:POKE214,4:SYS58640:PRINT"(CYN)"N"(LEFT). FILE"
2040 POKE211,0:POKE214,6:SYS58640:PRINTCA$
2050 POKE214,6:SYS58640:PRINTLTAB(7)NO$TAB(24)TY$TAB(30)TRTAB(35)SE
2060 POKE211,25:POKE214,9:SYS58640:PRINT"(CYN)N(LEFT)";:GOSUB340
2070 IFX$="N"GOTO2360
2080 IFX$="M"GOTO2380
2090 POKE211,0:POKE214,6:SYS58640:PRINTCA$
2100 OPEN1,0:POKE647,4
2110 POKE211,0:POKE214,6:SYS58640:PRINTL;:POKE211,1:INPUT#1,X$:L=VAL(X$)
2120 IFL<1ORL>255GOTO2110
2130 POKE211,7:POKE214,6:SYS58640:PRINTNO$;:POKE211,7:INPUT#1,NO$
2140 POKE211,24:PRINTTY$;:POKE211,24:INPUT#1,TY$
2150 Z=128
2160 FORY=0TO4:IFLEFT$(TY$,3)=TY$(Y)THENZ=Z+Y:GOTO2190
2170 NEXTY
```

```
2180 GOTO2140
2190 IFMID$(TY$,4,1)="<"THENZ=ZOR64
2200 IFMID$(TY$,4,1)="*"THENZ=ZAND127
2210 POKE211,30:PRINTTR;:POKE211,31:INPUT#1,TR$:TR=VAL(TR$)
2220 IFTR>35ORTR<1GOTO2210
2230 POKE211,35:PRINTSE;:POKE211,96:INPUT#1,SE$:SE=VAL(SE$)
2240 IFSE>20ORSE<0GOTO2230
2250 CLOSE1
2260 POKE211,25:POKE214,9:SYS58640:PRINT"(CYN)N(LEFT)";:GOSUB340
2270 IFX$="S"GOTO2090
2280 IFX$="M"GOTO2380
2290 PRINT#15,"B-P";5;X*32+2
2300 PRINT#5,CHR$(Z)CHR$(TR)CHR$(SE)LEFT$(NO$+SR$,16);
2310 PRINT#15,"B-P";5;X*32+30
2320 PRINT#5,CHR$(L);
2330 PRINT#15,"U2";5;0;18;V
2340 POKE211,0:POKE214,22:SYS58640:PRINTCA$
2350 GOSUB420:IFE1$<>"00"GOTO2430
2360 NEXTX
2370 IFT1=18GOTO1860
2380 POKE650,0:GOSUB260:PRINTTAB(9)"(DOWN)(DOWN)ATTUALIZZO LA BAM ? N(LEFT)";:GO
SUB340
2390 IFX$="N"ORX$="M"GOTO2440
2400 GOSUB260:PRINTTAB(12)"(DOWN)(DOWN)SCRITTURA BAM ..."
2410 PRINT#15,"V"
2420 GOSUB420
2430 FORX=1TO2500:NEXT
2440 CLOSE5:CLOSE15:GOTO510
2450 REM *
2460 REM * LETTURA DIRECTORY
2470 REM *
2480 PRINT"(CLR)"
2490 OPEN15,8,15,"I":CLOSE15:SYS49152
2500 PRINTTAB(16)"(DOWN)(DOWN)'RETURN'"
2510 POKE198,0
2520 GETX$:IFX$<>CHR$(13)GOTO2520
2530 GOTO510
2540 REM *
2550 REM * DATA LETTURA DIRECTORY
2560 REM *
2570 DATA 169,36,133,251,169,251,133,187
2580 DATA 169,0,133,188,169,1,133,183
2590 DATA 169,8,133,186,169,96,133,185
2600 DATA 32,213,243,165,186,32,180,255
2610 DATA 165,185,32,150,255,169,0,133
2620 DATA 144,160,3,32,165,255,133,251
2630 DATA 32,165,255,133,252,165,144,208
2640 DATA 53,136,208,239,32,228,255,201
2650 DATA 13,240,43,169,6,133,211,165
