```
00001                       NAM    EDOS     2.4
00002                       TTL    VERSION #2.4 - SUPPORTS MACRO ASSEMBLER & LINKING LOADER
00003                       OPT    O,S
00004                       OPT    NOG
00005                       OPT    CREF,LLEN=120
00006                *
00007                *ASCII CHARACTER DEFINITIONS
00008                *
00009         000D  A CR     EQU    $0D      ASCII CARRIAGE RETURN <CR>
00010         000A  A LF     EQU    $0A      ASCII LINE FEED <LF>
00011         0010  A DLE    EQU    $10      ASCII DLE
00012         0020  A SPC    EQU    $20      ASCII SPACE
00013         001B  A ESC    EQU    $1B      ASCII ESCAPE
00014         001A  A ASUB   EQU    $1A      ASCII SUBSTITUTE CHAR
00015         0012  A DC2    EQU    $12      ASCII DEVICE CONTROL #2
00016         0014  A DC4    EQU    $14      ASCII DEVICE CONTROL #4
00017         00FF  A RUB    EQU    $FF      ASCII RUBOUT CHARACTER
00018                *
00019                *DEFINITION OF CONSTANTS
00020                *
00021         0001  A LBYT   EQU    $1       DISPLACEMENT TO LOWER BYTE OF 2-BYTE VALUE
00022         0100  A HBYT   EQU    $100     DIVISOR TO GENERATE SINGLE UPPER BYTE OF 2-BYTE VALUE
00023                *
00024         0082  A SYSSZ  EQU    130      EDOS IS PERMANENTLY ALLOCATED 130 SECTORS (SYSTEM AREA)
00025         FFFF  A TIMAX  EQU    $FFFF    MAXIMUM TIMER COUNT-DOWN PERIOD
00026                *
00027                *EXTERNAL ROUTINE ADDRESS DEFINITIONS
00028                *
00029         E806  A INTIO  EQU    $E806    EDOS ROM DISK I/F INITIALIZATION ROUTINE
00030         E809  A XRI    EQU    $E809    EDOS ROM DISK BYTE READ ROUTINE
00031         E80C  A XWRT   EQU    $E80C    EDOS ROM DISK WRITE ROUTINE
00032         E815  A PROG   EQU    $E815    EDOS ROM LOAD A PROGRAM & GO TO EXBUG
00033         E824  A EDITR  EQU    $E824    EDOS ROM LOAD A PROGRAM, START IT @ 20 HEX
00034         EA03  A XUS    EQU    $EA03    EDOS ROM TRANSMIT UNIT/SECTOR(TRANSLATED) TO DISK
00035         EA22  A XUSP   EQU    $EA22    EDOS ROM TRANSMIT UNIT/ PHYSICAL SECTOR TO DISK
00036         EA2B  A SEEK   EQU    $EA2B    EDOS ROM SEEK SELECTED UNIT TO TRACK IN A
00037         EA38  A RFLAG  EQU    $EA38    EDOS ROM RESET ERROR FLAG
00038         EA3E  A LOOP   EQU    $EA3E    EDOS ROM ISSUE COMMAND TO DISK (LOOP ON BUSY)
00039         EAB0  A LPTC   EQU    $EAB0    EDOS ROM LINE PRINTER PRINT 1 CHARACTER
00040         EB04  A INITR  EQU    $EB04    EDOS ROM INITIALIZE H/S P/T READER
00041         EB02  A RDRIN  EQU    $EB02    EDOS ROM PAPER TAPE READER INPUT
00042                *
00043         F018  A PRINC  EQU    $F018    EXBUG PRINT SINGLE CHARACTER (ON CONSOLE)
00044         F9CF  A OCHAR  EQU    $F9CF    EXBUG OUTPUT CHARACTER (NO NULLS)
00045         FF02  A SPEED  EQU    $FF02    EXBUG SPEED FLAG 0 = 10, 1 = 30, FF = 120
00046         FF62  A EXPUN  EQU    $FF62    EXBUG PUNCH FLAG
00047         FF8A  A EXBST  EQU    $FF8A    EXBUG STACK ALLOCATION START ADDRESS
00048         F564  A EXBUG  EQU    $F564    EXBUG NORMAL ENTRY POINT
```

```
00050                  *PERIPHERAL INTERFACE ADDRESS DEFINITIONS
00051                  *
00052          EC00  A DKDID  EQU    $EC00    INPUT DATA FROM DISK
00053          EC01  A DKDIC  EQU    $EC01    INPUT DATA CONTROL
00054          EC02  A DKCOD  EQU    $EC02    OUTPUT COMMANDS TO DISK
00055          EC03  A DKCOC  EQU    $EC03    OUTPUT COMMAND CONTROL
00056          EC04  A PTDAT  EQU    $EC04    PAPER TAPE READER/PUNCH DATA
00057          EC05  A PTCTL  EQU    $EC05    PAPER TAPE READER/PUNCH CONTROL & STATUS
00058          EC06  A DKDOD  EQU    $EC06    OUTPUT DATA TO DISK
00059          EC07  A DKDOC  EQU    $EC07    OUTPUT DATA CONTROL
00060                  *
00061          FCF4  A ACIAS  EQU    $FCF4    ACIA STATUS REGISTER
00062          FCF4  A ACIAC  EQU    $FCF4    ACIA CONTROL REGISTER
00063          FCF5  A ACIAR  EQU    $FCF5    ACIA RX DATA BUFFER
00064          FCF5  A ACIAT  EQU    $FCF5    ACIA TX DATA BUFFER
00065          FCFD  A SBITC  EQU    $FCFD    STOP BIT CONTROL FOR ACIA
```

```
00067                  *DATA AREA ADDRESS ASSIGNMENTS
00068                  *
00069          0000  A PASS   EQU    $0       ASSEMBLY PASS
00070          0001  A OFILE  EQU    $1       OUTPUT FILE CLOSURE FLAG
00071          0002  A OUNIT  EQU    $2       OUTPUT FILE UNIT
00072                  *
00073          0003  A IUNIT  EQU    $3       INPUT FILE <UNIT><SECTOR>
00074          0004  A ISIZE  EQU    $4       INPUT FILE SIZE (2 BYTES)
00075          0006  A ITRK   EQU    $6       INPUT FILE TRACK
00076          0007  A ISCTR  EQU    $7       INPUT FILE SECTOR
00077          0008  A ICNTR  EQU    $8       INPUT FILE BYTE COUNTER
00078                  *
00079          0009  A OSIZE  EQU    $9       OUTPUT FILE SIZE (2 BYTES)
00080          000B  A OTRK   EQU    $B       OUTPUT FILE TRACK
00081          000C  A OSCTR  EQU    $C       OUTPUT FILE SECTOR
00082          000D  A OCNTR  EQU    $D       OUTPUT FILE BYTE COUNTER
00083                  *
00084          000E  A TITRK  EQU    $E       SAVE LOCATION FOR INPUT FILE TRACK NO.
00085          000F  A TISZE  EQU    $F       SAVE LOCATION FOR INPUT FILE SIZE (TWO BYTES)
00086                  *
00087                  *SWITCH & JUMP VECTORS ARE FOUND AT $20 TO $26 AND $30 TO $39
00088                  *THEY ARE USED TO DIRECT I/O AND AS PROGRAM START POINTS
00089                  *SEE THE CODE LISTING FOR SYMBOLS & ASSIGNMMENTS
00090                  *
00091                  *EDOS EXECUTIVE STACK ALLOCATION RUNS FROM $6B TO $3A
00092                  *
00093          006B  A EXSTK  EQU    $6B      EDOS EXECUTIVE STACK BEGINS HERE ^
00094                  *
00095A 006C                    ORG    $6C      80 CHARACTER INPUT BUFFER. USED FOR ORIGINAL COMMAND INPUT
00096                  *
00097A 006C    0050  A IBUF1  RMB    80       END OF INPUT BUFFER #1
00098A 00BC    0002  A BFPT1  RMB    2        BUFFER #1 SCAN POINTER SAVE LOCATION (TWO BYTES)
00099A 00BE    0002  A BFND1  RMB    2        BUFFER #1 TEXT END MARKER (NOT USED) (TWO BYTES)
00100                  *
00101                  *FILE NAME PROTOTYPE STRING, USED IN STRING SEARCHES & PARSING
00102                  *
00103A 00C0    0001  A FLNM0  RMB    1        INPUT FILE NAME BUFFER CHAR #0
00104A 00C1    0001  A FLNM1  RMB    1        INPUT FILE NAME BUFFER CHAR #1
00105A 00C2    0001  A FLNM2  RMB    1        INPUT FILE NAME BUFFER CHAR #2
00106A 00C3    0001  A FLNM3  RMB    1        INPUT FILE NAME BUFFER CHAR #3
00107A 00C4    0001  A FLNM4  RMB    1        INPUT FILE NAME BUFFER CHAR #4
00108A 00C5    0001  A FLNM5  RMB    1        INPUT FILE NAME BUFFER UNIT ID NO.
```

```
00110A 00C6    0001  A FILAT  RMB    1        SAVE LOCATION, ATTRIBUTES OF CREATED FILE
00111A 00C7    0002  A FILRQ  RMB    2        REQUESTED FILE SIZE, IN SECTORS (TWO BYTES)
00112          00C9  A HDIGT  EQU    *        SAVE LOCATION USED IN HEX TO BINARY CONVERSION
00113          00C9  A UNUMB  EQU    *        UNIT NUMBER SAVE LOCATON
00114          00C9  A NULC1  EQU    *        NULL COUNTER LOCATION
00115A 00C9    0001  A WRTRY  RMB    1        WRITE ERROR RETRY COUNT
00116          00CA  A BGBFP  EQU    *        TRACK BUFFER / LARGE DATA BUFFER POINTER
00117          00CA  A SECAV  EQU    *        CURRENT NUMBER OF SECTORS AVAILABLE
00118          00CA  A SECP1  EQU    *        DISC SECTOR BUFFER INDEX POINTER
00119A 00CA    0001  A BINHI  RMB    1        HIGH BYTE OF PACKED BINARY IN HEX TO BINARY CONVERSION
00120A 00CB    0001  A BINLO  RMB    1        LOW BYTE OF PACKED BINARY IN HEX TO BINARY CONVERSION
00121A 00CC    0002  A BFOP2  RMB    2        BUFFER #2 OUTPUT POINTER SAVE LOCATION (2 BYTES)
00122                *
00123A 00CE    0002  A SESP1  RMB    2        DISK SECTOR BUFFER #1 SCAN POINTER (TWO BYTES)
00124A 00D0    0080  A SECB1  RMB    128      DISK SECTOR BUFFER AREA - 128 BYTES IN A FULL SECTOR
00125                *
00126A 0150    0001  A USECT  RMB    1        SECTOR IN USE I/D BUFFER LOCATION
00127A 0151    0001  A CRTRK  RMB    1        CURRENT TRACK SAVE LOCATION
00128A 0152    0001  A CRSEC  RMB    1        CURRENT SECTOR SAVE LOCATION
00129A 0153    0001  A STRAK  RMB    1        SAVE LOC FOR NEW START TRACK
00130A 0154    0001  A STSEC  RMB    1        SAVE LOC FOR NEW START SECTOR
00131          0155  A SECCT  EQU    *        COUNTER USED IN TRACK/SECTOR CONVERSION
00132          0155  A BYTCT  EQU    *        SECTOR BYTE COUNTER, USED IN DUP
00133A 0155    0001  A TRYCT  RMB    1        DISK READ ERROR RETRY COUNTER
00134A 0156    0001  A TIUNT  RMB    1        SAVE LOC FOR NEW UNIT ID
00135A 0157    0001  A LGOFL  RMB    1        EDOS LOAD-&-GO FLAG
00136A 0158    0001  A EOFFL  RMB    1        END-OF-FILE FLAG
00137          0159  A APARS  EQU    *        ASSEMBLER FILE STRING PARSING POINTER
00138A 0159    0002  A TKBFP  RMB    2        DISK TRACK BUFFER SCAN POINTER
00139A 015B    0001  A AERFL  RMB    1        ASSEMBLER PARSING ERROR FLAG
```

```
00141                  *EDOS START @ JUMP VECTORS



00143A 0020                    ORG    $20

00145A 0020 7E 015C  A EXECS   JMP    EXEC     EDOS EXECUTIVE START VECTOR (GENERAL LOAD & GO VECTOR)
00146A 0023 7E 0692  A CLSFV   JMP    FLCLS    WRITE EOF, UPDATE INDEX, CALL MONITOR VECTOR



00148A 0030                    ORG    $30

00150A 0030 7E EAB0  A PRNVC   JMP    LPTC     PRINT OUTPUT SWITCH VECTOR
00151A 0033 7E F018  A PRICV   JMP    PRINC    CONSOLE CHARACTER OUTPUT VECTOR
00152A 0036 7E EB02  A PTAPV   JMP    RDRIN    PAPER TAPE READER HANDLER VECTOR
```

```
00154A 015C                    ORG    $015C
00155                  *
00156                  *EDOS STARTUP ENTRY POINT - PRINT OUT ID MESSAGE - WAIT FOR COMMAND I/P
00157                  *
00158A 015C 8E 006B  A EXEC   LDS    #EXSTK   INITIALIZE EXECUTIVE STACK
00159A 015F BD 03BD  A        JSR    CRLF     ESTABLISH NEW LINE ON CONSOLE
00160A 0162 F6 02E9  A        LDAB   IDMES    GET CHARACTER COUNT FOR I/D
00161A 0165 CE 02EA  A        LDX    #IDMES+1 SET STRING POINTER FOR I/D
00162A 0168 BD 03B3  A        JSR    PRINS    OUTPUT I/D MESSAGE ON CONSOLE
00163A 016B BD 03BD  A        JSR    CRLF     END I/D WITH <CR><LF>
00164A 016E 0F         WTCMD  SEI              WAIT FOR USER COMMAND (INTERRUPTS DISABLED)
00165A 016F BD E806  A        JSR    INTIO    INIT DISC INTERFACE TO KNOWN STATE
00166A 0172 8E 006B  A        LDS    #EXSTK   REINITIALIZE EDOS STACK SPACE
00167A 0175 BD 0327  A        JSR    USCMD    OUTPUT USER PROMPT CHARACTER, WAIT FOR COMMAND
00168A 0178 BD 0360  A        JSR    PRSFL    PARSE COMMAND & LEAVE IN FLNM0-FLNM4
00169                  *
00170                  *DO STRING COMPARE TO IDENTIFY COMMAND.  COMMAND LIST IS PACKED
00171                  *IN 5-CHARACTER STRINGS, PADDED WITH TRAILING SPACES.  THE SWITCH
00172                  *ADDRESS FOR EACH COMMAND FOLLOWS THE CHARACTER STRING.
00173                  *THE INPUT COMMAND STRING IS FOUND AT FLNM0 TO FLNM4
00174                  *
00175A 017B CE 01BB  A        LDX    #CMDST   POINT TO START OF PROTOTYPE COMMANDS
00176A 017E 8C 0278  A CMDLP  CPX    #CMDLM   IF ALL POSSIBLE COMMANDS EXAUSTED
00177A 0181 26 0B 018E        BNE    TRYCM    THEN
00178A 0183 F6 0278  A ERRM   LDAB   FERRM    GET CHARACTER COUNT FOR MESSAGE
00179A 0186 CE 0279  A        LDX    #FERRM+1 SET STRING POINTER FOR FORMAT ERROR MESSAGE
00180A 0189 BD 03B3  A MSWTC  JSR    PRINS    PRINT MESSAGE ON CONSOLE
00181A 018C 20 E0 016E        BRA    WTCMD    WAIT FOR ANOTHER COMMAND INPUT
00182                  *
00183A 018E A6 00    A TRYCM  LDAA   $0,X     IF FIRST PROTOTYPE CHARACTER EQUALS
00184A 0190 91 C0    A        CMPA   FLNM0    FIRST CHARACTER OF THIS COMMAND
00185A 0192 26 1E 01B2        BNE    SKPCM    AND
00186A 0194 A6 01    A        LDAA   $1,X     SECOND PROTOTYPE CHARACTER EQUALS
00187A 0196 91 C1    A        CMPA   FLNM1    SECOND CHARACTER OF THIS COMMAND
00188A 0198 26 18 01B2        BNE    SKPCM    AND
00189A 019A A6 02    A        LDAA   $2,X     THIRD PROTOTYPE CHARACTER EQUALS
00190A 019C 91 C2    A        CMPA   FLNM2    THIRD CHARACTER OF THIS COMMAND
00191A 019E 26 12 01B2        BNE    SKPCM    AND
00192A 01A0 A6 03    A        LDAA   $3,X     FOURTH PROTOTYPE CHARACTER EQUALS
00193A 01A2 91 C3    A        CMPA   FLNM3    FOURTH CHARACTER OF THIS COMMAND
00194A 01A4 26 0C 01B2        BNE    SKPCM    AND
00195A 01A6 A6 04    A        LDAA   $4,X     FITH PROTOTYPE CHARACTER EQUALS
00196A 01A8 91 C4    A        CMPA   FLNM4    FITH CHARACTER OF THIS COMMAND
00197A 01AA 26 06 01B2        BNE    SKPCM    THEN
00198A 01AC EE 05    A        LDX    $5,X     COMMAND IS IDENTIFIED, SELECT ADDRESS
00199A 01AE AD 00    A        JSR    $0,X     CALL THIS COMMAND SERVICE
00200A 01B0 20 BC 016E        BRA    WTCMD    WAIT FOR NEXT COMMAND
00201                  *
00202A 01B2 08         SKPCM  INX              STEP
00203A 01B3 08                INX              OVER
00204A 01B4 08                INX              FIVE
00205A 01B5 08                INX              PROTOTYPE
00206A 01B6 08                INX              CHARS &
00207A 01B7 08                INX              SERVICE
00208A 01B8 08                INX              ADDRESS.
00209A 01B9 20 C3 017E        BRA    CMDLP    TRY ANOTHER COMMAND
```

```
00211                   *
00212                   *COMMAND PROTOTYPE STRINGS BEGIN HERE.  EACH COMMAND IS PADDED TO
00213                   *5 CHARACTERS WITH ASCII SPACES AS REQUIRED.
00214                   *
00215                   *THE COMMAND SERVICE ADDRESS FOLLOWS EACH ASSOCIATED
00216                   *COMMAND PROTOTYPE STRING.
00217                   *
00218A 01BB    41    A CMDST  FCC    /ASM  /
00219A 01C0    0931  A        FDB    ASM
00220A 01C2    45    A        FCC    /EDIT /
00221A 01C7    0B12  A        FDB    EDIT
00222A 01C9    4C    A        FCC    /LOAD /
00223A 01CE    06D7  A        FDB    LOAD
00224A 01D0    45    A        FCC    /EXBUG/
00225A 01D5    F564  A        FDB    EXBUG
00226A 01D7    50    A        FCC    /PDIR /
00227A 01DC    0863  A        FDB    PDIR
00228A 01DE    50    A        FCC    /PLIST/
00229A 01E3    0708  A        FDB    PLIST
00230A 01E5    49    A        FCC    /INIT /
00231A 01EA    0826  A        FDB    INIT
00232A 01EC    44    A        FCC    /DUP  /
00233A 01F1    0D7F  A        FDB    DUP
00234A 01F3    52    A        FCC    /RENAM/
00235A 01F8    079D  A        FDB    RENAM
00236A 01FA    50    A        FCC    /PURGE/
00237A 01FF    0F45  A        FDB    PURGE
00238A 0201    4D    A        FCC    /MERGE/
00239A 0206    0ECC  A        FDB    MERGE
00240A 0208    43    A        FCC    /CDUMP/
00241A 020D    0CBA  A        FDB    CDUMP
00242A 020F    54    A        FCC    /TDUMP/
00243A 0214    0CBA  A        FDB    TDUMP
00244A 0216    43    A        FCC    /CTAPE/
00245A 021B    0BE3  A        FDB    CTAPE
00246A 021D    50    A        FCC    /PTAPE/
00247A 0222    0BE3  A        FDB    PTAPE
```

```
00249A 0224    54    A        FCC    /TTAPE/
00250A 0229    0BE3  A        FDB    TTAPE
00251A 022B    41    A        FCC    /ATTR /
00252A 0230    073A  A        FDB    ATTR
00253A 0232    43    A        FCC    /CREAT/
00254A 0237    07F3  A        FDB    CREAT
00255A 0239    43    A        FCC    /CXGEN/
00256A 023E    0BEC  A        FDB    CXGEN
00257A 0240    50    A        FCC    /PXGEN/
00258A 0245    0BEC  A        FDB    PXGEN
00259A 0247    54    A        FCC    /TXGEN/
00260A 024C    0BEC  A        FDB    TXGEN
00261A 024E    48    A        FCC    /HOME /
00262A 0253    0846  A        FDB    HOME
00263A 0255    49    A        FCC    /INITX/
00264A 025A    082C  A        FDB    INITX
00265A 025C    43    A        FCC    /CDIR /
00266A 0261    085E  A        FDB    CDIR
00267A 0263    43    A        FCC    /CLIST/
00268A 0268    0703  A        FDB    CLIST
00269A 026A    52    A        FCC    /RLOAD/
00270A 026F    06F5  A        FDB    RLOAD
00271A 0271    52    A        FCC    /RASM /
00272A 0276    0982  A        FDB    RASM
00273                 *
00274          0278  A CMDLM  EQU    *        END OF COMMAND PROTOTYPE STRINGS
```

```
00276                   *
00277                   *
00278                   *MESSAGE STRINGS BEGIN HERE.  EACH MESSAGE STRING BEGINS
00279                   *WITH A SINGLE BYTE GIVING THE CHARACTER COUNT IN BINARY
00280                   *FOLLOWED BY THE ASCII MESSAGE TEXT.
00281                   *
00282A 0278    0C     A FERRM  FCB    $0C
00283A 0279    46     A        FCC    /FORMAT ERROR/
00284A 0285    0C     A NOSFM  FCB    $0C
00285A 0286    4E     A        FCC    /NO SUCH FILE/
00286A 0292    09     A DUNMM  FCB    $09
00287A 0293    44     A        FCC    /DUPL NAME/
00288A 029C    07     A NORMM  FCB    $07
00289A 029D    4E     A        FCC    /NO ROOM/
00290A 02A4    1B     A DIRHD  FCB    $1B
00291A 02A5    4E     A        FCC    /NAME   ATTR TRAK SCTR  SIZE/
00292A 02C0    0E     A DKNRM  FCB    $0E
00293A 02C1    44     A        FCC    /DISK NOT READY/
00294A 02CF    0B     A MEDEM  FCB    $0B
00295A 02D0    4D     A        FCC    /MEDIA ERROR/
00296A 02DB    0D     A NOEXM  FCB    $0D
00297A 02DC    20     A        FCC    / NON EXISTENT/
00298A 02E9    13     A IDMES  FCB    $13
00299A 02EA    4D     A        FCC    /M6800 EDOS VER 2.4/
00300A 02FC    0E     A NOSRM  FCB    $0E
00301A 02FD    4E     A        FCC    /NO SOURCE FILE/
00302A 030B    0C     A DEVEM  FCB    $0C
00303A 030C    44     A        FCC    /DEVICE ERROR/
00304A 0318    0E     A OFLCM  FCB    $0E
00305A 0319    4F     A        FCC    /OFILE CONFLICT/
00306                   *
00307                   *END OF MESSAGE STRINGS
```

```
00309                   *
00310A 0327 BD 03BD  A USCMD  JSR    CRLF     ESTABLISH NEW LINE ON CONSOLE
00311A 032A 86 21    A        LDAA   #'!      PRINT AN ASCII "!"
00312A 032C BD F018  A        JSR    PRINC    USER PROMPT CHARACTER
00313A 032F CE 006C  A        LDX    #IBUF1   SET POINTER TO INPUT BUFFER START
00314A 0332 DF BC    A        STX    BFPT1    INITIALIZE POINTER COPY TO BUFFER START
00315A 0334 BD 03C1  A USCM1  JSR    ECHOC    GET A CONSOLE INPUT CHARACTER
00316A 0337 81 7F    A        CMPA   #$7F     IF NOT AN ASCII RUBOUT OR "BREAK"
00317A 0339 27 15 0350        BEQ    USCM3    AND
00318A 033B 4D                TSTA            NOT AN ASCII NULL
00319A 033C 27 1F 035D        BEQ    USCM4    THEN
00320A 033E A7 00    A        STAA   $0,X     STORE THE CHARACTER
00321A 0340 08                INX             STEP ON TO THE NEXT BUFFER LOCATION
00322A 0341 8C 00BC  A        CPX    #IBUF1+80 IF BUFFER OVERFLOW
00323A 0344 26 03 0349        BNE    USCM2    THEN
00324A 0346 7E 0183  A        JMP    ERRM     TELL USER FORMAT ERROR, WAIT FOR NEW COMMAND
00325A 0349 81 0D    A USCM2  CMPA   #CR      IF A <CR> CHARACTER
00326A 034B 26 E7 0334        BNE    USCM1    THEN
00327A 034D DF BE    A        STX    BFND1    COMMAND ENDED, COPY END POINT (NOT USED!)
00328A 034F 39                RTS             RETURN THE COMPLETE COMMAND STRING
00329                   *
00330                   *THIS ROUTINE DELETES CHARACTERS FROM THE COMMAND INPUT BUFFER
00331                   *ECHOING THE CHARACTERS AS THEY ARE DELETED FROM THE BUFFER
00332                   *
00333A 0350 8C 006C  A USCM3  CPX    #IBUF1   RUBOUT FOUND, IF BUFFER EMPTY
00334A 0353 27 D2 0327        BEQ    USCMD    ISSUE NEW PROMPT, START AGAIN
00335A 0355 09                DEX             ELSE BACK UP POINTER
00336A 0356 A6 00    A        LDAA   $0,X     GET THROWN-OUT CHARACTER
00337A 0358 BD F018  A        JSR    PRINC    AND PRINT IT
00338A 035B 20 D7 0334        BRA    USCM1    GET NEXT INPUT CHARACTER
00339                   *
00340A 035D 7E 016E  A USCM4  JMP    WTCMD    WAIT FOR NEXT USER COMMAND INPUT
00341                   *
00342                   *
00343                   *THIS ROUTINE PARSES A SINGLE COMMAND OR FILE NAME AND
00344                   *LEAVES IT IN FLNM0-4.  IF A UID WAS SPECIFIED, IT IS LEFT IN FLNM5.
00345                   *
00346A 0360 CE 00C0  A PRSFL  LDX    #FLNM0   FILL FLNM0-4 WITH
00347A 0363 86 20    A        LDAA   #SPC     ASCII SPACE CHARACTERS
00348A 0365 C6 05    A        LDAB   #$5      AS A DEFAULT VALUE
00349A 0367 A7 00    A PRSF1  STAA   $0,X     STORE A SPACE CHARACTER
00350A 0369 08                INX             STEP ON TO THE NEXT POSITION
00351A 036A 5A                DECB            WHILE FLNM0 TO FLNM4
00352A 036B 26 FA 0367        BNE    PRSF1    CONTINUE
00353A 036D 7F 00C5  A        CLR    FLNM5    SET DEFAULT UNIT ID = 0 (SYS DISK)
00354A 0370 CE 00C0  A        LDX    #FLNM0   START TO TRANSFER IN A COMMAND OR FILE NAME
00355A 0373 BD 0398  A PRSF2  JSR    NXITM    GET A CHARACTER FROM THE STRING
00356A 0376 81 3A    A        CMPA   #':      IF IT IS ASCII ":"
00357A 0378 26 14 038E        BNE    NOUID    THEN
00358A 037A BD 0398  A        JSR    NXITM    GET THE NEXT CHARACTER
00359A 037D 80 30    A        SUBA   #'0      STRIP DECIMAL ASCII TO BINARY
00360A 037F 16                TAB             COPY THE RESULT IN B
00361A 0380 84 FC    A        ANDA   #$FC     IF INTEGER RESULT IS .LE. 3
00362A 0382 26 07 038B        BNE    UIDER    THEN
00363A 0384 D7 C5    A        STAB   FLNM5    ACCEPT & STORE UNIT ID
00364A 0386 BD 0398  A PRSF3  JSR    NXITM    GET DELIMITER (COMMA)
00365A 0389 20 FB 0386        BRA    PRSF3    OR TERMINATOR <CR>
00366                   *
```

```
00367A 038B 7E 0183  A UIDER  JMP    ERRM     TELL USER FORMAT ERROR, TRY AGAIN
00368                *
00369A 038E 8C 00C5  A NOUID  CPX    #FLNM5   IF ROOM IN FIELD
00370A 0391 27 E0 0373        BEQ    PRSF2    THEN
00371A 0393 A7 00    A        STAA   $0,X     STORE THE CHARACTER
00372A 0395 08                INX             STEP OVER THE LOCATION
00373A 0396 20 DB 0373        BRA    PRSF2    GET THE NEXT CHARACTER
00374                *
00375                *THIS ROUTINE PASSES SINGLE CHARACTERS BACK TO PRSFL.  WHEN A DELIMITER
00376                *(COMMA) OR TERMINATOR <CR> IS FOUND, PRSFL IS TERMINATED (ERSATZ RTS)
00377                *
00378A 0398 FF 03B1  A NXITM  STX    ITMDX    SAVE CURRENT INDEX REGISTER
00379A 039B DE BC    A        LDX    BFPT1    PICK UP SCAN POINTER
00380A 039D A6 00    A        LDAA   $0,X     PICK UP A CHARACTER
00381A 039F 81 0D    A        CMPA   #CR      IF NOT AN ASCII <CR>
00382A 03A1 27 0B 03AE        BEQ    NXIT1    THEN
00383A 03A3 08                INX             STEP POINTER TO NEXT CHARACTER
00384A 03A4 DF BC    A        STX    BFPT1    STORE THE SCAN POINTER
00385A 03A6 81 2C    A        CMPA   #',      IF NOT A COMMA
00386A 03A8 27 04 03AE        BEQ    NXIT1    THEN
00387A 03AA FE 03B1  A        LDX    ITMDX    RESTORE INDEX REGISTER
00388A 03AD 39                RTS             RETURN THE CHARACTER
00389                *
00390A 03AE 31         NXIT1  INS             IF CHARACTER COMMA OR <CR>
00391A 03AF 31                INS             DELETE RETURN ADDRESS
00392A 03B0 39                RTS             TERMINATE & RETURN FROM PRSFL
00393                *
00394A 03B1    0002  A ITMDX  RMB    2        NXITM INDEX SAVE LOCATION
00395                *
00396                *
00397                *THIS ROUTINE PRINTS OUT A STRING OF ASCII CHARACTERS
00398                *START ADDRESS IS IN X, NUMBER OF CHARACTERS IN B
00399                *
00400A 03B3 A6 00    A PRINS  LDAA   $0,X     GET A CHARACTER
00401A 03B5 BD F018  A        JSR    PRINC    PRINT IT OUT
00402A 03B8 08                INX             STEP TO NEXT CHARACTER
00403A 03B9 5A                DECB            WHILE CHARACTERS
00404A 03BA 26 F7 03B3        BNE    PRINS    CONTINUE
00405A 03BC 39                RTS             RETURN TO CALLER
00406                *
00407                *
00408                *THIS ROUTINE PRODUCES <CR> <LF> ON THE CONSOLE
00409                *(ESTABLISH A NEW LINE FOR THE NEXT FUNCTION)
00410                *
00411A 03BD 86 0D    A CRLF   LDAA   #CR      DEFINE AN ASCII <CR>
00412A 03BF 20 09 03CA        BRA    ECHO1    BORROW THE ECHO FUNCTION
00413                *
00414                *THIS ROUTINE ECHOS INPUT CHARACTERS, EXCEPT RUBOUT OR "BREAK"
00415                *THE PARITY BIT IS STRIPPED, AND <CR> IS AMPLIFIED TO <CR> <LF>
00416                *THE INPUT CHARACTER IS PASSED THROUGH THIS ROUTINE
00417                *
00418A 03C1 BD 03DB  A ECHOC  JSR    GETCH    GET AN INPUT CHARACTER
00419A 03C4 84 7F    A        ANDA   #$7F     STRIP OFF THE PARITY BIT
00420A 03C6 81 7F    A        CMPA   #$7F     IF NOT A RUBOUT OR "BREAK"
00421A 03C8 27 10 03DA        BEQ    ECHO2    THEN
00422A 03CA 36         ECHO1  PSHA            SAVE THE CHARACTER
00423A 03CB BD F018  A        JSR    PRINC    PRINT THE CHARACTER
00424A 03CE 32                PULA            RESTORE THE CHARACTER
```

```
00425A 03CF 81 0D    A        CMPA   #CR      IF IT IS ASCII <CR>
00426A 03D1 26 07 03DA        BNE    ECHO2    THEN
00427A 03D3 36                PSHA            SAVE THE CHARACTER AGAIN
00428A 03D4 86 0A    A        LDAA   #LF      DEFINE AN ASCII <LF>
00429A 03D6 BD F018  A        JSR    PRINC    AND DO A CONSOLE LINE FEED
00430A 03D9 32                PULA            RESTORE THE CHARACTER
00431A 03DA 39         ECHO2  RTS             RETURN TO CALLER
00432                  *
00433                  *
00434                  *THIS ROUTINE LOOPS ON THE ACIA STATUS REGISTER DONE BIT WAITING FOR
00435                  *A CHARACTER.  THE CPU WAITS HERE UNTIL A CHARACTER IS READY.  WHEN
00436                  *A CHARACTER IS PRESENTED, IT IS RETURNED TO CALLER IN A.
00437                  *
00438A 03DB B6 FCF4  A GETCH  LDAA   ACIAS    READ THE ACIA STATUS
00439A 03DE 47                ASRA            IF CHARACTER IS AVAILABLE
00440A 03DF 24 FA 03DB        BCC    GETCH    THEN
00441A 03E1 B6 FCF5  A        LDAA   ACIAR    READ THE CHARACTER
00442A 03E4 0C                CLC             CLEAR THE "C" BIT
00443A 03E5 39                RTS             RETURN THE CHARACTER
00444                  *
00445                  *
00446                  *THIS ROUTINE PERFORMS BINARY TO HEX CONVERSION,
00447                  *AND PRINTS THE RESULT USING THE SWITCHED PRINT
00448                  *VECTOR TO SELECT THE OUTPUT DEVICE.
00449                  *
00450A 03E6 97 C9    A PRBIN  STAA   HDIGT    COPY THE BINARY DATA
00451A 03E8 44                LSRA            SHIFT THE 4
00452A 03E9 44                LSRA            MOST SIGNIFICANT
00453A 03EA 44                LSRA            BITS INTO A
00454A 03EB 44                LSRA            BINARY VALUE
00455A 03EC BD 03F6  A        JSR    PRBN1    PRINT THE VALUE(H)
00456A 03EF 96 C9    A        LDAA   HDIGT    GET THE BINARY DATA
00457A 03F1 84 0F    A        ANDA   #$0F     STRIP OFF THE HIGH DIGIT
00458A 03F3 7E 03F6  A        JMP    PRBN1    PRINT THE VALUE(L) & RETURN
00459                  *
00460A 03F6 81 0A    A PRBN1  CMPA   #$0A     IF .GE. 9
00461A 03F8 2B 02 03FC        BMI    PRBN2    THEN
00462A 03FA 8B 07    A        ADDA   #$7      SKIP OVER : TO @
00463A 03FC 8B 30    A PRBN2  ADDA   #'0      CONVERT TO ASCII (HEX)
00464A 03FE 7E 0030  A        JMP    PRNVC    PRINT, USING SWITCHED PRINT & RETURN
00465                  *
00466                  *
00467                  *THIS ROUTINE READS A FULL DISK SECTOR FROM THE DEFINED
00468                  *UNIT/TRACK/SECTOR, LEAVING THE RESULT BEGINNING AT THE
00469                  *ADDRESS IN X.  SUCCESS RETURN IS A = 0
00470                  *
00471A 0401 DF CA    A RED11  STX    SECP1    SAVE INDEX REGISTER
00472A 0403 37                PSHB            SAVE DEFINED TRACK
00473A 0404 BD EA03  A        JSR    XUS      TRANSMIT <UNIT><SECTOR> TO DISK
00474A 0407 BD 0468  A        JSR    DSKRD    IF DISK IS READY, THEN
00475A 040A 32                PULA            GET DEFINED TRACK
00476A 040B BD EA2B  A        JSR    SEEK     SEEK SELECTED UNIT TO DEFINED TRACK
00477A 040E 86 05    A        LDAA   #$5      SET RETRY COUNT FOR 5
00478A 0410 B7 0155  A        STAA   TRYCT    ATTEMPTED RETRIES ON READ ERROR
00479A 0413 86 02    A RED12  LDAA   #$2      DEFINE READ COMMAND
00480A 0415 BD EA3E  A        JSR    LOOP     ISSUE COMMAND TO DISK (LOOP ON BUSY)
00481A 0418 B6 EC00  A        LDAA   DKDID    READ THE DISK STATUS
00482A 041B 84 80    A        ANDA   #$80     IF DELETED DATA (DD), REPORT IT
```

```
00483A 041D 26 18 0437        BNE    RED14    ELSE
00484A 041F B6 EC00  A        LDAA   DKDID    READ THE DISK STATUS
00485A 0422 84 08    A        ANDA   #$08     IF CRC ERROR
00486A 0424 27 17 043D        BEQ    RED15    THEN
00487A 0426 BD EA38  A        JSR    RFLAG    RESET ERROR FLAG ON DISK I/F
00488A 0429 7A 0155  A        DEC    TRYCT    DECREMENT TRY COUNTER
00489A 042C 26 E5 0413        BNE    RED12    THEN
00490A 042E F6 02CF  A MEDER  LDAB   MEDEM    GET CHARACTER COUNT AND POINTER
00491A 0431 CE 02D0  A        LDX    #MEDEM+1 FOR "MEDIA ERROR" MESSAGE
00492A 0434 7E 0189  A        JMP    MSWTC    PRINT IT & WAIT FOR COMMAND INPUT
00493                 *
00494A 0437 BD EA38  A RED14  JSR    RFLAG    RESET ERROR FLAG ON DISK I/F
00495A 043A 86 80    A        LDAA   #$80     DEFINE OPERATION RESULT NON-ZERO, MINUS
00496A 043C 39                RTS             RETURN TO CALLER (FAILED)
00497                 *
00498A 043D DE CA    A RED15  LDX    SECP1    PICK UP THE INDEX POINTER
00499A 043F 86 80    A        LDAA   #$80     DEFINE COUNT OF BYTES IN A FULL SECTOR
00500A 0441 B7 0155  A        STAA   BYTCT    FOR A COUNT (128 BYTES)
00501A 0444 86 3C    A RED16  LDAA   #$3C     DEFINE DISK COMMAND MODE
00502A 0446 B7 EC03  A        STAA   DKCOC    SET DISK TO COMMAND MODE
00503A 0449 86 40    A        LDAA   #$40     DEFINE READ DATA COMMAND
00504A 044B B7 EC02  A        STAA   DKCOD    COMMAND DISK TO READ
00505A 044E B6 EC00  A        LDAA   DKDID    READ THE DATA FROM THE IDISK
00506A 0451 A7 00    A        STAA   $00,X    STORE AWAY READ BYTE
00507A 0453 86 2C    A        LDAA   #$2C     DEFINE RESET COMMAND MODE
00508A 0455 B7 EC03  A        STAA   DKCOC    RESET DISK FROM COMMAND MODE
00509A 0458 86 40    A        LDAA   #$40     DEFINE READ DATA COMMAND
00510A 045A B7 EC02  A        STAA   DKCOD    STROBE IN RESET COMMAND MODE
00511A 045D 7F EC02  A        CLR    DKCOD    CLEAR COMMAND OUT OF PIA
00512A 0460 08                INX             TAKE THE NEXT BYTE
00513A 0461 7A 0155  A        DEC    BYTCT    WHILE BYTES
00514A 0464 26 DE 0444        BNE    RED16    CONTINUE
00515A 0466 4F                CLRA            DEFINE OPERATION RESULT ZERO (SUCCESS)
00516A 0467 39                RTS             RETURN TO CALLER (SUCCESS)
00517                 *
00518                 *
00519                 *THIS ROUTINE CHECKS ON DISK READY FOR USE.  IF READY,
00520                 *RETURNS TO CALLER, IF NOT READY,  ABORTS OPERATION
00521                 *PRINTS "DISK NOT READY" & WAITS FOR COMMAND
00522                 *
00523A 0468 B6 EC00  A DSKRD  LDAA   DKDID    READ DISK STATUS REGISTER
00524A 046B 84 20    A        ANDA   #$20     IF SELECTED DISK READY
00525A 046D 26 01 0470        BNE    DSKR1    THEN
00526A 046F 39                RTS             RETURN, ELSE
00527A 0470 F6 02C0  A DSKR1  LDAB   DKNRM    GET CHARACTER COUNT AND POINTER
00528A 0473 CE 02C1  A        LDX    #DKNRM+1 FOR "DISK NOT READY" MESSAGE
00529A 0476 7E 0189  A        JMP    MSWTC    PRINT IT, AND WAIT FOR COMMAND INPUT
00530                 *
00531                 *
00532                 *THIS ROUTINE WRITES DATA TO DISK.  DATA IS POINTED TO
00533                 *BY X REGISTER. A FULL SECTOR OF 128 BYTES IS ASSUMED AND
00534                 *WRITTEN.  A WRITE FAILURE RETURNS A = 1, SUCCESS - A = 0.