2660 DATA 252,166,251,32,205,189,169,32
2670 DATA 32,210,255,32,165,255,166,144
2680 DATA 208,20,201,0,240,6,32,210
2690 DATA 255,76,83,192,169,13,32,210
2700 DATA 255,160,2,76,43,192,32,66
2710 DATA 246,96
2720 REM *
2730 REM * DATA TIPO FILE
2740 REM *
2750 DATA"DEL","SEQ","PRG","USR","REL"
2760 REM *
2770 REM * DATA RECUPERO FILE
2780 REM *
2790 DATA 169,255,141,80,5,173,2,4
2800 DATA 201,1,240,5,201,0,240,19
2810 DATA 96,162,1,189,0,4,201,1
2820 DATA 208,246,232,208,246,169,0,141
2830 DATA 80,5,96,173,0,4,201,0
2840 DATA 208,248,162,3,189,0,4,201
2850 DATA 0,208,239,232,208,246,169,0
2860 DATA 141,80,5,96
2870 REM *
2880 REM * DATA TRACCIA - SETTORE
2890 REM *
2900 DATA1,17,20,19,24,18,25,30,17,31,35,16
2910 REM *
2920 REM * DATA SCRITTURA TRACCIA 18
2930 REM *
2940 DATA4,1,7,4,10,7,13,10,16,13
2950 DATA2,16,5,2,8,5,11,8,14,11,17,14
2960 DATA3,17,6,3,9,6,12,9,15,12,18,15,0,18
2970 REM *
2980 REM * FINE
2990 REM *
3000 PRINT"(CLR)(LBLU)";:POKE53280,14:POKE53281,6:POKE647,14
```

directory senza dover uscire dal programma. Poiché la lettura mediante Basic è lenta, viene usata una breve routine in L.M.

Premendo RETURN si può arrestare la lettura a qualsiasi punto.

4) Fine Programma

Resetta il video senza cancellare il programma dalla memoria. In conclusione si può dire che:

a) - «File Rescue» è adatto soprattutto per recuperare file a pezzo unico e che non siano file relativi;

b) - non funzionerà invece con i file relativi dei quali la complessa struttura può essere solo difficilmente ricostruita;

c) - nel caso di file che vengano caricati in più parti, bisogna individuare il primo file e trovare in esso i nomi di quelli che vengono caricati successivamente pena un FILE NOT FOUND error (per quanto riguarda il tipo di file, basta fare qualche prova);

d) - nel caso di dischi sui quali si sono ripetutamente salvati e cancellati dei file, saranno trovati più «primi» blocchi di quelli realmente esistenti; basterà caricarli uno ad uno per rendersi conto di quali siano quelli giusti.

Commento al listato Basic

poke211,X = posizionamento orizzontale del cursore

poke214,X:sys58640 = posizionamento verticale del cursore

open1,0:input#1,X = input senza "?"