00535                 *
00536A 0479 DF CA    A WRTSE  STX    SECP1    SAVE THE DATA POINTER
00537A 047B 37                PSHB            SAVE THE SELECTED TRACK
00538A 047C BD EA03  A        JSR    XUS      TRANSMIT <UNIT><SECTOR> TO DISK
00539A 047F BD 0468  A        JSR    DSKRD    IF DISK IS READY, THEN
00540A 0482 32                PULA            GET THE SELECTED TRACK
```

```
00541A 0483 BD EA2B  A        JSR    SEEK     SEEK TO THE SELECTED TRACK
00542A 0486 86 05    A        LDAA   #$5      SET TRY COUNT FOR 5
00543A 0488 B7 0155  A        STAA   TRYCT    ATTEMPTED RETRIES ON READ ERROR
00544A 048B DE CA    A        LDX    SECP1    PICK UP THE DATA POINTER
00545A 048D C6 80    A        LDAB   #128     DEFINE A SECTOR = 128 DECIMAL BYTES
00546                 *
00547A 048F A6 00    A WRTS1  LDAA   0,X      PICK UP A DATA BYTE
00548A 0491 B7 EC06  A        STAA   DKDOD    LOAD DATA TO BUFFER
00549A 0494 86 30    A        LDAA   #$30     DEFINE LOAD COMMAND
00550A 0496 B7 EC02  A        STAA   DKCOD    COMMAND BUFFER TO LOAD
00551A 0499 08                INX            STEP TO THE NEXT DATA BYTE
00552A 049A 5A                DECB           WHILE SECTOR BYTES
00553A 049B 26 F2 048F        BNE    WRTS1    CONTINUE
00554                 *
00555A 049D 86 04    A WRTS2  LDAA   #$04     DEFINE WRITE COMMAND
00556A 049F BD EA3E  A        JSR    LOOP     ISSUE COMMAND TO DISK (LOOP ON BUSY)
00557A 04A2 86 06    A        LDAA   #$06     ASK DISK FOR STATUS
00558A 04A4 BD EA3E  A        JSR    LOOP     ISSUE COMMAND TO DISK (LOOP ON BUSY)
00559A 04A7 B6 EC00  A        LDAA   DKDID    READ THE REQUESTED STATUS
00560A 04AA 84 08    A        ANDA   #$08     IF THERE WAS A CRC ERROR
00561A 04AC 27 0F 04BD        BEQ    WRTS3    THEN (SUCCESS EXIT, A = 0)
00562A 04AE BD EA38  A        JSR    RFLAG    RESET ERROR FLAG ON DISK I/F
00563A 04B1 7A 0155  A        DEC    TRYCT    IF RETRY COUNT RUN OUT
00564A 04B4 26 E7 049D        BNE    WRTS2    THEN
00565A 04B6 86 0E    A        LDAA   #$0E     ASK DISK TO MARK THIS SECTOR = DD
00566A 04B8 BD EA3E  A        JSR    LOOP     ISSUE COMMAND TO DISK (LOOP ON BUSY)
00567A 04BB 86 01    A        LDAA   #1       FAILURE EXIT, SET A .NE. 0
00568A 04BD 39         WRTS3  RTS            RETURN TO CALLER
00569                 *
00570                 *
00571                 *THIS ROUTINE SEARCHES THE DEFINED DISK DIRECTORY FOR A FILE NAME.
00572                 *IF FOUND, X POINTS TO THE FCB OF THAT FILE IN THE BUFFER, AND A = 0.
00573                 *
00574A 04BE 0C         FNDFL  CLC            CLEAR THE C BIT FOR SHIFT
00575A 04BF 96 C5    A        LDAA   FLNM5    PICK UP CURRENT UNIT ID
00576A 04C1 46                RORA           SHIFT UID BITS 1,0
00577A 04C2 46                RORA           INTO BITS 7,6 OF RESULT
00578A 04C3 46                RORA           BIT 7 BECOMES OLD 1, 6 BECOMES OLD 0
00579A 04C4 8A 04    A        ORAA   #4       START AT SECTOR 4
00580A 04C6 B7 0150  A        STAA   USECT    INITIALIZE SEARCH SECTOR VALUE
00581A 04C9 B6 0150  A FNDF1  LDAA   USECT    PICK UP SEARCH SECTOR VALUE
00582A 04CC 84 3F    A        ANDA   #$3F     STRIP OFF UID
00583A 04CE 81 1B    A        CMPA   #$1B     IF NOT RUN OUT OF SECTORS
00584A 04D0 27 42 0514        BEQ    FNDFX    THEN (FAILURE EXIT A .NE. 0)
00585A 04D2 CE 00D0  A        LDX    #SECB1   DEFINE DISK SECTOR BUFFER #1
00586A 04D5 B6 0150  A        LDAA   USECT    PICK UP SEARCH SECTOR VALUE
00587A 04D8 5F                CLRB           ON TRACK 0
00588A 04D9 BD 0401  A        JSR    RED11    READ A DISK SECTOR INTO BUFFER
00589A 04DC 4D                TSTA           IF NOT SUCCESSFUL
00590A 04DD 27 05 04E4        BEQ    FNDF2    THEN
00591A 04DF 7C 0150  A        INC    USECT    TRY THE NEXT SECTOR
00592A 04E2 20 E5 04C9        BRA    FNDF1    WHILE SECTORS LAST
00593A 04E4 CE 00D0  A FNDF2  LDX    #SECB1   START AT BEGINNING OF BUFFER
00594A 04E7 DF CE    A        STX    SESP1    INITIALIZE SCAN POINTER
00595A 04E9 DE CE    A FNDF3  LDX    SESP1    PICK UP SCAN POINTER
00596A 04EB A6 05    A        LDAA   $5,X     PICK UP THIS FILE'S ATTRIBUTES
00597A 04ED 81 FF    A        CMPA   #$FF     IF THIS IS NOT THE END OF DIRECTORY
00598A 04EF 27 23 0514        BEQ    FNDFX    AND (FAILURE EXIT A .NE. 0)
```

```
00599A 04F1 81 80    A        CMPA   #$80     IF THIS IS NOT A DELETED FILE
00600A 04F3 27 20 0515        BEQ    FNDF4    AND
00601A 04F5 A6 00    A        LDAA   $0,X     THE FIRST CHARACTER EQUALS
00602A 04F7 91 C0    A        CMPA   FLNM0    THE FIRST SEARCH CHARACTER
00603A 04F9 26 1A 0515        BNE    FNDF4    AND
00604A 04FB A6 01    A        LDAA   $1,X     THE SECOND CHARACTER EQUALS
00605A 04FD 91 C1    A        CMPA   FLNM1    THE SECOND SEARCH CHARACTER
00606A 04FF 26 14 0515        BNE    FNDF4    AND
00607A 0501 A6 02    A        LDAA   $2,X     THE THIRD CHARACTER EQUALS
00608A 0503 91 C2    A        CMPA   FLNM2    THE THIRD SEARCH CHARACTER
00609A 0505 26 0E 0515        BNE    FNDF4    AND
00610A 0507 A6 03    A        LDAA   $3,X     THE FOURTH CHARACTER EQUALS
00611A 0509 91 C3    A        CMPA   FLNM3    THE FOURTH SEARCH CHARACTER
00612A 050B 26 08 0515        BNE    FNDF4    AND
00613A 050D A6 04    A        LDAA   $4,X     THE FIFTH CHARACTER EQUALS
00614A 050F 91 C4    A        CMPA   FLNM4    THE FIFTH SEARCH CHARACTER
00615A 0511 26 02 0515        BNE    FNDF4    THEN
00616A 0513 4F                CLRA            FILE NAME IS FOUND (SUCCESS EXIT A = 0)
00617A 0514 39          FNDFX RTS             RETURN TO CALLER
00618                   *
00619A 0515 86 0B    A FNDF4  LDAA   #11      DEFINE DIRECTORY ENTRY SIZE (11 BYTES / ENTRY)
00620A 0517 5F                CLRB            ALLOWING A HIGH BYTE OF 0
00621A 0518 9B CF    A        ADDA   SESP1+LBYT ADD ENTRY SIZE TO CURRENT
00622A 051A D9 CE    A        ADCB   SESP1    BUFFER SCAN POINTER (2 BYTE ADD)
00623A 051C 97 CF    A        STAA   SESP1+LBYT UPDATE THE BUFFER SCAN POINTER
00624A 051E D7 CE    A        STAB   SESP1    MODULO <ENTRY SIZE>
00625A 0520 DE CE    A        LDX    SESP1    IF THE BUFFER HAS
00626A 0522 8C 0149  A        CPX    #SECB1+121 NO MORE DIRECTORY ENTRIES
00627A 0525 26 C2 04E9        BNE    FNDF3    THEN
00628A 0527 7C 0150  A        INC    USECT    TRY THE NEXT SECTOR
00629A 052A 20 9D 04C9        BRA    FNDF1    AND KEEP LOOKING
00630                   *
00631                   *
00632                   *THIS ROUTINE SETS UP AN INPUT FILE ON THE FILE DEFINITION LOCATIONS.
00633                   *IF NO FILE, FAILURE IS REPORTED.  SUCCESS - A = 0, FAILURE - A .NE. 0
00634                   *
00635A 052C BD 04BE  A DFINF  JSR    FNDFL    LOOK FOR THIS FILE NAME IN THE DIRECTORY
00636A 052F 4D                TSTA            IF IT DOES NOT EXIST
00637A 0530 27 01 0533        BEQ    DFIN1    THEN
00638A 0532 39                RTS             FAILURE EXIT A .NE. 0
00639A 0533 DE CE    A DFIN1  LDX    SESP1    GET FILE CONTROL BLOCK ADDRESS
00640A 0535 E6 06    A        LDAB   $6,X     USE DESCRIPTOR "TRACK"
00641A 0537 D7 06    A        STAB   ITRK     TO DEFINE ITRK
00642A 0539 E6 08    A        LDAB   $8,X     USE DESCRIPTOR SIZE(H)
00643A 053B D7 04    A        STAB   ISIZE    TO DEFINE ISIZE(H)
00644A 053D E6 09    A        LDAB   $9,X     USE DESCRIPTOR SIZE(L)
00645A 053F D7 05    A        STAB   ISIZE+1  TO DEFINE ISIZE(L)
00646A 0541 7F 0008  A        CLR    ICNTR    CLEAR INPUT FILE BYTE COUNTER
00647A 0544 D6 C5    A        LDAB   FLNM5    PICK UP CURRENT UNIT ID
00648A 0546 0C                CLC             CLEAR "C" FOR SHIFTING
00649A 0547 56                RORB            SHIFT UID BITS 1,0
00650A 0548 56                RORB            INTO BITS 7,6
00651A 0549 56                RORB            7 BECOMES OLD 1, 6 BECOMES OLD 0
00652A 054A EA 07    A        ORAB   $7,X     OR IN SECTOR NUMBER FROM DESCRIPTOR
00653A 054C 5A                DECB            MINUS ONE
00654A 054D D7 07    A        STAB   ISCTR    STORE IN ISCTR
00655A 054F 0C                CLC             CLEAR "C" FOR REVERSE SHIFT
00656A 0550 59                ROLB            SHIFT UID BITS BACK INTO BITS 1,0
```

```
00657A 0551 59                ROLB            SECTOR NUMBER GOES INTO
00658A 0552 59                ROLB            BITS 6-2 OF RESULT
00659A 0553 D7 03    A        STAB   IUNIT    STORE COMBINED RESULT IN IUNIT
00660A 0555 4F                CLRA            CLEAR A, SIGNALING SUCCESS
00661A 0556 39                RTS             RETURN TO CALLER
00662                 *
00663                 *
00664                 *
00665                 *
00666                 *THIS ROUTINE IS SIMILAR TO DFINF.  IF CALLED, IT WOULD DEFINE AN
00667                 *OUTPUT FILE IN THE OUTPUT FILE DEFINITION LOCATIONS.  IT DOES NOT
00668                 *APPEAR TO BE USED.  AN EXISTING FILE WILL BE PRECISELY OVERWRITTEN
00669                 *IF IT IS CALLED, WITH NO CHANGE IN FILE SIZE.
00670                 *
00671A 0557 BD 04BE  A DFOFL  JSR    FNDFL    LOOK FOR THIS FILE NAME IN THE DIRECTORY
00672A 055A 4D                TSTA            IF IT DOES NOT EXIST
00673A 055B 27 01 055E        BEQ    DFOF1    THEN
00674A 055D 39                RTS             FAILURE EXIT A .NE. 0
00675A 055E DE CE    A DFOF1  LDX    SESP1    GET FILE CONTROL BLOCK ADDRESS
00676A 0560 E6 06    A        LDAB   $6,X     USE DESCRIPTOR "TRACK"
00677A 0562 D7 0B    A        STAB   OTRK     TO DEFINE OTRK
00678A 0564 E6 08    A        LDAB   $8,X     USE DESCRIPTOR "SIZE"
00679A 0566 A6 09    A        LDAA   $9,X     (2 BYTES)
00680A 0568 80 01    A        SUBA   #$1      MINUS 1 SECTOR
00681A 056A C2 00    A        SBCB   #$0      (2 BYTES)
00682A 056C D7 09    A        STAB   OSIZE    TO DEFINE OSIZE
00683A 056E 97 0A    A        STAA   OSIZE+LBYT (2 BYTES)
00684A 0570 7F 000D  A        CLR    OCNTR    CLEAR OUTPUT FILE BYTE COUNTER
00685A 0573 D6 C5    A        LDAB   FLNM5    PICK UP UNIT I/D
00686A 0575 D7 02    A        STAB   OUNIT    AND DEFINE OUNIT
00687A 0577 0C                CLC             CLEAR "C" FOR SHIFTING
00688A 0578 56                RORB            SHIFT UID BITS 1,0
00689A 0579 56                RORB            INTO BITS 7,6
00690A 057A 56                RORB            7 BECOMES OLD 1, 6 BECOMES OLD 0
00691A 057B EA 07    A        ORAB   $7,X     OR IN SECTOR NUMBER FROM DESCRIPTOR
00692A 057D D7 0C    A        STAB   OSCTR    WHICH DEFINES OSCTR
00693A 057F 4F                CLRA            CLEAR A, SIGNALLING SUCCESS
00694A 0580 39                RTS             RETURN TO CALLER
00695                 *
00696                 *
00697                 *
00698                 *THIS ROUTINE ATTEMPTS TO CREATE A FILE ENTRY IN THE DIRECTORY
00699                 *VARIOUS RETURN FLAGS EXIST. THEY ARE ASSIGNED AS FOLLOWS:
00700                 *A = 0 SUCCESS, FILE CREATED IN DIRECTORY
00701                 *A = 1 FAILURE, NO ROOM LEFT IN DIRECTORY
00702                 *A = 2 FAILURE, THIS FILE NAME ALREADY EXISTS
00703                 *A = 3 FAILURE, NOT ENOUGH ROOM ON THE DISK
00704                 *
00705A 0581 BD 04BE  A CREAF  JSR    FNDFL    SEARCH FOR FILE NAME (DUPLICATE) IN DIRECTORY
00706A 0584 4D                TSTA            IF DUPLICATE FILE NAME IS FOUND
00707A 0585 26 03 058A        BNE    CREA1    THEN
00708A 0587 86 02    A        LDAA   #$2      DEFINE FLAG VALUE = DUPLICATE FILE
00709A 0589 39                RTS             AND RETURN (FAILURE)
00710A 058A 81 FF    A CREA1  CMPA   #$FF     IF DIRECTORY IS FULL (NO EOD MARK)
00711A 058C 27 03 0591        BEQ    CREA2    THEN
00712A 058E 86 01    A        LDAA   #$1      DEFINE FLAG VALUE = NO ROOM
00713A 0590 39                RTS             AND RETURN (FAILURE)
00714A 0591 DE CE    A CREA2  LDX    SESP1    LOAD SCAN POINTER
```

```
00715A 0593 86 1A    A        LDAA   #26      THERE ARE 26 SECTORS PER TRACK
00716A 0595 B7 0155  A        STAA   SECCT    PREPARE TO COUNT BY <SECTOR><TRACK>
00717A 0598 4F                CLRA           STARTING FROM
00718A 0599 5F                CLRB           0, COUNT OFF SECTOR MODULUS
00719A 059A AB 06    A CREA3  ADDA   $6,X     TIMES THE TRACKS
00720A 059C C9 00    A        ADCB   #$0      ALREADY USED
00721A 059E 7A 0155  A        DEC    SECCT    GIVING THE NO. OF SECTORS
00722A 05A1 26 F7 059A        BNE    CREA3    ALREADY USED (TO THIS TRACK)
00723A 05A3 AB 07    A        ADDA   $7,X     THEN ADD THE SECTORS ALREADY USED ON
00724A 05A5 C9 00    A        ADCB   #$0      THIS TRACK, GIVING SECTORS USED (TOTAL)
00725A 05A7 B7 0156  A        STAA   SECCT+1  PUT AWAY THE CALCULATED
00726A 05AA F7 0155  A        STAB   SECCT    NUMBER OF SECTORS ALREADY USED
00727A 05AD CE 07D0  A        LDX    #2000    THERE ARE 2000 SECTORS AVAILABLE ON THE DISK
00728A 05B0 DF CA    A        STX    SECAV    CALCULATE
00729A 05B2 D6 CA    A        LDAB   SECAV    THAT THE
00730A 05B4 96 CB    A        LDAA   SECAV+1  AVAILABLE SECTORS
00731A 05B6 B0 0156  A        SUBA   SECCT+1  MINUS THE USED
00732A 05B9 F2 0155  A        SBCB   SECCT    SECTORS GIVES THE FREE SECTORS
00733A 05BC D7 CA    A        STAB   SECAV    SAVE THE NEW
00734A 05BE 97 CB    A        STAA   SECAV+1  FREE SECTORS
00735A 05C0 90 C8    A        SUBA   FILRQ+1  SUBTRACT THE NEW FILE'S SECTOR REQUIREMENT
00736A 05C2 D2 C7    A        SBCB   FILRQ    IF NOT ENOUGH SECTORS ARE FREE
00737A 05C4 2A 05 05CB        BPL    CREA4    THEN
00738A 05C6 DE CA    A        LDX    SECAV    DEFINE HOW MANY SECTORS ARE AVAILABLE
00739A 05C8 86 03    A        LDAA   #$3      FLAG "NOT ENOUGH ROOM ON DISK"
00740A 05CA 39                RTS            RETURN TO CALLER (FAILURE)
00741                 *
00742                 *FILE IS PERMISSIBLE, CREATE THE DIRECTORY ENTRY
00743                 *
00744A 05CB DE CE    A CREA4  LDX    SESP1    PICK UP THE POINTER TO CURRENT E.O.D. FCB
00745A 05CD 96 C0    A        LDAA   FLNM0    CONVERT THE CURRENT END-OF-DIRECTORY
00746A 05CF A7 00    A        STAA   $0,X     FILE CONTROL BLOCK INTO A DIRECTORY
00747A 05D1 96 C1    A        LDAA   FLNM1    ENTRY FOR THE REQUESTED FILE
00748A 05D3 A7 01    A        STAA   $1,X     BY TRANSFERRING THE REQUESTED
00749A 05D5 96 C2    A        LDAA   FLNM2    FILE NAME INTO THE CURRENT FCB
00750A 05D7 A7 02    A        STAA   $2,X     AS IT RESIDES IN THE SECTOR BUFFER.
00751A 05D9 96 C3    A        LDAA   FLNM3    (A NEW END-OF-DIRECTORY
00752A 05DB A7 03    A        STAA   $3,X     FILE CONTROL BLOCK MUST BE
00753A 05DD 96 C4    A        LDAA   FLNM4    CREATED TO REPLACE THIS ONE)
00754A 05DF A7 04    A        STAA   $4,X     LAST NAME CHARACTER, NOW
00755                 *
00756A 05E1 96 C6    A WTNDX  LDAA   FILAT    LOAD REQUESTED FILE ATTRIBUTE
00757A 05E3 A7 05    A        STAA   5,X      INTO "ATTRIBUTE" IN FCB
00758A 05E5 D6 C7    A        LDAB   FILRQ    PICK UP REQUESTED FILE SIZE
00759A 05E7 96 C8    A        LDAA   FILRQ+1  IN SECTORS TO BE WRITTEN
00760A 05E9 8B 01    A        ADDA   #1       ADD 1, TO ALLOW FOR A POSSIBLE
00761A 05EB C9 00    A        ADCB   #0       SECTOR OVERFLOW
00762A 05ED E7 08    A        STAB   8,X      SET THIS ALLOCATION INTO THE
00763A 05EF A7 09    A        STAA   9,X      FILE LENGTH IN FCB TEMPLATE
00764A 05F1 E6 06    A        LDAB   6,X      LOAD START TRACK FROM OLD FCB
00765A 05F3 A6 07    A        LDAA   7,X      LOAD START SECTOR FROM OLD FCB
00766A 05F5 F7 0153  A        STAB   STRAK    SAVE THE TRACK
00767A 05F8 B7 0154  A        STAA   STSEC    SAVE THE SECTOR
00768A 05FB DE C7    A        LDX    FILRQ    IF THE SECTORS REQUIRED
00769A 05FD 8C 0000  A        CPX    #$0      ARE DEFINED AS 0
00770A 0600 26 07 0609        BNE    WTND1    THEN
00771A 0602 08                INX            ALLOCATE ONE SECTOR
00772A 0603 DF C7    A        STX    FILRQ    AS A MINIMUM REQUEST
```

```
00773A 0605 DE CE    A        LDX    SESP1    GET THE SECTOR BUFFER POINTER
00774A 0607 20 D8 05E1        BRA    WTNDX    ALLOCATE THE 1 SECTOR MINIMUM
00775                 *
00776A 0609 8C 0000  A WTND1  CPX    #$0      IF ALLOCATION IS NOT COMPLETE
00777A 060C 27 0B 0619        BEQ    WTND3    THEN
00778A 060E 4C                INCA            TAKE THE NEXT SECTOR
00779A 060F 81 1B    A        CMPA   #27      IF NO MORE SECTORS
00780A 0611 26 03 0616        BNE    WTND2    THEN
00781A 0613 5C                INCB            TAKE THE NEXT TRACK
00782A 0614 86 01    A        LDAA   #$1      SECTOR NUMBER 1
00783A 0616 09          WTND2  DEX             COUNT OFF THIS SECTOR, IF ALL GONE
00784A 0617 26 F0 0609        BNE    WTND1    THEN
00785A 0619 F7 0151  A WTND3  STAB   CRTRK    SAVE THE NEW START TRACK
00786A 061C B7 0152  A        STAA   CRSEC    AND NEW START SECTOR
00787A 061F DE CE    A        LDX    SESP1    IF NEW FCB & EOD FCB
00788A 0621 8C 013E  A        CPX    #SECB1+110 WILL NOT FIT IN THIS SECTOR
00789A 0624 26 26 064C        BNE    WTND6    THEN
00790                 *
00791A 0626 5F          WTND4  CLRB            SELECT TRACK 0
00792A 0627 B6 0150  A        LDAA   USECT    USING THE CURRENT SECTOR
00793A 062A CE 00D0  A        LDX    #SECB1   AND SECTOR BUFFER
00794A 062D BD 0479  A        JSR    WRTSE    WRITE THE SECTOR BUFFER TO DISK
00795A 0630 4D                TSTA            IF WRITE WAS NOT SUCCESSFUL
00796A 0631 27 0F 0642        BEQ    WTND5    THEN
00797A 0633 7C 0150  A        INC    USECT    TAKE THE NEXT SECTOR
00798A 0636 F6 0150  A        LDAB   USECT    IF THIS SECTOR
00799A 0639 C4 3F    A        ANDB   #$3F     WITH THE UNIT ID STRIPPED
00800A 063B C1 1B    A        CMPB   #27      IS THE LAST SECTOR
00801A 063D 26 E7 0626        BNE    WTND4    THEN
00802A 063F 7E 042E  A        JMP    MEDER    REPORT "MEDIA ERROR" & WAIT FOR NEW COMMAND
00803                 *
00804                 *CURRENT SECTOR HAS NO ROOM FOR EOD, OPEN NEXT SECTOR
00805                 *
00806A 0642 CE 00D0  A WTND5  LDX    #SECB1   INITIALIZE THE SECTOR BUFFER POINTER
00807A 0645 DF CE    A        STX    SESP1    TO THE START OF THE SECTOR BUFFER
00808A 0647 7C 0150  A        INC    USECT    TAKE THE NEXT SECTOR
00809A 064A 20 0B 0657        BRA    WTND7    AND PUT AN E.O.D. FCB THERE
00810                 *
00811                 *PUT AN E.O.D. FILE CONTROL BLOCK IN THE CURRENT SECTOR
00812                 *
00813A 064C 5F          WTND6  CLRB            DEFINE THE NUMBER OF
00814A 064D 86 0B    A        LDAA   #11      BYTES IN A FILE CONTROL BLOCK = 11
00815A 064F 9B CF    A        ADDA   SESP1+LBYT USE THIS NUMBER TO
00816A 0651 D9 CE    A        ADCB   SESP1    STEP OVER THE CURRENT
00817A 0653 97 CF    A        STAA   SESP1+LBYT FILE CONTROL BLOCK
00818A 0655 D7 CE    A        STAB   SESP1    AND ACCESS THE NEXT ONE
00819A 0657 DE CE    A WTND7  LDX    SESP1    LOAD SECTOR BUFFER POINTER
00820A 0659 C6 05    A        LDAB   #$5      CREATE A FILE NAME OF FIVE
00821A 065B 86 2A    A        LDAA   #'*      ASCII ASTERISK CHARACTERS
00822                 *
00823A 065D A7 00    A WTND8  STAA   0,X      STORE A CHARACTER
00824A 065F 08                INX             IN THE NAME FIELD
00825A 0660 5A                DECB            WHILE NAME FIELD
00826A 0661 26 FA 065D        BNE    WTND8    CONTINUE
00827                 *
00828A 0663 86 FF    A        LDAA   #$FF     DEFINE ATTRIBUTE CHARACTER AS
00829A 0665 A7 00    A        STAA   0,X      END OF DIRECTORY MARK
00830A 0667 08                INX             IN THE ATTRIBUTE FIELD
```

```
00831A 0668 B6 0151  A        LDAA   CRTRK    SET THIS FILE'S START TRACK
00832A 066B A7 00    A        STAA   0,X      AS "TRACK"
00833A 066D 08                INX            IN THE TRACK FIELD
00834A 066E B6 0152  A        LDAA   CRSEC    SET THIS FILE'S START SECTOR
00835A 0671 A7 00    A        STAA   0,X      AS "SECTOR"
00836A 0673 5F          WTND9  CLRB           SELECT TRACK 0
00837A 0674 B6 0150  A        LDAA   USECT    USING THE CURRENT SECTOR
00838A 0677 84 3F    A        ANDA   #$3F     MASK OFF THE UNIT ID
00839A 0679 81 1B    A        CMPA   #27      IF SECTORS HAVE RUN OUT
00840A 067B 26 03 0680        BNE    WTNDA    THEN
00841A 067D 7E 042E  A MEDE1  JMP    MEDER    PRINT "MEDIA ERROR" & WAIT FOR NEW COMMAND
00842                 *
00843A 0680 B6 0150  A WTNDA  LDAA   USECT    USING THE CURRENT SECTOR
00844A 0683 CE 00D0  A        LDX    #SECB1   AND THE CURRENT SECTOR BUFFER LOAD
00845A 0686 BD 0479  A        JSR    WRTSE    WRITE THIS SECTOR TO DISK
00846A 0689 4D                TSTA           IF THE WRITE WAS SUCCESFUL
00847A 068A 26 01 068D        BNE    WTNDB    THEN
00848A 068C 39                RTS            RETURN TO CALLER (SUCCESS)
00849                 *
00850A 068D 7C 0150  A WTNDB  INC    USECT    TAKE THE NEXT SECTOR
00851A 0690 20 E1 0673        BRA    WTND9    AND TRY AGAIN
00852                 *
00853                 *
00854                 *
00855                 *
00856                 *THIS ROUTINE CLOSES ANY OPEN FILES, AND RETURNS TO
00857                 *LOOK FOR THE NEXT USER INPUT COMMAND STRING
00858                 *
00859A 0692 0F          FLCLS  SEI            RUN THE CPU NON-INTERRUPTABLE
00860A 0693 8E 006B  A        LDS    #EXSTK   REINITIALIZE EDOS STACK
00861A 0696 7D 0001  A        TST    OFILE    IF THERE IS NO OUTPUT FILE CURRENTLY OPEN
00862A 0699 26 03 069E        BNE    FLCL2    THEN
00863A 069B 7E 016E  A FLCL1  JMP    WTCMD    WAIT FOR NEW COMMAND
00864                 *
00865A 069E 7D 000D  A FLCL2  TST    OCNTR    IF WRITE BUFFER IS NOT FULL
00866A 06A1 27 06 06A9        BEQ    FLCL3    THEN
00867A 06A3 4F                CLRA           USING NULL FILL CHARACTERS
00868A 06A4 BD E80C  A        JSR    XWRT     PAD BUFFER
00869A 06A7 20 F5 069E        BRA    FLCL2    UNTIL FULL
00870                 *
00871A 06A9 CE 0000  A FLCL3  LDX    #0       CREATE A NULL FILE NAME
00872A 06AC DF C0    A        STX    FLNM0    (PREVENTING A MATCH
00873A 06AE DF C2    A        STX    FLNM2    THEREBY FINDING
00874A 06B0 DF C3    A        STX    FLNM3    END OF DIRECTORY)
00875A 06B2 96 02    A        LDAA   OUNIT    DEFINE NULL FILE AS ON THE CURRENT UNIT
00876A 06B4 97 C5    A        STAA   FLNM5    (LOOK FOR EOD ON UNIT HAVING OPEN FILE)
00877A 06B6 4F                CLRA           DEFINE AN ATTRIBUTE OF 0
00878A 06B7 97 C6    A        STAA   FILAT    IN THE ATTRIBUTE LOCATION
00879A 06B9 BD 04BE  A        JSR    FNDFL    TURN TO LAST ENTRY IN DIRECTORY
00880A 06BC 81 FF    A        CMPA   #$FF     IF IT IS END OF DIRECTORY MARK
00881A 06BE 26 BD 067D        BNE    MEDE1    THEN
00882A 06C0 DE CE    A        LDX    SESP1    AT CURRENT SCAN POSITION IN SECTOR BUFFER
00883A 06C2 A6 09    A        LDAA   9,X      GET THE "FREE SPACE" VALUE FROM
00884A 06C4 E6 08    A        LDAB   8,X      THE CURRENT EOD RECORD
00885A 06C6 80 01    A        SUBA   #1       SUBTRACT ONE SECTOR
00886A 06C8 C2 00    A        SBCB   #0       TO BE HELD AS A SPARE
00887A 06CA 90 0A    A        SUBA   OSIZE+LBYT SUBTRACT SIZE OF FILE JUST
00888A 06CC D2 09    A        SBCB   OSIZE    WRITTEN (FILE NOW BEING CLOSED)
```

```
00889A 06CE 97 C8    A        STAA   FILRQ+1  STORE SIZE OF NEW FREE SPACE
00890A 06D0 D7 C7    A        STAB   FILRQ    (NUMBER OF SECTORS NOW AVAILABLE ON DISK)
00891A 06D2 BD 05E1  A        JSR    WTNDX    REWRITE THE DISK DIRECTORY
00892A 06D5 20 C4 069B        BRA    FLCL1    RETURN TO MONITOR, WAIT FOR NEXT COMMAND
00893                 *
00894                 *THIS ROUTINE LOADS A USER OBJECT FILE, OR PROGRAM FROM
00895                 *THE DISK.  IF REQUIRED, THE USER MUST START IT, USING MAID
00896                 *
00897A 06D7 CE E815  A LOAD   LDX    #PROG    LOAD VECTOR TO PROGRAM LOADER IN EDOS ROM
00898A 06DA FF 06F3  A        STX    LOADV    MODIFY JUMP INSTRUCTION FOR PROGRAM LOADER
00899A 06DD BD 0360  A LOAD1  JSR    PRSFL    PARSE FILE NAME TO BE LOADED
00900A 06E0 BD 052C  A        JSR    DFINF    SET UP POINTERS TO FILE
00901A 06E3 4D                TSTA            IF FILE DOES NOT EXIST (DEFINITION FAILED)
00902A 06E4 27 09 06EF        BEQ    LOAD2    THEN
00903A 06E6 F6 0285  A NOSFL  LDAB   NOSFM    DEFINE CHARACTER COUNT AND POINTER
00904A 06E9 CE 0286  A        LDX    #NOSFM+1 FOR "NO SUCH FILE" MESSAGE
00905A 06EC 7E 0189  A        JMP    MSWTC    PRINT IT & WAIT FOR NEW COMMAND
00906                 *
00907A 06EF 8E FF8A  A LOAD2  LDS    #EXBST   SWITCH TO EXBUG STACK SPACE
00908A 06F2 7E E815  A        JMP    PROG     PULL IN THE PROGRAM, START EXBUG
00909          06F3  A LOADV  EQU    *-2      SELF-MODIFIED JUMP ADDRESS FOR LOADER FUNCTION
00910                 *
00911                 *PULL IN RELOCATING LOADER & START IT RUNNING (SYSTEM FUNCTION)
00912                 *
00913A 06F5 CE E824  A RLOAD  LDX    #EDITR   LOAD VECTOR FOR LOAD & GO LOADER (START = 20)
00914A 06F8 FF 06F3  A        STX    LOADV    MODIFY JUMP INSTRUCTION FOR LOAD & GO
00915A 06FB CE 006C  A        LDX    #IBUF1   BACK UP TO GET FILENAME "RLOAD"
00916A 06FE DF BC    A        STX    BFPT1    REDO THE COMMAND AS A FILENAME
00917A 0700 7E 06DD  A        JMP    LOAD1    USING PRSFL,DFINF & LOADER
00918                 *
00919                 *
00920                 *THIS GROUP OF ROUTINES LIST A FILE, USING VARIOUS OUTPUT DEVICES
00921                 *
00922A 0703 CE F018  A CLIST  LDX    #PRINC   DEFINE CONSOLE O/P ROUTINE
00923A 0706 20 03 070B        BRA    LIST1    LIST THE FILE USING IT
00924A 0708 CE EAB0  A PLIST  LDX    #LPTC    DEFINE LINE PRINTER O/P ROUTINE
00925A 070B DF 31    A LIST1  STX    PRNVC+1  STORE SELECTED O/P ROUTINE ADDRESS
00926A 070D 7F 0ECB  A        CLR    NULCT    CLEAR NULL COUNTER
00927A 0710 BD 0360  A        JSR    PRSFL    LOAD FILE NAME AND UID TO FLNM0-5
00928A 0713 BD 052C  A        JSR    DFINF    FIND AND DEFINE FILE FOR READ
00929A 0716 4D                TSTA            IF NO FILE FOUND
00930A 0717 27 03 071C        BEQ    LIST2    THEN
00931A 0719 7E 06E6  A        JMP    NOSFL    PRINT "NO SUCH FILE" & WAIT FOR NEW COMMAND
00932A 071C BD E809  A LIST2  JSR    XRI      READ A 8-BIT BYTE FROM DISK
00933A 071F 25 0C 072D        BCS    LIST4    IF NOT EOF, THEN
00934A 0721 84 7F    A        ANDA   #$7F     STRIP THE PARITY BIT
00935A 0723 27 0B 0730        BEQ    LIST5    IF NOT A NULL, THEN
00936A 0725 7F 0ECB  A        CLR    NULCT    CLEAR NULL COUNTER
00937A 0728 BD 0030  A LIST3  JSR    PRNVC    OUTPUT THE CHARACTER ON THE DEFINED DEVICE
00938A 072B 20 EF 071C        BRA    LIST2    READ THE NEXT BYTE
00939                 *
00940A 072D 7E 016E  A LIST4  JMP    WTCMD    WAIT FOR NEW COMMAND
00941                 *
00942A 0730 7D 0ECB  A LIST5  TST    NULCT    IF PREVIOUS CHAR NOT NULL
00943A 0733 26 E7 071C        BNE    LIST2    THEN
00944A 0735 7C 0ECB  A        INC    NULCT    RECORD THIS NULL
00945A 0738 20 EE 0728        BRA    LIST3    AND PRINT IT
00946                 *
```

```
00947                   *
00948                   *THIS FUNCTION ALLOWS THE USER TO CHANGE FILE ATTRIBUTES
00949                   *
00950A 073A BD 0360  A ATTR   JSR    PRSFL    LOAD FILE NAME & UID TO FLNM0-5
00951A 073D BD 04BE  A        JSR    FNDFL    FIND THE REFERENCED FILE DIRECTORY ENTRY
00952A 0740 4D                TSTA            IF NO SUCH FILE EXISTS
00953A 0741 27 03 0746        BEQ    ATTR1    THEN
00954A 0743 7E 06E6  A        JMP    NOSFL    PRINT "NO SUCH FILE" & WAIT FOR NEW COMMAND
00955A 0746 BD 0360  A ATTR1  JSR    PRSFL    LOAD NEW ATTRIBUTE TO FLNM0
00956A 0749 CE 00C0  A        LDX    #FLNM0   POINT X TO THE NEW ATTRIBUTE (FLNM0)
00957A 074C BD 075F  A        JSR    CONHX    CONVERT INPUT ATTRIBUTE STRING TO BINARY
00958A 074F DE CE    A        LDX    SESP1    RESTORE THE SECTOR BUFFER POINTER
00959A 0751 96 CB    A        LDAA   BINLO    GET THE CONVERTED NEW ATTRIBUTE
00960A 0753 81 FF    A        CMPA   #$FF     END-OF-DIRECTORY IS FORBIDDEN, IF ANYTHING ELSE
00961A 0755 27 05 075C        BEQ    ATTR2    THEN
00962A 0757 A7 05    A        STAA   $5,X     STORE THE NEW ATTRIBUTE IN THE FCB
00963A 0759 7E 07E2  A        JMP    RENA4    AND REWRITE THE SECTOR TO DISK
00964                   *
00965A 075C 7E 0183  A ATTR2  JMP    ERRM     CAN'T ALLOW BLOCKING OFF THE DISK, ERROR!!!