100 = inizializzazione video

110-160 = definizione variabili

170 = allocamento routine L.M. per la lettura della directory a partire da 49152

340 - 380 = input sì/no/menu

340 = accende il cursore, svuota il buffer della tastiera

370 = stampa "S" = sì nell'opportuna posizione del video

380 = spegne il cursore

660 = allunga l'intervallo tra un interrupt e l'altro nel drive aumentando la velocità d'accesso al disco

690 = allocamento routine L.M. per il recupero file nel buffer #2 del driver

700 - 810 = carica tutti i blocchi del disco a prescindere di quelli della traccia 18 nel buffer #1 del drive

740 = esegue la routine L.M. nel buffer #2

750 - 760 = se flag = $00 (blocco vuoto) → BL(y,z) = " "

770 - 800 = se flag = $FF (blocco dati) → BL(y,z) = 1 se puntato da un altro blocco e quindi non "primo" blocco o BL(y,z) = 2 se non puntato da nessun altro blocco

820 - 860 = output sul video

920 - 960 = input nome - id disco per la nuova directory

980 - 1060 = inizializzazione traccia 18

980 = necessario per rendere possibile

la scrittura sul disco se dovesse essere stato alterato il terzo byte di 18/0 (codifica la compatibilità delle versioni DOS e viene scritto nella locazione di memoria 257 del drive durante l'inizializzazione del disco)

990 = azzera il buffer #1

1000 = azzera 18/2-18

1020 - 1050 = scrive 18/0

1070 - 1100 = trascrive traccia e settore dei "primi" blocchi in BN(N,0-1)

1150 - 1510 = produce le varie entrate nella directory

1150 = calcola quanti blocchi della traccia 18 servono e quanti file devono essere scritti nell'ultimo blocco

1190 = primi due byte per l'ultimo blocco

1200 = concatenazione col prossimo blocco

1240 - 1330 = legge e calcola la locazione di memoria (=U) a partire dalla quale verrebbe caricato il file e determina la sua lunghezza; se "return" in 1300 il tipo di file sarà = DEL* (C=0) ed esso verrà trattato come un file cancellato con una SCRATCH

1340 - 1370 = trasforma U in numero esadecimale

1400 - 1420 = scrive l'entrata per ciascun file (tipo, traccia - settore del primo blocco, nome, lunghezza)

1450 - 1490 = se più di 144 "primi" blocchi → fine

1560 = normalizza l'interrupt del drive

1630 = esegue la routine $C7B7 del DOS (essa prepara l'intestazione della directory e la scrive nel buffer "output directory" in $02B1)

1640 - 1650 = preleva nome - id disco

1660 - 1800 = cambia nome - id disco

1810 — 2370 = cambia i parametri dei vari file (lunghezza, nome, tipo, traccia - settore del primo blocco)

Lista Variabili

C = tipo file nella scrittura della directory; per default = 130 (PRG)

L = lunghezza file

U = locazione di caricamento di un file

BL(35,20) = 2 se "primo" blocco

BN(N, 0-1) = traccia - settore "primo" blocco

N = numero di "primi" blocchi trovati

N1, N2 = numero di blocchi della directory da scrivere e numero di file da scrivere nell'ultimo blocco

V = settore della traccia 18 da leggere o scrivere

T1, T2, S, TR, SE, TR$, SE$ = traccia - settore

NO$ = nome disco - file

ID$ = id disco

TY$, TY$(4) = tipo file

E1$ - E4$ = stato disco

B%, S$ = trasformazione decimale - esadecimale

BF$ = per azzerare il buffer del drive

SR$ = spazi reverse per la scrittura della directory

CA$ = per cancellare una linea dal video

W, X, Y, Z, X$, Y$ = variabili ad uso vario (cicli for - next, input da tastiera o drive ecc.)

Routine L.M. Lettura Directory

La routine verrà caricata nella memoria del computer a partire da $C000 (49152) da parte del programma principale.

$C000 - $C018 = apre il file "$" sul bus IEC
$C01B - $C022 = manda talk e indirizzo secondario per load al drive
$C025 - $C027 = azzera lo stato
$C029 - $C03A = salta i primi 6 byte; stato <> 0 ? ($C035) → fine ($C06E)
$C03C - $C041 = premuto "return" ? → fine
$C043 - $C045 = poke211,6 → posiziona il cursore sulla linea
$C047 - $C04B = trasformazione di 16 bit in numero decimale ed output
$C04E - $C050 = output "spazio"
$C053 - $C058 = input prossimo byte; stato <> 0 ? → fine
$C05A - $C05C = fine linea ? → carriage return ($C064)
$C05E - $C061 = output byte ed input prossimo byte ($C053)
$C064 - $C06B = carriage return ed input prossima linea ($C02B)
$C06E - $C071 = chiude il file → fine

```
.