00966                   *
00967                   *
00968                   *THIS ROUTINE CONVERTS ASCII TO HEX
00969                   *
00970A 075F 7F 00CA  A CONHX  CLR    BINHI    CLEAR OUT THE BINARY SCRATCH
00971A 0762 7F 00CB  A        CLR    BINLO    LOCATIONS, PREPARING TO PACK IN HEX DIGITS
00972A 0765 A6 00    A CONH1  LDAA   $0,X     PICK UP THE CURRENT DIGIT
00973A 0767 BD 0772  A        JSR    CNHXS    CONVERT & PACK A HEX DIGIT
00974A 076A 24 03 076F        BCC    CONH2    IF HEX - BIN CONVERSION TERMINATED, THEN
00975A 076C DE CC    A        LDX    BFOP2    PICK UP X FROM BFOP2
00976A 076E 39                RTS             AND RETURN, CONVERSION COMPLETE
00977                   *
00978A 076F 08          CONH2  INX             ELSE POINT TO THE NEXT DIGIT
00979A 0770 20 F3 0765        BRA    CONH1    WHILE DIGITS, CONTINUE
00980                   *
00981A 0772 80 30    A CNHXS  SUBA   #'0      IF .GE. ASCII 0
00982A 0774 25 26 079C        BCS    CNHS2    AND
00983A 0776 8B E9    A        ADDA   #$E9     .LE. ASCII F
00984A 0778 25 22 079C        BCS    CNHS2    THEN
00985A 077A 8B 06    A        ADDA   #$6      IF .LT. ASCII A
00986A 077C 2A 04 0782        BPL    CNHS1    CHECK
00987A 077E 8B 07    A        ADDA   #$7      IF NOT ASCII : TO @
00988A 0780 25 1A 079C        BCS    CNHS2    THEN
00989A 0782 8B 0A    A CNHS1  ADDA   #$A      COMPLETE BINARY CONVERSION
00990A 0784 0C                CLC             (CLEAR C TO PATCH ERROR)
00991A 0785 25 15 079C        BCS    CNHS2    (PROGRAM ERROR-BRANCH NEVER TAKEN)
00992A 0787 97 C9    A        STAA   HDIGT    SAVE THE 4-BIT HEX DIGIT
00993A 0789 D6 CA    A        LDAB   BINHI    PICK UP THE BINARY
00994A 078B 96 CB    A        LDAA   BINLO    STRING IN PROCESS
00995A 078D 48                ASLA            SHIFT IN A 0 BIT
00996A 078E 59                ROLB            4 TIMES, DOUBLE
00997A 078F 48                ASLA            PRECISION, TO
00998A 0790 59                ROLB            PREPARE A HOLE
00999A 0791 48                ASLA            FOR THE NEW HEX DIGIT
01000A 0792 59                ROLB            THE PACK IS RIGHT JUSTIFIED
01001A 0793 48                ASLA            IF MORE THAN 4 DIGITS ARE
01002A 0794 59                ROLB            INPUT, THE 4 LEAST SIGNIFICANT WILL BE USED
01003A 0795 9B C9    A        ADDA   HDIGT    PACK IN 4 BIT BINARY
01004A 0797 D7 CA    A        STAB   BINHI    STORE PACKED BINARY (H)
```

```
01005A 0799 97 CB    A        STAA   BINLO    STORE PACKED BINARY (L)
01006A 079B 0C                CLC            CLEAR C SHOWING SUCCESS (PACK STILL IN PROGRESS)
01007A 079C 39         CNHS2  RTS            RETURN TO CALLER
01008                  *
01009                  *
01010                  *THIS FUNCTION RENAMES AN EXISTING FILE.  THE ONLY EFFECT
01011                  *IS TO CHANGE THE NAME FIELD IN THE SPECIFIED FILE'S FCB.
01012                  *
01013A 079D BD 0360  A RENAM  JSR    PRSFL    PARSE OLD FILE NAME
01014A 07A0 96 C5    A        LDAA   FLNM5    AND ACCEPT THE UNIT NUMBER
01015A 07A2 97 C9    A        STAA   UNUMB    COMMIT TO THIS UNIT NUMBER
01016A 07A4 BD 0360  A        JSR    PRSFL    PARSE NEW FILE NAME
01017A 07A7 96 C9    A        LDAA   UNUMB    BUT USE OLD UNIT NUMBER
01018A 07A9 97 C5    A        STAA   FLNM5    TO FORCE CONSISTENT FILE UID
01019A 07AB BD 04BE  A        JSR    FNDFL    LOOK TO SEE IF THIS NAME ALREADY EXISTS
01020A 07AE 4D                TSTA           IF IT DOES NOT EXIST
01021A 07AF 27 60 0811        BEQ    DUPNM    THEN
01022A 07B1 CE 006C  A        LDX    #IBUF1   START ACTION PASS ON COMMAND STRING
01023A 07B4 DF BC    A        STX    BFPT1    REINITIALIZE COMMAND STRING POINTER
01024A 07B6 BD 0360  A        JSR    PRSFL    STEP OVER "RENAM" COMMAND
01025A 07B9 BD 0360  A        JSR    PRSFL    PARSE THE FILE NAME TO BE CHANGED
01026A 07BC BD 04BE  A        JSR    FNDFL    FIND THE FILE TO BE CHANGED
01027A 07BF 4D                TSTA           IF IT DOES NOT EXIST
01028A 07C0 27 03 07C5        BEQ    RENA2    THEN
01029A 07C2 7E 06E6  A RENA1  JMP    NOSFL    PRINT "NO SUCH FILE" & WAIT FOR NEW COMMAND
01030                  *
01031A 07C5 BD 0360  A RENA2  JSR    PRSFL    PARSE THE NEW FILE NAME
01032A 07C8 DE CE    A        LDX    SESP1    POINT TO THE FILE FCB IN SECTOR BUFFER #1
01033A 07CA 96 C0    A        LDAA   FLNM0    IF THE FIRST CHARACTER OF THE NEW NAME
01034A 07CC 81 20    A        CMPA   #SPC     IS A SPACE, THE NEW NAME IS ILLEGAL
01035A 07CE 27 F2 07C2        BEQ    RENA1    ELSE,
01036A 07D0 A7 00    A        STAA   $0,X     STORE THE FIRST CHARACTER IN THE FCB
01037A 07D2 96 C1    A        LDAA   FLNM1    GET THE SECOND CHARACTER
01038A 07D4 A7 01    A        STAA   $1,X     STORE THE SECOND CHARACTER IN THE FCB
01039A 07D6 96 C2    A        LDAA   FLNM2    GET THE THIRD CHARACTER
01040A 07D8 A7 02    A        STAA   $2,X     STORE THE THIRD CHARACTER IN THE FCB
01041A 07DA 96 C3    A        LDAA   FLNM3    GET THE FOURTH CHARACTER
01042A 07DC A7 03    A        STAA   $3,X     STORE THE FOURTH CHARACTER IN THE FCB
01043A 07DE 96 C4    A        LDAA   FLNM4    GET THE FIFTH CHARACTER
01044A 07E0 A7 04    A        STAA   $4,X     STORE THE FIFTH CHARACTER IN THE FCB
01045A 07E2 5F         RENA4  CLRB           SELECT TRACK 0
01046A 07E3 B6 0150  A        LDAA   USECT    LOAD <UNIT><SECTOR> SET BY FNDFL
01047A 07E6 CE 00D0  A        LDX    #SECB1   RESET X TO POINT TO THE START OF THE SECTOR BUFFER
01048A 07E9 BD 0479  A        JSR    WRTSE    REWRITE THE SECTOR TO THE DISK DIRECTORY
01049A 07EC 4D                TSTA           IF WRITE OF SECTOR WAS SUCCESSFUL
01050A 07ED 26 01 07F0        BNE    RENA5    THEN
01051A 07EF 39                RTS            RETURN TO CALLER (SUCCESS)
01052                  *
01053A 07F0 7E 042E  A RENA5  JMP    MEDER    PRINT "MEDIA ERROR" & WAIT FOR NEW COMMAND
01054                  *
01055                  *
01056                  *THIS FUNCTION CREATES A FILE DIRECTORY ENTRY OF REQUESTED
01057                  *SIZE, APPENDING IT TO THE END OF EXISTING FILES.  ANY DATA
01058                  *PRESENT IN THE FILE AREA REMAINS UNCHANGED, ONLY THE DIRECTORY
01059                  *AREA IS ALTERED.
01060                  *
01061A 07F3 7F 00C6  A CREAT  CLR    FILAT    DEFINE CREATED FILE ATTRIBUTE = 0
01062A 07F6 BD 0360  A        JSR    PRSFL    PARSE FILE NAME (NEXT ITEM IS SIZE IN HEX)
```

```
01063A 07F9 DE BC    A        LDX    BFPT1    POINT TO THE NEXT ITEM
01064A 07FB BD 075F  A        JSR    CONHX    AND CONVERT THE HEX TO BINARY
01065A 07FE DE CA    A        LDX    BINHI    AN ATTEMPT TO CREATE A FILE OF 0 SIZE
01066A 0800 27 21 0823        BEQ    FERR1    IS FORBIDDEN, ELSE
01067A 0802 DF C7    A        STX    FILRQ    DEFINE THE REQUESTED FILE SIZE
01068A 0804 BD 0581  A        JSR    CREAF    CREATE THE NEW FILE DIRECTORY ENTRY
01069A 0807 4D                TSTA            IF SUCCESSFUL IN CREATING THE NEW FILE
01070A 0808 26 03 080D        BNE    ERTYP    THEN
01071A 080A 7E 016E  A        JMP    WTCMD    WAIT FOR NEW COMMAND
01072                 *
01073A 080D 81 02    A ERTYP  CMPA   #$2      IF A = 2
01074A 080F 26 09 081A        BNE    NRMMS    THEN
01075A 0811 F6 0292  A DUPNM  LDAB   DUNMM    LOAD CHARACTER COUNT
01076A 0814 CE 0293  A        LDX    #DUNMM+1 AND POINTER FOR "DUPLICATE NAME"
01077A 0817 7E 0189  A        JMP    MSWTC    PRINT "DUPL NAME" & WAIT FOR NEW COMMAND
01078                 *
01079A 081A F6 029C  A NRMMS  LDAB   NORMM    LOAD CHARACTER COUNT
01080A 081D CE 029D  A        LDX    #NORMM+1 AND POINTER FOR "NO ROOM"
01081A 0820 7E 0189  A        JMP    MSWTC    PRINT "NO ROOM" & WAIT FOR NEW COMMAND
01082                 *
01083A 0823 7E 0183  A FERR1  JMP    ERRM     PRINT "FORMAT ERROR" & WAIT FOR NEW COMMAND
01084                 *
01085                 *
01086A 0826 96 C5    A INIT   LDAA   FLNM5    IT IS FORBIDDEN TO INIT UID = 0
01087A 0828 27 31 085B        BEQ    FERR2    BY COMMAND INIT:0 (INITX MUST BE USED)
01088A 082A 20 03 082F        BRA    INIT1    SAFEGUARD THE SYSTEM DISK
01089                 *
01090          082C  A INITX  EQU    *        THIS ENTRY MUST BE USED TO INIT DISK 0
01091                 *
01092A 082C 7F 00C5  A        CLR    FLNM5    DEFINE UID = 0 (SELECT SYSTEM DISK)
01093A 082F 96 C5    A INIT1  LDAA   FLNM5    LOAD THE DEFINED UID
01094A 0831 0C                CLC             CLEAR THE CARRY BIT
01095A 0832 46                RORA            ROTATE THE UID INTO
01096A 0833 46                RORA            THE TOP 2 BITS
01097A 0834 46                RORA            7 BECOMES OLD 1, 6 BECOMES OLD 0
01098A 0835 8A 03    A        ORAA   #$3      START AT SECTOR 3 (PER IBM FORMAT)
01099A 0837 B7 0150  A        STAA   USECT    <UNIT><SECTOR> IS NOW DEFINED
01100A 083A 86 01    A        LDAA   #$1      DEFINE END-OF-DIRECTORY
01101A 083C B7 0151  A        STAA   CRTRK    AS TRACK #1
01102A 083F B7 0152  A        STAA   CRSEC    AND SECTOR #1
01103A 0842 BD 0642  A        JSR    WTND5    CREATE THE END OF DIRECTORY MARKER
01104A 0845 39                RTS             RETURN TO CALLER
01105                 *
01106                 *
01107                 *THIS FUNCTION WILL HOME THE SELECTED DISK TO TRACK 0
01108                 *
01109A 0846 96 C5    A HOME   LDAA   FLNM5    TAKE THE SPECIFIED UID
01110A 0848 0C                CLC             CLEAR THE C BIT FOR SHIFTING
01111A 0849 46                RORA            SHIFT THE UID INTO
01112A 084A 46                RORA            THE TOP 2 BITS
01113A 084B 46                RORA            7 BECOMES OLD 1, 6 BECOMES OLD 0
01114A 084C 4C                INCA            SPECIFY SECTOR 1
01115A 084D B7 EC06  A        STAA   DKDOD    COMMAND DISK <UNIT><SECTOR>
01116A 0850 86 20    A        LDAA   #$20     USING COMMAND $20
01117A 0852 B7 EC02  A        STAA   DKCOD    TO THE COMMAND REGISTER
01118A 0855 86 0C    A        LDAA   #$C      DEFINE "SEEK TO TRACK 0" COMMAND
01119A 0857 BD EA3E  A        JSR    LOOP     ISSUE COMMAND TO DISK (LOOP ON BUSY)
01120A 085A 39                RTS             RETURN TO CALLER
```

```
01121                   *
01122A 085B 7E 0183  A FERR2  JMP    ERRM     PRINT "FORMAT ERROR" & WAIT FOR NEW COMMAND
01123                   *
01124                   *
01125                   *THIS FUNCTION WILL PRINT A DIRECTORY ON THE CONSOLE DEVICE
01126                   *
01127A 085E CE F018  A CDIR   LDX    #PRINC   DEFINE CONSOLE O/P ROUTINE
01128A 0861 20 03 0866       BRA    DIR1     PRINT THE DIRECTORY, USING CONSOLE
01129                   *
01130                   *
01131                   *THIS FUNCTION WILL PRINT A DIRECTORY ON THE LINE PRINTER
01132                   *
01133A 0863 CE EAB0  A PDIR   LDX    #LPTC    DEFINE LINE PRINTER O/P ROUTINE
01134A 0866 DF 31    A DIR1   STX    PRNVC+1  SELECT DEFINED OUTPUT DEVICE AS PRINTER
01135A 0868 96 C5    A        LDAA   FLNM5    LOAD UID SPECIFIED
01136A 086A 0C                CLC             CLEAR CARRY BIT FOR SHIFTING
01137A 086B 46                RORA            SHIFT UID TO TOP 2 BITS
01138A 086C 46                RORA            OF <UNIT><SECTOR> PACK
01139A 086D 46                RORA            7 BECOMES OLD 1, 6 BECOMES OLD 0
01140A 086E 8A 04    A        ORAA   #$4      START AT SECTOR 4, TO PRINT DIRECTORY
01141A 0870 B7 0150  A        STAA   USECT    DEFINE <UNIT><SECTOR>
01142A 0873 BD 0927  A        JSR    CRLFV    DO <CR><LF> ON PRINT DEVICE
01143A 0876 F6 02A4  A        LDAB   DIRHD    DEFINE MESSAGE COUNT & POINTER
01144A 0879 CE 02A5  A        LDX    #DIRHD+1 "NAME  ATTR TRAK SCTR SIZE"
01145A 087C A6 00    A DIR2   LDAA   $00,X    GET A MESSAGE CHARACTER
01146A 087E BD 0030  A        JSR    PRNVC    PRINT, USING SWITCHED DEVICE
01147A 0881 08                INX             TAKE THE NEXT CHARACTER
01148A 0882 5A                DECB            WHILE CHARACTERS
01149A 0883 26 F7 087C        BNE    DIR2     CONTINUE
01150A 0885 BD 0927  A        JSR    CRLFV    DO <CR><LF> TWICE TO
01151A 0888 BD 0927  A        JSR    CRLFV    SPACE HEADING FROM DIRECTORY
01152A 088B B6 0150  A DIR3   LDAA   USECT    GET CURRENT SECTOR
01153A 088E 84 3F    A        ANDA   #$3F     MASK OFF UID
01154A 0890 81 1B    A        CMPA   #27      IF END OF DIRECTORY TRACK
01155A 0892 26 06 089A        BNE    DIR5     THEN
01156A 0894 BD 0927  A DIR4   JSR    CRLFV    CLOSE DIRECTORY PRINTOUT
01157A 0897 7E 016E  A        JMP    WTCMD    WAIT FOR NEXT COMMAND
01158                   *
01159A 089A B6 0150  A DIR5   LDAA   USECT    LOAD CURRENT <UNIT><SECTOR>
01160A 089D 5F                CLRB            DEFINE TRACK 0
01161A 089E CE 00D0  A        LDX    #SECB1   POINT TO SECTOR BUFFER #1
01162A 08A1 BD 0401  A        JSR    RED11    READ CURRENT SECTOR TO BUFFER
01163A 08A4 4D                TSTA            IF DISK ERROR $80
01164A 08A5 27 05 08AC        BEQ    DIR7     THEN
01165A 08A7 7C 0150  A        INC    USECT    TAKE THE NEXT SECTOR
01166A 08AA 20 DF 088B DIR6   BRA    DIR3     AND READ IT
01167                   *
01168A 08AC CE 00D0  A DIR7   LDX    #SECB1   POINT TO BEGINNING OF SECTOR DATA
01169A 08AF DF CE    A        STX    SESP1    SET THE SECTOR BUFFER POINTER
01170A 08B1 DE CE    A DIR8   LDX    SESP1    GET THE CURRENT SECTOR BUFFER POINTER
01171A 08B3 A6 05    A        LDAA   $5,X     IF THIS FCB ATTRIBUTE IS NOT
01172A 08B5 81 FF    A        CMPA   #$FF     "END-OF-DIRECTORY MARKER"
01173A 08B7 27 DB 0894        BEQ    DIR4     THEN
01174A 08B9 A6 00    A        LDAA   $0,X     LOAD FIRST CHARACTER OF FILE NAME
01175A 08BB BD 0030  A        JSR    PRNVC    AND PRINT IT
01176A 08BE A6 01    A        LDAA   $1,X     LOAD SECOND CHARACTER OF FILE NAME
01177A 08C0 BD 0030  A        JSR    PRNVC    AND PRINT IT
01178A 08C3 A6 02    A        LDAA   $2,X     LOAD THIRD CHARACTER OF FILE NAME
```

```
01179A 08C5 BD 0030  A        JSR    PRNVC    AND PRINT IT
01180A 08C8 A6 03    A        LDAA   $3,X     LOAD FOURTH CHARACTER OF FILE NAME
01181A 08CA BD 0030  A        JSR    PRNVC    AND PRINT IT
01182A 08CD A6 04    A        LDAA   $4,X     LOAD FIFTH CHARACTER OF FILE NAME
01183A 08CF BD 0030  A        JSR    PRNVC    AND PRINT IT
01184A 08D2 BD 0918  A        JSR    OP3SP    SPACE OVER 3 (FORMAT O/P)
01185A 08D5 A6 05    A        LDAA   $5,X     LOAD ATTRIBUTE
01186A 08D7 BD 03E6  A        JSR    PRBIN    CONVERT TO HEX & PRINT IT (TWO DIGITS)
01187A 08DA BD 0918  A        JSR    OP3SP    SPACE OVER 3 (FORMAT O/P)
01188A 08DD A6 06    A        LDAA   $6,X     LOAD FILE STARTING TRACK ADDRESS
01189A 08DF BD 03E6  A        JSR    PRBIN    CONVERT TO HEX & PRINT IT (TWO DIGITS)
01190A 08E2 BD 0918  A        JSR    OP3SP    SPACE OVER 3 (FORMAT O/P)
01191A 08E5 A6 07    A        LDAA   $7,X     LOAD FILE STARTING SECTOR ADDRESS
01192A 08E7 BD 03E6  A        JSR    PRBIN    CONVERT TO HEX & PRINT IT (TWO DIGITS)
01193A 08EA BD 0918  A        JSR    OP3SP    SPACE OVER 3 (FORMAT O/P)
01194A 08ED E6 08    A        LDAB   $8,X     LOAD FILE LENGTH HIGH BYTE
01195A 08EF A6 09    A        LDAA   $9,X     LOAD FILE LENGTH LOW BYTE
01196A 08F1 80 01    A        SUBA   #$1      SUBTRACT 1 SECTOR FROM
01197A 08F3 C2 00    A        SBCB   #$0      FILE LENGTH
01198A 08F5 36                PSHA           SAVE THE LOW BYTE
01199A 08F6 17                TBA            SET UP HIGH BYTE FOR CONVERSION
01200A 08F7 BD 03E6  A        JSR    PRBIN    CONVERT TO HEX & PRINT FIRST TWO DIGITS
01201A 08FA 32                PULA           GET LOW BYTE BACK
01202A 08FB BD 03E6  A        JSR    PRBIN    CONVERT TO HEX & PRINT NEXT TWO DIGITS
01203A 08FE BD 0927  A        JSR    CRLFV    DO <CR><LF> (END OF THIS DIRECTORY ENTRY)
01204A 0901 86 0B    A        LDAA   #11      THERE ARE 11 BYTES IN A FILE CONTROL BLOCK
01205A 0903 5F                CLRB           STEP ON TO THE NEXT FCB OR SECTOR
01206A 0904 9B CF    A        ADDA   SESP1+LBYT ADD 11 TO CURRENT SECTOR POINTER (LOW)
01207A 0906 D9 CE    A        ADCB   SESP1    WITH CARRY AS NEEDED INTO HIGH BYTE
01208A 0908 97 CF    A        STAA   SESP1+LBYT UPDATE NEW CURRENT SECTOR POINTER (LOW)
01209A 090A D7 CE    A        STAB   SESP1    UPDATE NEW CURRENT SECTOR POINTER (HIGH)
01210A 090C DE CE    A        LDX    SESP1    IF THE CURRENT SECTOR POINTER
01211A 090E 8C 0149  A        CPX    #SECB1+121 HAS PASSED THE LAST FCB IN THIS SECTOR
01212A 0911 26 9E 08B1        BNE    DIR8     THEN
01213A 0913 7C 0150  A        INC    USECT    TAKE THE NEXT SECTOR
01214A 0916 20 92 08AA        BRA    DIR6     AND CONTINUE
01215                  *
01216                  *OUTPUT 3 SPACES TO THE PRINT DEVICE (USING THE PRINT VECTOR)
01217                  *
01218A 0918 86 20    A OP3SP  LDAA   #SPC     LOAD AN ASCII SPACE
01219A 091A BD 0030  A        JSR    PRNVC    AND PRINT IT
01220A 091D 86 20    A        LDAA   #SPC     LOAD AN ASCII SPACE
01221A 091F BD 0030  A        JSR    PRNVC    AND PRINT IT
01222A 0922 86 20    A        LDAA   #SPC     LOAD AN ASCII SPACE
01223A 0924 7E 0030  A        JMP    PRNVC    PRINT IT & RETURN TO CALLER
01224                  *
01225                  *
01226                  *OUTPUT <CR><LF> TO THE PRINT DEVICE (USING THE PRINT VECTOR)
01227                  *
01228A 0927 86 0D    A CRLFV  LDAA   #CR      LOAD AN ASCII <CR>
01229A 0929 BD 0030  A        JSR    PRNVC    AND PRINT IT
01230A 092C 86 0A    A        LDAA   #LF      LOAD AN ASCII <LF>
01231A 092E 7E 0030  A        JMP    PRNVC    PRINT IT & RETURN TO CALLER
01232                  *
01233                  *
01234                  *
01235                  *THIS FUNCTION PREPARES & RUNS THE RESIDENT ASSEMBLER
01236                  *
```

```
01237A 0931 BD 0360  A ASM    JSR    PRSFL    PARSE THE REQUESTED ASSEMBLY OPTION NUMBER
01238A 0934 CE 00C0  A        LDX    #FLNM0   POINT TO THE PARSED NUMBER
01239A 0937 BD 075F  A        JSR    CONHX    CONVERT IT TO A BINARY INTEGER
01240A 093A 96 CB    A        LDAA   BINLO    PICK UP THE CONVERTED VALUE
01241A 093C 36                PSHA           SAVE IT ON THE STACK
01242A 093D 80 05    A        SUBA   #$5      IF IT WAS GREATER THAN 5
01243A 093F 2A 3E 097F        BPL    ASM2     A FORMAT ERROR HAS OCCURED
01244A 0941 32                PULA           RECOVER THE VALUE
01245A 0942 36                PSHA           AND RESTORE THE STACK COPY
01246A 0943 80 02    A        SUBA   #$2      IF IT WAS LESS THAN 2
01247A 0945 2B 38 097F        BMI    ASM2     A FORMAT ERROR HAS OCCURED
01248A 0947 32                PULA           RECOVER THE VALUE
01249A 0948 48                ASLA           DOUBLE IT (2=4, 3=6, 4=8)
01250A 0949 4C                INCA           & ADD 1 (2=5, 3=7, 4=9)
01251A 094A 97 00    A        STAA   PASS     THIS IS THE ASSEMBLER OPTION SWITCH VALUE
01252A 094C BD 0BD8  A        JSR    SVIOF    PARSE & SET UP OUTPUT FILE
01253A 094F BD 0B6E  A        JSR    NONAM    PARSE INPUT FILE NAME, CONFIRM EXISTENCE
01254A 0952 BD 0BDE  A        JSR    SVIPF    SAVE INPUT FILE DEFS DURING ASSEMBLER LOAD
01255A 0955 86 41    A        LDAA   #'A      DEFINE "ASMB" AS IMMEDIATE DATA
01256A 0957 B7 096A  A        STAA   TCHR0    TO SELECT THE ASMB ASSEMBLER FILE
01257A 095A 96 00    A        LDAA   PASS     IF SELECTED ASSEMBLER OPTION
01258A 095C 81 07    A        CMPA   #$7      DOES NOT REQUIRE AN OUTPUT FILE (LISTING ONLY)
01259A 095E 26 03 0963        BNE    ASM1     THEN
01260A 0960 7F 0001  A        CLR    OFILE    CLEAR THE OUTPUT FILE REQUEST FLAG
01261A 0963 7F 00C5  A ASM1   CLR    FLNM5    SELECT SYSTEM DISK FOR PROGRAM LOAD
01262A 0966 CE 00C0  A        LDX    #FLNM0   SET POINTER TO BUILD FILENAME
01263A 0969 86 41    A        LDAA   #'A      THIS ROUTINE IS LOADED
01264          096A  A TCHR0  EQU    *-1
01265A 096B A7 00    A        STAA   $0,X     WITH APPROPRIATE IMMEDIATE
01266A 096D 86 53    A        LDAA   #'S      DATA TO DEFINE
01267A 096F A7 01    A        STAA   $1,X     A PROTOTYPE TEMPLATE
01268A 0971 86 4D    A        LDAA   #'M      FOR THE SYSTEM FILE (PROGRAM)
01269A 0973 A7 02    A        STAA   $2,X     TO BE LOADED & RUN
01270A 0975 86 42    A        LDAA   #'B      (ASMB, RSMB)
01271A 0977 A7 03    A        STAA   $3,X     BEFORE THIS ROUTINE IS USED
01272A 0979 7F 0157  A        CLR    LGOFL    CLEAR LOAD-&-GO FLAG
01273A 097C 7E 0B3A  A        JMP    GEPRO    PULL IN PROGRAM & RUN IT
01274                  *
01275A 097F 7E 0183  A ASM2   JMP    ERRM     PRINT "FORMAT ERROR" & WAIT FOR NEW COMMAND
01276                  *
01277                  *
01278                  *
01279                  *THIS FUNCTION SETS UP AND RUNS THE RELOCATING ASSEMBLER
01280                  *
01281A 0982 86 52    A RASM   LDAA   #'R      DEFINE "RSMB" AS IMMEDIATE DATA
01282A 0984 B7 096A  A        STAA   TCHR0    TO SELECT THE RSMB ASSEMBLER FILE
01283A 0987 CE 1200  A        LDX    #BGBUF   LOAD POINTER TO LARGE BUFFER AREA
01284A 098A FF 0159  A        STX    TKBFP    SET TRACK BUFFER #1 POINTER
01285A 098D 4F                CLRA           DEFINE VALUE "0"
01286A 098E B7 015B  A        STAA   AERFL    CLEAR FORMAT ERROR FLAG
01287A 0991 97 01    A        STAA   OFILE    ASSUME NO O/P FILE, CLEAR UPDATE FLAG
01288A 0993 86 4F    A        LDAA   #'O      DEFINE "OUTPUT" FILE FLAG
01289A 0995 B7 0A36  A        STAA   LSFI2    SET FOR LISTING FILE OR DEVICE DETERMINATION
01290A 0998 B7 0A80  A        STAA   LSFIB    SET FOR OUTPUT FILE CONFLICT DETERMINATION
01291A 099B BD 0360  A        JSR    PRSFL    PARSE LISTING DEVICE & OPTIONS
01292A 099E 96 C0    A        LDAA   FLNM0    IF THE FIRST CHARACTER
01293A 09A0 81 20    A        CMPA   #SPC     IS A SPACE (NO LISTING FILE OR DEVICE)
01294A 09A2 26 05 09A9        BNE    RASM2    THEN
```

```
01295A 09A4 BD 0A92  A        JSR    ZERFL    ENTER "NULL ENTRY" IN LISTING POS'N
01296A 09A7 20 03 09AC        BRA    RASM3    AND SKIP LISTING FILE DEFINITION
01297A 09A9 BD 0A1E  A RASM2  JSR    LSFIL    DEFINE THE LISTING FILE/DEVICE
01298A 09AC BD 0360  A RASM3  JSR    PRSFL    PARSE OBJECT FILE NAME
01299A 09AF 96 C0    A        LDAA   FLNM0    IF FIRST CHARACTER IS
01300A 09B1 81 20    A        CMPA   #SPC     A SPACE (NO OBJECT FILE REQUIRED)
01301A 09B3 26 05 09BA        BNE    RASM4    THEN
01302A 09B5 BD 0A92  A        JSR    ZERFL    ENTER "NULL ENTRY" IN OBJECT POS'N
01303A 09B8 20 03 09BD        BRA    RASM5    AND SKIP OBJECT FILE DEFINITION
01304A 09BA BD 0A1E  A RASM4  JSR    LSFIL    DEFINE THE OBJECT FILE
01305A 09BD BD 0360  A RASM5  JSR    PRSFL    PARSE SOURCE FILE NAME
01306A 09C0 96 C0    A        LDAA   FLNM0    GET FIRST CHAR OF FIRST FILE NAME
01307A 09C2 81 20    A        CMPA   #SPC     IF THERE IS AT LEAST ONE SOURCE FILE
01308A 09C4 27 46 0A0C        BEQ    RASMA    THEN
01309A 09C6 86 49    A        LDAA   #'I      DEFINE "INPUT" FLAG VALUE
01310A 09C8 BD 0A18  A        JSR    LSFI0    DEFINE FIRST SOURCE FILE
01311A 09CB 4D                TSTA            IF THERE WAS AN ERROR RETURN
01312A 09CC 2F 03 09D1        BLE    RASM6    THEN
01313A 09CE 7C 015B  A        INC    AERFL    SET THE FORMAT ERROR FLAG
01314A 09D1 BD 0360  A RASM6  JSR    PRSFL    PARSE THE NEXT SOURCE FILE NAME
01315A 09D4 96 C0    A        LDAA   FLNM0    GET FIRST CHARACTER OF FILE NAME
01316A 09D6 81 20    A        CMPA   #SPC     IF FILES HAVE NOT RUN OUT
01317A 09D8 27 13 09ED        BEQ    RASM8    THEN
01318A 09DA BD 0A1E  A        JSR    LSFIL    DEFINE THIS SOURCE FILE
01319A 09DD 4D                TSTA            IF THERE WAS AN ERROR RETURN
01320A 09DE 2F 03 09E3        BLE    RASM7    THEN
01321A 09E0 7C 015B  A        INC    AERFL    SET THE FORMAT ERROR FLAG
01322A 09E3 CE 127D  A RASM7  LDX    #TRKBF+125 IF THE LIMIT THAT WILL GO IN A SECTOR
01323A 09E6 BC 0159  A        CPX    TKBFP    HAS NOT BEEN REACHED (25 ENTRIES MAX)
01324A 09E9 27 02 09ED        BEQ    RASM8    THEN
01325A 09EB 20 E4 09D1        BRA    RASM6    LOOK FOR ANOTHER SOURCE FILE
01326A 09ED 86 FF    A RASM8  LDAA   #$FF     DEFINE A DELIMITER CHARACTER
01327A 09EF FE 0159  A        LDX    TKBFP    USING THE BUFFER POINTER
01328A 09F2 A7 00    A        STAA   $0,X     STORE THE DELIMITER (END MARK)
01329A 09F4 CE 1200  A        LDX    #TRKBF   SET A POINTER AT THE BEGINNING OF THE BUFFER
01330A 09F7 86 03    A        LDAA   #$3      REQUEST DISK 0, SECTOR 3 (FILE STRING SAVE SECTOR)
01331A 09F9 C6 00    A        LDAB   #$0      USING TRACK 0
01332A 09FB BD 0479  A        JSR    WRTSE    WRITE THIS SECTOR TO DISK
01333A 09FE 4D                TSTA            IF DISK WRITE WAS SUCCESSFUL
01334A 09FF 26 14 0A15        BNE    RASMB    AND IF
01335A 0A01 7D 015B  A        TST    AERFL    FORMAT ERROR FLAG WAS NOT SET
01336A 0A04 26 03 0A09        BNE    RASM9    THEN
01337A 0A06 7E 0963  A        JMP    ASM1     PULL IN THE PROGRAM & RUN IT
01338                 *
01339A 0A09 7E 0183  A RASM9  JMP    ERRM     PRINT "FORMAT ERROR" & WAIT FOR A NEW COMMAND
01340                 *
01341A 0A0C F6 02FC  A RASMA  LDAB   NOSRM    LOAD CHARACTER COUNT AND
01342A 0A0F CE 02FD  A        LDX    #NOSRM+1 POINTER TO "NO SOURCE FILE"
01343A 0A12 7E 0189  A        JMP    MSWTC    PRINT "NO SOURCE FILE" & WAIT FOR NEW COMMAND
01344                 *
01345A 0A15 7E 042E  A RASMB  JMP    MEDER    PRINT "MEDIA ERROR" & WAIT FOR NEW COMMAND
01346                 *
01347                 *IDENTIFY LISTING FILE OR DEVICE & SET IT UP
01348                 *
01349A 0A18 B7 0A80  A LSFI0  STAA   LSFIB    SAVE INP/OP FLAG FOR FILE ANALYSIS
01350A 0A1B B7 0A36  A        STAA   LSFI2    SAVE INP/OP FLAG FOR CONFLICT CHECK
01351                 *
01352A 0A1E 96 C0    A LSFIL  LDAA   FLNM0    IF THE FIRST CHARACTER OF THE FILENAME
```

```
01353A 0A20 81 23    A        CMPA   #'#      IS AN ASCII "#"
01354A 0A22 26 5B 0A7F        BNE    LSFIA    THEN
01355A 0A24 CE 0B00  A        LDX    #FLABF   LOOK IN THE FILE ASSIGNMENT BUFFER
01356A 0A27 96 C1    A LSFI1  LDAA   FLNM1    IF THE FIRST CHAR OF THE FILE NAME
01357A 0A29 E6 00    A        LDAB   $0,X     AND THE FIRST CHAR OF THE ASSIGNMENT
01358A 0A2B 11                CBA            ARE THE SAME
01359A 0A2C 26 13 0A41        BNE    LSFI3    AND
01360A 0A2E 96 C2    A        LDAA   FLNM2    IF THE SECOND CHAR OF THE FILE NAME
01361A 0A30 E6 01    A        LDAB   $1,X     AND THE SECOND CHAR OF THE ASSIGNMENT
01362A 0A32 11                CBA            ARE THE SAME
01363A 0A33 26 0C 0A41        BNE    LSFI3    AND
01364A 0A35 86 00    A        LDAA   #$0      THE SWITCH CHARACTER
01365          0A36  A LSFI2  EQU    *-1      WHICH WAS STORED HERE
01366A 0A37 E6 02    A        LDAB   $2,X     AND THE THIRD CHAR OF THE ASSIGNMENT
01367A 0A39 11                CBA            ARE THE SAME
01368A 0A3A 27 16 0A52        BEQ    LSFI4    TRANSLATE THE FILE NAME
01369A 0A3C 86 20    A        LDAA   #SPC     IF A THIRD CHAR OF "SPACE"
01370A 0A3E 11                CBA            PRODUCES A MATCH
01371A 0A3F 27 11 0A52        BEQ    LSFI4    TRANSLATE THE FILE NAME, ELSE
01372A 0A41 08          LSFI3  INX            STEP OVER AN ASSIGNMENT
01373A 0A42 08                INX            PROTOTYPE RECORD
01374A 0A43 08                INX            OF 3 CHARACTERS
01375A 0A44 8C 0B12  A        CPX    #FLABE   IF THIS WAS THE LAST RECORD
01376A 0A47 26 DE 0A27        BNE    LSFI1    THEN
01377A 0A49 F6 030B  A        LDAB   DEVEM    LOAD CHARACTER COUNT AND
01378A 0A4C CE 030C  A        LDX    #DEVEM+1 POINTER TO "DEVICE ERROR"
01379A 0A4F 7E 0189  A        JMP    MSWTC    PRINT "DEVICE ERROR" & WAIT FOR NEW COMMAND
01380                 *
01381A 0A52 FE 0159  A LSFI4  LDX    TKBFP    LOAD A BUFFER POINTER
01382A 0A55 96 C1    A        LDAA   FLNM1    LOAD FIRST CHAR OF DEVICE
01383A 0A57 A7 00    A        STAA   $0,X     (FIRST CHAR OF THIS ITEM WAS "#")
01384A 0A59 96 C2    A        LDAA   FLNM2    LOAD SECOND CHAR OF DEVICE
01385A 0A5B A7 01    A        STAA   $1,X     STORE IT IN THE BUFFER
01386A 0A5D 96 C3    A        LDAA   FLNM3    LOAD "OPTION" CHARACTER
01387A 0A5F C6 39    A        LDAB   #'9      DEFINE ASCII "9"
01388A 0A61 11                CBA            IF LESS THAN ASCII "9" OR EQUAL
01389A 0A62 2A 0F 0A73        BPL    LSFI8    AND
01390A 0A64 C6 30    A        LDAB   #'0      DEFINE ASCII "0"
01391A 0A66 11                CBA            GREATER THAN ASCII "0" OR EQUAL
01392A 0A67 2B 0A 0A73        BMI    LSFI8    THEN
01393A 0A69 A7 02    A LSFI5  STAA   $2,X     STORE THE OPTION IN THE BUFFER
01394A 0A6B 4F          LSFI6  CLRA           DEFINE CHARACTER = 0
01395A 0A6C A7 03    A        STAA   $3,X     AS NEXT CHARACTER
01396A 0A6E 4A                DECA           DEFINE CHARACTER = $FF (EOF MARK)
01397A 0A6F A7 04    A LSFI7  STAA   $4,X     AS NEXT CHARACTER (FILE DELIMITER)
01398A 0A71 20 03 0A76        BRA    LSFI9    STEP OVER THE CREATED RECORD
01399                 *
01400A 0A73 4F          LSFI8  CLRA           DEFINE "OPTION" = 0
01401A 0A74 20 F3 0A69        BRA    LSFI5    AND COMPLETE THE RECORD
01402                 *
01403A 0A76 08          LSFI9  INX            STEP THE
01404A 0A77 08                INX            BUFFER POINTER
01405A 0A78 08                INX            OVER A COMPLETE
01406A 0A79 08                INX            5 CHARACTER
01407A 0A7A 08                INX            O/P DEVICE/OPTION RECORD
01408A 0A7B FF 0159  A        STX    TKBFP    AND UPDATE THE BUFFER POINTER
01409A 0A7E 39                RTS            RETURN TO CALLER
01410                 *
```

```
01411A 0A7F 86 4F    A LSFIA  LDAA   #'O      LOAD I/O FILE ID SWITCH
01412          0A80  A LSFIB  EQU    *-1      SWITCH VALUE SET HERE
01413A 0A81 81 49    A        CMPA   #'I      IF NOT AN INPUT FILE
01414A 0A83 27 19 0A9E        BEQ    IPFIL    AND
01415A 0A85 81 4F    A        CMPA   #'O      IF NOT AN OUTPUT FILE
01416A 0A87 27 53 0ADC        BEQ    OPFIL    THEN
01417A 0A89 F6 0318  A        LDAB   OFLCM    A LISTING FILE IS IN CONFLICT
01418A 0A8C CE 0319  A        LDX    #OFLCM+1 DEFINE "OFILE CONFLICT"
01419A 0A8F 7E 0189  A        JMP    MSWTC    PRINT "OFILE CONFLICT" & WAIT FOR NEW COMMAND