.,C000 A9 24     LDA #$24
.,C002 85 FB     STA $FB
.,C004 A9 FB     LDA #$FB
.,C006 85 BB     STA $BB
.,C008 A9 00     LDA #$00
.,C00A 85 BC     STA $BC
.,C00C A9 01     LDA #$01
.,C00E 85 B7     STA $B7
.,C010 A9 08     LDA #$08
.,C012 85 BA     STA $BA
.,C014 A9 60     LDA #$60
.,C016 85 B9     STA $B9
.,C018 20 D5 F3  JSR $F3D5
.,C01B A5 BA     LDA $BA
.,C01D 20 B4 FF  JSR $FFB4
.,C020 A5 B9     LDA $B9
.,C022 20 96 FF  JSR $FF96
.,C025 A9 00     LDA #$00
.,C027 85 90     STA $90
.,C029 A0 03     LDY #$03
.,C02B 20 A5 FF  JSR $FFA5
.,C02E 85 FB     STA $FB
.,C030 20 A5 FF  JSR $FFA5
.,C033 85 FC     STA $FC
.,C035 A5 90     LDA $90
.,C037 D0 35     BNE $C06E
.,C039 88        DEY
.,C03A D0 EF     BNE $C02B
.,C03C 20 E4 FF  JSR $FFE4
.,C03F C9 0D     CMP #$0D
.,C041 F0 2B     BEQ $C06E
.,C043 A9 06     LDA #$06
.,C045 85 D3     STA $D3
.,C047 A5 FC     LDA $FC
.,C049 A6 FB     LDX $FB
.,C04B 20 CD BD  JSR $BDCD
.,C04E A9 20     LDA #$20
.,C050 20 D2 FF  JSR $FFD2
.,C053 20 A5 FF  JSR $FFA5
.,C056 A6 90     LDX $90
.,C058 D0 14     BNE $C06E
.,C05A C9 00     CMP #$00
.,C05C F0 06     BEQ $C064
.,C05E 20 D2 FF  JSR $FFD2
.,C061 4C 53 C0  JMP $C053
.,C064 A9 0D     LDA #$0D
.,C066 20 D2 FF  JSR $FFD2
.,C069 A0 02     LDY #$02
.,C06B 4C 2B C0  JMP $C02B
.,C06E 20 42 F6  JSR $F642
.,C071 60        RTS
```

Routine L.M. Recupero File

La routine verrà caricata nel buffer del drive #2 ($0500 - $05FF) da parte del programma principale (il listato è stato ottenuto invece caricandola nella memoria del computer a partire da $5000).

$0500 - $0502 =
mette il flag ($0550) "blocco vuoto" - "blocco dati" a $FF = "blocco dati"

$0505 - $0510 =
carica il terzo byte del blocco da esaminare (che è stato caricato dalla parte basic nel buffer #1 = $0400 - $04ff); se = $01 → $0511, se = $00 → $0523, se <> $01 o $00 → fine, si tratta di "blocco dati"

$0511 - $0522 =
se tutti i byte escluso il primo = $01 → flag = $00 = "blocco vuoto", fine

$0523 - $053B =
se tutti i byte escluso il secondo = $00 → flag = $00, fine

```
.
.,5000 A9 FF     LDA #$FF
.,5002 8D 50 05  STA $0550
.,5005 AD 02 04  LDA $0402
.,5008 C9 01     CMP #$01
.,500A F0 05     BEQ $5011
.,500C C9 00     CMP #$00
.,500E F0 13     BEQ $5023
.,5010 60        RTS
.,5011 A2 01     LDX #$01
.,5013 BD 00 04  LDA $0400,X
.,5016 C9 01     CMP #$01
.,5018 D0 F6     BNE $5010
.,501A E8        INX
.,501B D0 F6     BNE $5013
.,501D A9 00     LDA #$00
.,501F 8D 50 05  STA $0550
.,5022 60        RTS
.,5023 AD 00 04  LDA $0400
.,5026 C9 00     CMP #$00
.,5028 D0 F8     BNE $5022
.,502A A2 03     LDX #$03
.,502C BD 00 04  LDA $0400,X
.,502F C9 00     CMP #$00
.,5031 D0 EF     BNE $5022
.,5033 E8        INX
.,5034 D0 F6     BNE $502C
.,5036 A9 00     LDA #$00
.,5038 8D 50 05  STA $0550
.,503B 60        RTS
```

Elenco del software disponibile su cassetta o minifloppy

Per ovviare alle difficoltà incontrate da molti lettori nella digitazione dei listati pubblicati nelle varie rubriche di software sulla rivista, MCmicrocomputer mette a disposizione i programmi più significativi direttamente su supporto magnetico. Riepiloghiamo qui sotto i programmi disponibili per le varie macchine, ricordando che i titoli non sono previsti per computer diversi da quelli indicati. Il numero della rivista su cui viene descritto ciascun programma è riportato nell'apposita colonna; consigliamo gli interessati di procurarsi i relativi numeri arretrati, eventualmente rivolgendosi al nostro Servizio Arretrati utilizzando il tagliando pubblicato in fondo alla rivista.