01420                  *
01421                  *DEFINE A ALL "0" ENTRY IN BUFFER, UNUSED LIST OR OBJECT FILES
01422                  *
01423A 0A92 4F         ZERFL  CLRA             DEFINE VALUE OF "0"
01424A 0A93 FE 0159  A        LDX    TKBFP    USING THE CURRENT BUFFER POINTER
01425A 0A96 A7 00    A        STAA   $0,X     BUILD A
01426A 0A98 A7 01    A        STAA   $1,X     RECORD OF
01427A 0A9A A7 02    A        STAA   $2,X     000
01428A 0A9C 20 CD 0A6B        BRA    LSFI6    FINISH THE RECORD & EXIT
01429                  *
01430                  *
01431                  *DEFINE & SET UP AN INPUT FILE
01432                  *
01433A 0A9E BD 052C  A IPFIL  JSR    DFINF    IF THE REQUESTED FILE
01434A 0AA1 4D                TSTA             DOES NOT EXIST
01435A 0AA2 27 21 0AC5        BEQ    IPFI1    THEN
01436A 0AA4 C6 05    A        LDAB   #$5      FOR THE 5 CHARACTERS
01437A 0AA6 CE 00C0  A        LDX    #FLNM0   OF THE REQUESTED FILE NAME
01438A 0AA9 BD 03B3  A        JSR    PRINS    PRINT THE FILE NAME
01439A 0AAC 86 3A    A        LDAA   #':     DEFINE ASCII COLON
01440A 0AAE BD F018  A        JSR    PRINC    PRINT IT
01441A 0AB1 96 C5    A        LDAA   FLNM5    GET CURRENT UID
01442A 0AB3 8B 30    A        ADDA   #'0      CONVERT TO ASCII NUMBER
01443A 0AB5 BD F018  A        JSR    PRINC    PRINT IT
01444A 0AB8 F6 02DB  A        LDAB   NOEXM    DEFINE CHARACTER COUNT
01445A 0ABB CE 02DC  A        LDX    #NOEXM+1 AND PONTER FOR " NON EXISTENT"
01446A 0ABE BD 03B3  A        JSR    PRINS    PRINT " NON EXISTENT"
01447A 0AC1 BD 03BD  A        JSR    CRLF     PRINT <CR><LF>
01448A 0AC4 39                RTS             RETURN TO CALLER
01449                  *
01450A 0AC5 FE 0159  A IPFI1  LDX    APARS    LOAD POINTER TO STRING BUFFER
01451A 0AC8 96 06    A        LDAA   ITRK     GET THE START TRACK OF THE FILE
01452A 0ACA A7 00    A        STAA   $0,X     AND STORE IT IN THE BUFFER
01453A 0ACC 96 07    A        LDAA   ISCTR    GET THE DRIVE & SECTOR OF THE FILE
01454A 0ACE A7 01    A        STAA   $1,X     AND STORE IT IN THE BUFFER
01455A 0AD0 96 04    A        LDAA   ISIZE    LOAD THE SIZE HIGH BYTE
01456A 0AD2 A7 02    A        STAA   $2,X     AND STORE IT IN THE BUFFER
01457A 0AD4 96 05    A        LDAA   ISIZE+LBYT LOAD THE SIZE LOW BYTE
01458A 0AD6 A7 03    A        STAA   $3,X     AND STORE IT IN THE BUFFER
01459A 0AD8 4F                CLRA             DELIMIT THIS RECORD WITH "0"
01460A 0AD9 7E 0A6F  A        JMP    LSFI7    COMPLETING THIS BUFFER RECORD
01461                  *
01462                  *
01463A 0ADC BD 0B8A  A OPFIL  JSR    DOOF1    CREATE A PROTOTYPE FILE CONTROL BLOCK
01464A 0ADF BD 0BAC  A        JSR    DEFOP    DEFINE THE OUTPUT FILE
01465A 0AE2 FE 0159  A        LDX    APARS    LOAD POINTER TO STRING BUFFER
01466A 0AE5 96 0B    A        LDAA   OTRK     GET THE START TRACK OF THE FILE
01467A 0AE7 A7 00    A        STAA   $0,X     AND STORE IT IN THE BUFFER
01468A 0AE9 96 0C    A        LDAA   OSCTR    GET THE DRIVE & SECTOR OF THE FILE
```

```
01469A 0AEB A7 01    A        STAA   $1,X     AND STORE IT IN THE BUFFER
01470A 0AED 96 09    A        LDAA   OSIZE    LOAD THE SIZE HIGH BYTE
01471A 0AEF A7 02    A        STAA   $2,X     AND STORE IT IN THE BUFFER
01472A 0AF1 96 0A    A        LDAA   OSIZE+LBYT LOAD THE SIZE LOW BYTE
01473A 0AF3 A7 03    A        STAA   $3,X     AND STORE IT IN THE BUFFER
01474A 0AF5 86 FF    A        LDAA   #$FF     DEFINE FLAG VALUE = FF (DO UPDATE)
01475A 0AF7 97 01    A        STAA   OFILE    FOR FILE CLOSURE ON EDOS RETURN
01476A 0AF9 7C 0A80  A        INC    LSFIB    SET FLAG TO SHOW O/P FILE NOW ASSIGNED
01477A 0AFC 4F                 CLRA            DELIMIT THIS RECORD WITH "0"
01478A 0AFD 7E 0A6F  A        JMP    LSFI7    COMPLETING THIS BUFFER RECORD
01479                *
01480                *
01481A 0B00    0012  A FLABF  RMB    18       INPUT COMMAND STRING FILE ASSIGNMENT BUFFER
01482          0B12  A FLABE  EQU    *        END OF BUFFER SPACE ALLOCATED
01483                *
01484                *
01485A 0B12 BD 0360  A EDIT   JSR    PRSFL    PARSE INPUT STRING FOR SOURCE FILE NAME
01486A 0B15 96 C0    A        LDAA   FLNM0    IF NO CHARACTERS, SOURCE
01487A 0B17 81 20    A        CMPA   #SPC     IS FROM KEYBOARD ONLY
01488A 0B19 27 3C 0B57        BEQ    EDIT2    ELSE
01489A 0B1B BD 0B77  A        JSR    EDINF    GET SOURCE FILE
01490A 0B1E BD 0BD8  A        JSR    SVIOF    DEFINE OUTPUT FILE & SAVE I/P FILE
01491                *
01492A 0B21 7F 00C5  A EDIT1  CLR    FLNM5    SELECT THE SYSTEM DISK
01493A 0B24 CE 00C0  A        LDX    #FLNM0   POINT TO THE SEARCH STRING
01494A 0B27 86 45    A        LDAA   #'E      BUILD UP
01495A 0B29 A7 00    A        STAA   0,X      AN IMAGE
01496A 0B2B 86 44    A        LDAA   #'D      OF "EDIT "
01497A 0B2D A7 01    A        STAA   1,X      AS A SEARCH
01498A 0B2F 86 49    A        LDAA   #'I      PATTERN
01499A 0B31 A7 02    A        STAA   2,X      (LOOK FOR
01500A 0B33 86 54    A        LDAA   #'T      THE EDITOR
01501A 0B35 A7 03    A        STAA   3,X      PROGRAM FILE)
01502A 0B37 B7 0157  A        STAA   LGOFL    SET LOAD-&-GO FLAG
01503A 0B3A 86 20    A GEPRO  LDAA   #SPC     PAD THE NAME
01504A 0B3C A7 04    A        STAA   4,X      TO 5 CHARACTERS
01505A 0B3E BD 052C  A        JSR    DFINF    IF THE REQUESTED PROGRAM OBJECT FILE
01506A 0B41 4D                 TSTA            IS NOT RESIDENT ON THE DISK
01507A 0B42 27 03 0B47        BEQ    GEPR1    THEN
01508A 0B44 7E 06E6  A        JMP    NOSFL    PRINT "NO SUCH FILE" & WAIT FOR NEW COMMAND
01509A 0B47 B6 0156  A GEPR1  LDAA   TIUNT    ELSE DEFINE THE LOAD MODULE DISK
01510A 0B4A 97 03    A        STAA   IUNIT    IN IUNIT
01511A 0B4C 7D 0157  A        TST    LGOFL    IF FLAG IS .NE. 0
01512A 0B4F 27 03 0B54        BEQ    GEPR2    THEN
01513A 0B51 7E E824  A        JMP    EDITR    PULL IN THE LOAD MODULE
01514A 0B54 7E E824  A GEPR2  JMP    EDITR    ELSE PULL IN THE LOAD MODULE
01515                *
01516                *EDITOR FILE CREATION, NO I/P FILE, SET UP A DUMMY FILE
01517                *
01518A 0B57 CE 0001  A EDIT2  LDX    #$1      DEFINE A DUMMY FILE SIZE OF 1 SECTOR
01519A 0B5A DF 04    A        STX    ISIZE    FOR THE ISIZE SETUP VALUE
01520A 0B5C 86 01    A        LDAA   #$1      DEFINE TRACK NUMBER
01521A 0B5E 97 06    A        STAA   ITRK     AS TRACK 1
01522A 0B60 7F 0008  A        CLR    ICNTR    CLEAR THE INPUT BYTE COUNTER
01523A 0B63 7F 0007  A        CLR    ISCTR    DEFINE INPUT <UNIT><SECTOR> AS 0
01524A 0B66 7F 0003  A        CLR    IUNIT    ON UNIT 0
01525A 0B69 BD 0BD8  A        JSR    SVIOF    PREPARE FCB, AND SAVE I/O FILE DEFINITIONS
01526A 0B6C 20 B3 0B21        BRA    EDIT1    AND CALL IN THE EDITOR (KEYBOARD INPUT ONLY)
```

```
01527                 *
01528                 *THIS ROUTINE USED BY ASSEMBLER TO CHECK FOR A MISSING NAME
01529                 *
01530A 0B6E BD 0360  A NONAM  JSR    PRSFL    IF NEXT NAME IN THE INPUT COMMAND STRING
01531A 0B71 96 C0    A        LDAA   FLNM0    DOES NOT OVERWITE THE FIRST SPACE
01532A 0B73 81 20    A        CMPA   #SPC     IT IS ASSUMED NOT TO EXIST
01533A 0B75 27 07 0B7E        BEQ    EDNOF    ELSE
01534                 *
01535A 0B77 BD 052C  A EDINF  JSR    DFINF    IF THE INPUT FILE IS FOUND
01536A 0B7A 4D                TSTA            AND SUCCESSFULLY DEFINED
01537A 0B7B 26 01 0B7E        BNE    EDNOF    THEN
01538A 0B7D 39                RTS             RETURN (SUCCESS)
01539                 *
01540A 0B7E 7E 06E6  A EDNOF  JMP    NOSFL    PRINT "NO SUCH FILE" & WAIT FOR NEW COMMAND
01541                 *
01542A 0B81 BD 0360  A DOOFL  JSR    PRSFL    PARSE OUTPUT FILE NAME
01543A 0B84 96 C0    A        LDAA   FLNM0    IF OUTPUT FILE WAS SPECIFIED
01544A 0B86 81 20    A        CMPA   #SPC     (VALID CHAR OVERWRITES FIRST SPACE CHAR)
01545A 0B88 27 F4 0B7E        BEQ    EDNOF    THEN
01546                 *
01547                 *
01548                 *CREATE A PROTOTYPE FCB, ATTR = FF (EOD), SIZE = FREE SPACE ON DISK
01549                 *
01550A 0B8A CE 07B7  A DOOF1  LDX    #1975    ALLOW AN ALLOCATION OF 1975 SECTORS (FULL DISK)
01551A 0B8D DF 09    A        STX    OSIZE    IN THE OSIZE LOCATION
01552A 0B8F DF C7    A        STX    FILRQ    AND DEFINE THE REQUESTED FILE SIZE
01553A 0B91 86 FF    A DOOF2  LDAA   #$FF     DEFINE END OF DIRECTORY (NEW FILE GOES ON THE END)
01554A 0B93 97 C6    A        STAA   FILAT    WITH ATTRIBUTE = $FF
01555A 0B95 BD 0581  A        JSR    CREAF    CREATE THE FILE DIRECTORY ENTRY
01556A 0B98 4D                TSTA            IF THE FILE CAN BE CREATED
01557A 0B99 26 01 0B9C        BNE    DOOF3    THEN
01558A 0B9B 39                RTS             RETURN (SUCCESS)
01559                 *
01560A 0B9C 81 03    A DOOF3  CMPA   #$3      IF ERROR WORSE THAN DISK NOT EMPTY
01561A 0B9E 27 03 0BA3        BEQ    DOOF4    THEN
01562A 0BA0 7E 080D  A        JMP    ERTYP    SORT OUT & REPORT THE ERROR
01563                 *
01564A 0BA3 DF 09    A DOOF4  STX    OSIZE    THIS IS THE FREE SPACE AVAILABLE
01565A 0BA5 DF C7    A        STX    FILRQ    IF FREE SPACE IS = 0
01566A 0BA7 26 E8 0B91        BNE    DOOF2    THEN
01567A 0BA9 7E 081A  A        JMP    NRMMS    PRINT "NO ROOM" & WAIT FOR NEW COMMAND
01568                 *
01569                 *
01570A 0BAC B6 0153  A DEFOP  LDAA   STRAK    PICK UP START TRACK ADDRESS FOR PROTOTYPE
01571A 0BAF 97 0B    A        STAA   OTRK     DEFINE OUTPUT FILE TRACK
01572A 0BB1 96 C5    A        LDAA   FLNM5    PICK UP FILE NAME UID
01573A 0BB3 97 02    A        STAA   OUNIT    AND DEFINE OUTPUT UNIT NUMBER
01574A 0BB5 0C                CLC             CLEAR C FOR SHIFTING
01575A 0BB6 46                RORA            SHIFT UID INTO TOP 2 BITS
01576A 0BB7 46                RORA            OF <UNIT><SECTOR> PACK
01577A 0BB8 46                RORA            BIT 7 BECOMES OLD 1, 6 BECOMES OLD 0
01578A 0BB9 BA 0154  A        ORAA   STSEC    OR IN START SECTOR ADDRESS
01579A 0BBC 97 0C    A        STAA   OSCTR    DEFINE OUTPUT <UNIT><SECTOR>
01580A 0BBE 7F 000D  A        CLR    OCNTR    CLEAR OUTPUT BYTE COUNT
01581A 0BC1 39                RTS             RETURN DEFINED OUTPUT FILE
01582                 *
01583                 *
01584A 0BC2 96 06    A MOVIP  LDAA   ITRK     TAKE CURRENT DEFINED INPUT FILE TRACK ADDRESS
```

```
01585A 0BC4 97 0E    A        STAA   TITRK    AND SAVE IT (DURING PROGRAM LOAD)
01586A 0BC6 DE 04    A        LDX    ISIZE    TAKE CURRENT DEFINED INPUT FILE SIZE
01587A 0BC8 DF 0F    A        STX    TISZE    AND SAVE IT (DURING PROGRAM LOAD)
01588A 0BCA 96 03    A        LDAA   IUNIT    TAKE THE CURRENT DEFINED UNIT ID
01589A 0BCC B7 0156  A        STAA   TIUNT    AND SAVE IT (DURING PROGRAM LOAD)
01590A 0BCF 39                RTS             RETURN SAVED INPUT DEFINITIONS TO CALLER
01591                  *
01592A 0BD0 86 FF    A SVIO1  LDAA   #$FF     SET FLAG VALUE = $FF (CLOSE FILE)
01593A 0BD2 97 01    A        STAA   OFILE    COMMANDING EDOS TO CLOSE FILE & UPDATE
01594A 0BD4 39                RTS             THE DIRECTORY WHEN EDOS RELOADS
01595                  *
01596                  *
01597                  *
01598A 0BD5 BD 0B6E  A        JSR    NONAM    THIS CALL IS NOT USED
01599                  *
01600                  *
01601A 0BD8 BD 0B81  A SVIOF  JSR    DOOFL    PARSE COMMAND FOR OUTPUT FILE
01602A 0BDB BD 0BAC  A        JSR    DEFOP    SET UP FILE DEFINITION LOCATIONS TO OUTPUT FILE
01603                  *
01604A 0BDE BD 0BC2  A SVIPF  JSR    MOVIP    SAVE INPUT FILE DEFINITIONS DURING PROGRAM LOAD
01605A 0BE1 20 ED 0BD0        BRA    SVIO1    FLAG REQUIRED OUTPUT FILE CLOSURE
01606                  *
01607                  *
01608                  *THESE ROUTINES GENERATE DISK FILES FROM:
01609                  *
01610                  *CASSETTE
01611                  *
01612                  *HS PAPER TAPE READER
01613                  *
01614                  *TELETYPE READER
01615                  *
01616          0BE3  A CTAPE  EQU    *
01617          0BE3  A PTAPE  EQU    *
01618          0BE3  A TTAPE  EQU    *
01619A 0BE3 BD 0BD8  A        JSR    SVIOF    VALIDATE NEW FILE NAME, & ALLOCATE
01620A 0BE6 BD 0C2E  A        JSR    GENDV    SWITCH TO SELECTED TERMINAL, INPUT DATA & WRITE IT TO DISK
01621A 0BE9 7E 0692  A        JMP    FLCLS    CLOSE THE NEW FILE, WAIT FOR NEXT COMMAND
01622                  *
01623                  *THESE ROUTINES GENERATE A COPY OF A NEW SYSTEM FROM:
01624                  *
01625                  *
01626                  *CASSETTE
01627                  *
01628                  *HS PAPER TAPE READER
01629                  *
01630                  *TELETYPE READER
01631                  *
01632          0BEC  A PXGEN  EQU    *
01633          0BEC  A CXGEN  EQU    *
01634          0BEC  A TXGEN  EQU    *
01635A 0BEC 86 01    A        LDAA   #1       SELECT TRACK 1
01636A 0BEE 97 0B    A        STAA   OTRK     FOR THE SYSTEM AREA
01637A 0BF0 97 0C    A        STAA   OSCTR    BEGIN AT SECTOR 1
01638A 0BF2 7F 0002  A        CLR    OUNIT    USE THE SYSTEM DISK
01639A 0BF5 7F 000D  A        CLR    OCNTR    CLEAR THE O/P BYTE COUNT
01640A 0BF8 CE 0082  A        LDX    #SYSSZ   ALLOCATE SYSSZ SECTORS
01641A 0BFB DF 09    A        STX    OSIZE    FOR THE NEW SYSTEM AREA
01642A 0BFD BD 0C2E  A        JSR    GENDV    SWITCH TO SELECTED TERMINAL, INPUT DATA & WRITE IT TO DISK
```

```
01643A 0C00 86 53    A        LDAA   #'S      DEFINE FILE BLOCK "S"
01644A 0C02 BD E80C  A        JSR    XWRT     AND WRITE IT TO DISK
01645A 0C05 86 1B    A        LDAA   #ESC     GENERATE END OF OBJECT FILE MARK (ESC CHAR)
01646A 0C07 BD E80C  A        JSR    XWRT     AND WRITE IT TO DISK
01647A 0C0A 86 14    A        LDAA   #DC4     COMMAND "PUNCH OFF" IN THE FILE TO TERMINATE DUMPS
01648A 0C0C BD E80C  A        JSR    XWRT     WRITE THE DC4 TO THE DISK
01649A 0C0F 7D 000D  A GENS1  TST    OCNTR    IF DISK DATA BUFFER IS NOT FULL (128 BYTES)
01650A 0C12 27 06 0C1A        BEQ    GENS2    THEN
01651A 0C14 4F                CLRA            PAD WITH NULLS
01652A 0C15 BD E80C  A        JSR    XWRT     WRITING TO DISK
01653A 0C18 20 F5 0C0F        BRA    GENS1    UNTIL BUFFER IS FULL
01654                 *
01655A 0C1A 86 03    A GENS2  LDAA   #3       DEFINE SECTOR 3
01656A 0C1C B7 0150  A        STAA   USECT    AS THE CURRENT SECTOR
01657A 0C1F 86 01    A        LDAA   #1       DEFINE SECTOR 1
01658A 0C21 B7 0152  A        STAA   CRSEC    AS THE STARTING SECTOR
01659A 0C24 96 0B    A        LDAA   OTRK     DEFINE THE START TRACK
01660A 0C26 4C                INCA            AS THE CURRENT TRACK +1
01661A 0C27 B7 0151  A        STAA   CRTRK    FOR USE BY WTNDX
01662A 0C2A BD 0642  A        JSR    WTND5    GENERATE THE DIRECTORY ENTRY
01663A 0C2D 39                RTS             AND RETURN
01664                 *
01665                 *
01666                 *IDENTIFY INPUT DEVICE, & READ TO DISK
01667                 *UNTIL END OF MEDIA OCCURS.