Per l'ordinazione inviare l'importo (a mezzo assegno, c/c o vaglia postale) alla Technimedia srl, Via Carlo Perrier 9, 00157 Roma.

Codice	Titolo programma	MC n.	Prezzo	Note
	APPLE II			
DA2/00	Shape Tablet	22	15000	
DA2/01	Motomuro	26	15000	
DA2/02	&DEBUG	28	15000	
DA2/03	EDIT + INPUT	29	15000	
DA2/04	Basic modulare	34	15000	
DA2/05	ANNA Animation Lang.	35/37	15000	
DA2/06	Miniset + Leva-DOS	37	15000	
DA2/07	27 programmi grafici	38	30000	
DA2/08	Adventure Editor	38	15000	
DA2/09	Animazione funzioni	42	15000	
DA2/10	IL mondo di WA-TOR	43	15000	
DA2/11	Contest LOG	43	15000	
DA2/12	Rout.grafiche estese	44	15000	
DA2/13	Scroll 300 righe	46	15000	
DA2/14	Assembler in Basic	50	15000	
DA2/15	G-Basic II	53	15000	
DA2/16	Disk Editor	54	15000	
DA2/17	Latino	57	15000.	
DA2/18	Battaglia	61	15000	
DA2/19	Catalogo	64	15000	
	COMMODORE AMIGA			
DAM/01	F-15	63	15000	
DAM/02	Gest.list programmi	64	15000	
	COMMODORE 128			
C28/01	MMCalc	53	17000	
C28/03	Mega Bank 128	56	17000	
D28/01	MMCalc	53	15000	
D28/02	Hardcopy 128	55	15000	
D28/03	SheetIt	57	15000	
D28/04	Star Quest	58	15000	
D28/05	Family Budget	60	15000	
D28/06	La casa stregata	61	15000	
D28/07	Strutt 80/33	63	15000	
D28/08	Bas-80 V2.0a	64	15000	
	COMMODORE 64			
C64/01	Briscola	25	17000	
C64/02	Serpentone	29	17000	
C64/03	Othello	29	17000	
C64/04	Chase	33	17000	
C64/05	Spreadsheet	34	30000	
C64/06	Bilancio familiare	35	17000	
C64/07	The dark wood	36	17000	
C64/08	Totocalcio: sis.rid.	37	17000	
C64/09	Orchetes	37	17000	
C64/10	Wordprocessor	38	17000	
C64/11	Helicopt	38	17000	
C64/12	Finestra grafica	39	17000	
C64/13	Paroliamo	39	17000	
C64/14	Scarabeo	40	17000	
C64/15	Magazzino	41	17000	
C64/16	Rubrica	44	17000	
C64/17	World	45	17000	
C64/18	P.J.T. Basic	46	17000	
C64/19	Sistema Enalotto	47	17000	
C64/20	Simulat.reti logiche	48	17000	
C64/21	RTTY	48	17000	
C64/22	Mescola	49	17000	
C64/23	Othello	51	17000	
C64/24	Voters	51	17000	
C64/25	Flashtape	50/51	17000	
C64/26	Cross Reference	53	17000	
C64/27	Flib	54	17000	
C64/28	Boz's Adventure	57	17000	
D64/01	Spreadsheet	34	15000	
D64/02	ADP Basic da 35 a	39	15000	
D64/03	Wordprocessor	38	15000	
D64/04	Paroliamo	39	15000	
D64/05	Data base Galileo	40/41	15000	
D64/06	Magazzino	41	15000	
D64/07	Gestione biblioteca	46	15000	
D64/08	P.J.T. Basic	46	15000	
D64/09	Simulat.reti logiche	48	15000	
D64/10	Archiprog	50	15000	
D64/11	Anno Domini	57	15000	
D64/12	The Disk Editor	54/6/7	15000	
D64/13	Boz's Adventure	57	15000	
D64/14	Link-64	57	30000	
D64/15	New Char 2.2	58	15000	
D64/16	Music 64	59	15000	
D64/17	TRX-MEM	59	15000	
D64/18	WOS + WBasic	60	15000	
D64/19	Strange Basic + Dracula	63	15000	
D64/20	File Rescue	64	15000	
D64/21	La Casa	64	15000	
	MSX			
CMX/01	Sound editor	42	17000	
CMX/02	WP Reporter	43	30000	
CMX/03	Foresta maledetta	44	17000	
CMX/04	Monitor disassembler	45	17000	
CMX/05	Video Art	46	17000	
CMX/06	Othello	47	17000	
CMX/07	Joe's Chicken	48	17000	
CMX/08	Planet Hunter	49	17000	
CMX/09	Dune	50	17000	
CMX/10	Ramboman	51	17000	
CMX/11	Worm	52	17000	
CMX/12	Controparola	53	17000	
CMX/13	Shape Editor	54	17000	
CMX/14	Labirinto 3D	55	17000	
CMX/15	Fred	56	17000	
CMX/16	Il tesoro dei pirati	57	17000	
CMX/17	Omino	58	17000	
CMX/18	Toto 13	60	17000	
CMX/19	Painter	62	17000	
CMX/20	MSX Bank	63	17000	
DMX/01	Toto 13	60	15000	
DMX/02	Painter	62	15000	
DMX/03	MSX Bank	63	15000	
	SINCLAIR SPECTRUM			
CSS/01	TRILAB	28	17000	
CSS/02	SET di caratteri	27/29	17000	
CSS/03	Grafica TREDIM	29	17000	
CSS/04	Ippica	30	17000	
CSS/05	Graphic-Comp	32	17000	48 K RAM
CSS/06	Macchina del tempo	34	17000	48 K RAM
CSS/07	Piramide di Iunnuh	35	17000	48 K RAM
CSS/08	Over Basic	37	17000	48 K RAM
CSS/09	Prospettiva	38	17000	48 K RAM
CSS/10	Motomuro	39	17000	48 K RAM
CSS/11	Othello	40	17000	
CSS/12	The dark wood	40	17000	48 K RAM
CSS/13	Musica	41	17000	48 K RAM
CSS/14	Calcolo matriciale	42	17000	48 K RAM
CSS/15	Database	42	17000	
CSS/16	Snake	43	17000	
CSS/17	Life	44	17000	
CSS/18	Horses	45	17000	48 K RAM
CSS/19	42 colonne	46	17000	
CSS/20	3D Pacman	46	17000	48 K RAM
CSS/21	Forza 4	47	17000	48 K RAM
CSS/22	ZX Editor	47	17000	48 K RAM
CSS/23	Wa-Tor	48	17000	48 K RAM
CSS/24	Meta	49	17000	
CSS/25	Graphic Macro Lang.	49	17000	
CSS/26	Super Monitor	50	17000	48 K RAM
CSS/27	Database 64 colonne	50	17000	48 K RAM
CSS/28	MC Basic	52	17000	48 K RAM
CSS/29	Spectrum LOGO	53	17000	
CSS/30	Disassembler	54	17000	48 K RAM
CSS/31	Istogrammi	55	17000	48 K RAM
CSS/32	Finestre	56	17000	48 K RAM

Nota:
l'iniziale del codice e' C per le cassette, D per i floppy