01668                 *
01669A 0C2E 96 6C    A GENDV  LDAA   IBUF1    GET THE FIRST CHARACTER OF COMMAND
01670A 0C30 81 50    A        CMPA   #'P      IF IT IS AN ASCII "P", THEN
01671A 0C32 27 79 0CAD        BEQ    GENDA    USE HS PAPER TAPE READER I/P
01672A 0C34 7D FF02  A GEND1  TST    SPEED    IF SPEED FLAG IS NEGATIVE
01673A 0C37 2A 0C 0C45        BPL    GEND2    IT IS A TAPE CASSETTE TERMINAL
01674A 0C39 86 10    A        LDAA   #$10     LOAD THE RDC "ATTENTION"
01675A 0C3B BD F9CF  A        JSR    OCHAR    OUTPUT CHARACTER (NO NULLS)
01676A 0C3E 86 37    A        LDAA   #$37     LOAD "BLOCK FORWARD" MODE
01677A 0C40 BD F9CF  A        JSR    OCHAR    OUTPUT CHARACTER (NO NULLS)
01678A 0C43 20 0D 0C52        BRA    GEND3    INPUT RESULTING DATA STREAM
01679                 *
01680A 0C45 B6 FCFD  A GEND2  LDAA   SBITC    GET CONTROL TEMPLATE FOR ACIA
01681A 0C48 84 55    A        ANDA   #$55     MASK TO REQUEST FOR READER RELAY CONTROL
01682A 0C4A B7 FCF4  A        STAA   ACIAC    SET UP IN ACIA (RTS)
01683A 0C4D 86 11    A        LDAA   #$11     LOAD "AUTO READER ON"
01684A 0C4F BD F9CF  A        JSR    OCHAR    OUTPUT CHARACTER (NO NULLS)
01685A 0C52 CE 1200  A GEND3  LDX    #TRKBF   LOAD POINTER TO TRACK BUFFER AREA (END OF EDOS + MOD 2
01686A 0C55 DF CA    A GEND4  STX    BGBFP    SAVE WRITE BUFFER POINTER
01687A 0C57 CE FFFF  A        LDX    #TIMAX   BEGIN TIMEOUT FOR EOM
01688A 0C5A B6 FCF4  A GEND5  LDAA   ACIAS    READ ACIA STATUS REGISTER
01689A 0C5D 47                ASRA            SHIFT "DATA AVAILABLE" FLAG TO C BIT
01690A 0C5E 25 09 0C69        BCS    GEND6    IF DATA IS READY, GO GET IT
01691A 0C60 09                DEX             IF TIMEOUT HAS RUN DOWN
01692A 0C61 26 F7 0C5A        BNE    GEND5    THEN
01693A 0C63 86 1A    A        LDAA   #ASUB    LOAD A "SUB" CHARACTER TO SHOW EOM
01694A 0C65 DE CA    A        LDX    BGBFP    GET CURRENT BUFFER POINTER
01695A 0C67 20 10 0C79        BRA    GEND7    AND CLOSE OFF THE READ
01696                 *
01697A 0C69 DE CA    A GEND6  LDX    BGBFP    GET CURRENT BUFFER POINTER
01698A 0C6B B6 FCF5  A        LDAA   ACIAR    PICK UP THE DATA CHARACTER
01699A 0C6E 84 7F    A        ANDA   #$7F     STRIP OFF THE PARITY BIT
01700A 0C70 81 0D    A        CMPA   #CR      IF IT IS AN ASCII <CR>, CYCLE THE TTY
```

```
01701A 0C72 27 05 0C79        BEQ    GEND7    ELSE,
01702A 0C74 A7 00    A        STAA   0,X      STORE THE CHARACTER IN THE BUFFER
01703A 0C76 08                INX            MOVE ON TO THE NEXT FREE BUFFER LOCATION
01704A 0C77 20 DC 0C55        BRA    GEND4    AND CONTINUE READING
01705                 *
01706A 0C79 A7 00    A GEND7  STAA   0,X      STORE THE <CR> OR <SUB>(EOM) IN THE BUFFER
01707A 0C7B 08                INX            MOVE ON TO THE NEXT FREE BUFFER LOCATION
01708A 0C7C 86 FF    A        LDAA   #$FF     GENERATE AN END-OF-LINE MARKER
01709A 0C7E A7 00    A        STAA   0,X      IN THE BUFFER
01710A 0C80 B6 FCFD  A        LDAA   SBITC    GET CONTROL TEMPLATE FOR ACIA
01711A 0C83 84 35    A        ANDA   #$35     TURN OFF RELAY CONTROLLED READER
01712A 0C85 B7 FCF4  A        STAA   ACIAC    BY DROPPING RTS SIGNAL
01713A 0C88 86 13    A        LDAA   #$13     SIGNAL "READER OFF" OR CYCLE THE TTY
01714A 0C8A BD F9CF  A        JSR    OCHAR    OUTPUT CHARACTER (NO NULLS)
01715A 0C8D B6 FCF5  A        LDAA   ACIAR    REMOVE BAD CHARACTERS FROM ACIA INPUT BUFFER
01716A 0C90 B6 FCF5  A        LDAA   ACIAR    DUE TO READER SHUTDOWN ON TTY
01717                 *
01718                 *HAVING A LOAD OF CHARACTERS IN THE BUFFER
01719                 *WRITE THEM TO THE DISK, WITH READER OFF
01720                 *
01721A 0C93 CE 1200  A        LDX    #TRKBF   DEFINE THE START OF THE DATA BUFFER
01722A 0C96 DF CA    A        STX    BGBFP    SET THE BUFFER POINTER TO THE START OF THE BUFFER
01723A 0C98 DE CA    A GEND8  LDX    BGBFP    PICK UP THE CURRENT BUFFER POINTER
01724A 0C9A A6 00    A        LDAA   0,X      PICK UP A DATA CHARACTER
01725A 0C9C 08                INX            MOVE ON TO THE NEXT CHARACTER
01726A 0C9D DF CA    A        STX    BGBFP    SAVE THE BUFFER POINTER
01727A 0C9F 81 FF    A        CMPA   #$FF     AT END OF LINE, START THE READER AGAIN
01728A 0CA1 27 91 0C34        BEQ    GEND1    ELSE,
01729A 0CA3 81 1A    A        CMPA   #ASUB    AT EOM OR TIMEOUT, EXIT
01730A 0CA5 27 05 0CAC        BEQ    GEND9    ELSE,
01731A 0CA7 BD E80C  A        JSR    XWRT     WRITE THE CHARACTER TO DISK
01732A 0CAA 20 EC 0C98        BRA    GEND8    WHILE CHARACTERS IN THE BUFFER, CONTINUE
01733                 *
01734A 0CAC 39          GEND9  RTS            RETURN TO CALLER
01735                 *
01736A 0CAD BD EB04  A GENDA  JSR    INITR    INITIALIZE HS PAPER TAPE READER
01737A 0CB0 BD 0036  A GENDB  JSR    PTAPV    GET DATA TO THE END OF TAPE
01738A 0CB3 25 F7 0CAC        BCS    GEND9    WHILE MORE TAPE
01739A 0CB5 BD E80C  A        JSR    XWRT     WRITE DATA TO DISK
01740A 0CB8 20 F6 0CB0        BRA    GENDB    CONTINUE
01741                 *
01742                 *
01743          0CBA  A TDUMP  EQU    *
01744          0CBA  A CDUMP  EQU    *
01745A 0CBA BD 0360  A        JSR    PRSFL    PARSE THE INPUT FILE NAME
01746A 0CBD BD 052C  A        JSR    DFINF    IF THE REQUESTED FILE
01747A 0CC0 4D                TSTA           IS NOT FOUND
01748A 0CC1 27 03 0CC6        BEQ    DUMP1    THEN
01749A 0CC3 7E 06E6  A        JMP    NOSFL    PRINT "NO SUCH FILE" & WAIT FOR NEW COMMAND
01750A 0CC6 96 6C    A DUMP1  LDAA   IBUF1    IF FIRST CHARACTER OF COMMAND
01751A 0CC8 81 43    A        CMPA   #'C      IS NOT A "C" (CASSETTE DUMP)
01752A 0CCA 27 2D 0CF9        BEQ    CDMP1    THEN
01753A 0CCC BD 0D71  A        JSR    NULOC    OUTPUT 100 NULLS (10" LEADER ON TAPE)
01754A 0CCF BD 0D71  A        JSR    NULOC    OUTPUT 100 NULLS (20" LEADER ON TAPE)
01755A 0CD2 BD E809  A TDMP1  JSR    XRI      READ A BYTE FROM DISK
01756A 0CD5 25 09 0CE0        BCS    TDMP2    IF NOT END OF FILE, AND
01757A 0CD7 81 0D    A        CMPA   #CR      IF NOT A <CR>
01758A 0CD9 27 15 0CF0        BEQ    TDMP3    THEN
```

```
01759A 0CDB BD 0033  A        JSR    PRICV    PUNCH THIS CHARACTER
01760A 0CDE 20 F2 0CD2        BRA    TDMP1    AND CONTINUE
01761                 *
01762A 0CE0 86 1A    A TDMP2  LDAA   #ASUB    LOAD ASCII "SUB" (EOM MARK)
01763A 0CE2 BD 0033  A        JSR    PRICV    AND PUNCH IT TO TAPE
01764A 0CE5 86 0D    A        LDAA   #CR      LOAD AN ASCII <CR>
01765A 0CE7 BD 0033  A        JSR    PRICV    AND PUNCH IT TO TAPE
01766A 0CEA BD 0D71  A        JSR    NULOC    OUTPUT 100 NULLS (10" TRAILER)
01767A 0CED 7E 016E  A        JMP    WTCMD    WAIT FOR NEW COMMAND
01768                 *
01769A 0CF0 BD 0033  A TDMP3  JSR    PRICV    PUNCH THE <CR>
01770A 0CF3 4F                CLRA            DEFINE A NULL (ALLOW FOR PUNCH ON/OFF)
01771A 0CF4 BD 0033  A        JSR    PRICV    AND PUNCH IT
01772A 0CF7 20 D9 0CD2        BRA    TDMP1    CONTINUE PUNCHING
01773                 *
01774                 *
01775A 0CF9 7F 0158  A CDMP1  CLR    EOFFL    CLEAR END-OF-FILE FLAG FOR CASSETTE WRITE
01776A 0CFC 8D 0E 0D0C CDMP2  BSR    DUMRD    READ A RECORD TERMINATED BY EOF OR <CR>
01777A 0CFE 7D 0158  A        TST    EOFFL    IF NOT END OF FILE YET
01778A 0D01 26 04 0D07        BNE    CDMP3    THEN
01779A 0D03 8D 2C 0D31        BSR    DUMWT    WRITE THIS RECORD TO THE CASSETTE
01780A 0D05 20 F5 0CFC        BRA    CDMP2    CONTINUE
01781                 *
01782A 0D07 8D 28 0D31 CDMP3  BSR    DUMWT    WRITE THE LAST (EOF) RECORD TO CASSETTE
01783A 0D09 7E 016E  A        JMP    WTCMD    WAIT FOR NEXT COMMAND
01784                 *
01785                 *
01786                 *THIS SUBROUTINE READS A DISK FILE UNTIL <CR> OR EOF
01787                 *
01788A 0D0C CE 1200  A DUMRD  LDX    #TRKBF   DEFINE A POINTER TO A LARGE
01789A 0D0F DF CA    A        STX    BGBFP    BUFFER WORK AREA
01790                 *
01791A 0D11 BD E809  A DUMR1  JSR    XRI      READ A BYTE FROM DISK
01792A 0D14 25 0C 0D22        BCS    DUMR2    IF NOT EOF
01793A 0D16 DE CA    A        LDX    BGBFP    USE POINTER TO
01794A 0D18 A7 00    A        STAA   $0,X     STORE CHARACTER IN BUFFER
01795A 0D1A 08                INX             TAKE THE NEXT BUFFER LOCATION
01796A 0D1B DF CA    A        STX    BGBFP    AND UPDATE THE POINTER
01797A 0D1D 81 0D    A        CMPA   #CR      IF THE CHARACTER WAS NOT <CR>
01798A 0D1F 26 F0 0D11        BNE    DUMR1    CONTINUE
01799A 0D21 39                RTS             ELSE RETURN
01800                 *
01801A 0D22 86 1A    A DUMR2  LDAA   #ASUB    LOAD AN ASCII "SUB" AS EOM MARK
01802A 0D24 DE CA    A        LDX    BGBFP    USING THE BUFFER POINTER
01803A 0D26 A7 00    A        STAA   $0,X     STORE "SUB" IN THE BUFFER
01804A 0D28 08                INX             TAKE THE NEXT BUFFER LOCATION
01805A 0D29 86 0D    A        LDAA   #CR      LOAD AN ASCII <CR>
01806A 0D2B A7 00    A        STAA   $0,X     STORE "CR" IN THE BUFFER
01807A 0D2D 7C 0158  A        INC    EOFFL    SET END-OF-FILE FLAG
01808A 0D30 39                RTS             AND RETURN
01809                 *
01810                 *
01811                 *THIS SUBROUTINE WRITES A RECORD TO CASSETTE
01812                 *RETURN IS DELAYED UNTIL RECORD IS WRITTEN
01813                 *
01814A 0D31 0D          DUMWT  SEC             SET THE CARRY BIT TO "1"
01815A 0D32 79 FF62  A        ROL    EXPUN    SET THE EXBUG PUNCH FLAG BIT 0
01816A 0D35 86 10    A        LDAA   #DLE     LOAD A R.O.C. "ATTENTION" CHAR FOR CASSETTE TERMINAL
```

```
01817A 0D37 BD F9CF  A        JSR    OCHAR    SEND IT TO TERMINAL (NO NULLS)
01818A 0D3A 86 30    A        LDAA   #'0      DEFINE "PRINTER OFF" COMMAND
01819A 0D3C BD F9CF  A        JSR    OCHAR    SEND IT TO TERMINAL (NO NULLS)
01820A 0D3F 86 FF    A        LDAA   #RUB     DEFINE A RUBOUT CHAR (TIME FILLER)
01821A 0D41 BD F9CF  A        JSR    OCHAR    SEND IT TO TERMINAL (NO NULLS)
01822A 0D44 86 12    A        LDAA   #DC2     DEFINE "CASSETTE WRITE" COMMAND
01823A 0D46 BD F9CF  A        JSR    OCHAR    SEND IT TO TERMINAL (NO NULLS)
01824A 0D49 CE 1200  A        LDX    #TRKBF   SET POINTER AT BEGINNING OF BUFFER
01825A 0D4C A6 00    A DUMW1  LDAA   $0,X     PICK UP A CHARACTER
01826A 0D4E BD F018  A        JSR    PRINC    WRITE IT TO THE CASSETTE TAPE
01827A 0D51 A6 00    A        LDAA   $0,X     LOAD THE CHARACTER AGAIN
01828A 0D53 08                INX             TAKE THE NEXT BUFFER POSITION
01829A 0D54 81 0D    A        CMPA   #CR      IF THE CHARACTER WAS NOT <CR>, THEN
01830A 0D56 26 F4 0D4C        BNE    DUMW1    CONTINUE, ELSE
01831                   *
01832A 0D58 86 14    A        LDAA   #DC4     DEFINE "CASSETTE OFF" COMMAND
01833A 0D5A BD F9CF  A        JSR    OCHAR    SEND IT TO TERMINAL (NO NULLS)
01834A 0D5D 86 10    A        LDAA   #DLE     LOAD A R.O.C. "ATTENTION" CHAR FOR CASSETTE TERMINAL
01835A 0D5F BD F9CF  A        JSR    OCHAR    SEND IT TO TERMINAL (NO NULLS)
01836A 0D62 86 39    A        LDAA   #'9      DEFINE "PRINTER ON" COMMAND
01837A 0D64 BD F9CF  A        JSR    OCHAR    SEND IT TO TERMINAL (NO NULLS)
01838A 0D67 7F FF62  A        CLR    EXPUN    CLEAR THE PUNCH ON CONTROL FLAG
01839                   *
01840                   *NOW WAIT FOR CASSETTE TO WRITE LAST RECORD & STOP
01841                   *
01842A 0D6A CE 5DC0  A        LDX    #24000   DEFINE A COUNT OF 24000, DECIMAL
01843A 0D6D 09          WAIT1  DEX             COUNT DOWN TO 0, @ 8 USEC PER COUNT
01844A 0D6E 26 FD 0D6D        BNE    WAIT1    WHILE COUNT, CONTINUE (200 MILLISECONDS)
01845A 0D70 39                RTS             AFTER TIMED WAIT, RETURN TO CALLER
01846                   *
01847                   *
01848                   *
01849                   *THIS ROUTINE OUTPUTS 100 DECIMAL NULLS TO THE SWITCHED PRINT DEVICE
01850                   *
01851A 0D71 C6 64    A NULOC  LDAB   #100     DEFINE A COUNT OF
01852A 0D73 D7 C9    A        STAB   NULC1    100 NULLS TO BE O/P
01853A 0D75 4F          NULO1  CLRA            PRODUCE A NULL CHARACTER
01854A 0D76 BD 0033  A        JSR    PRICV    USING SWITCH VECTOR, PRINT IT
01855A 0D79 7A 00C9  A        DEC    NULC1    WHILE 100 NULLS
01856A 0D7C 26 F7 0D75        BNE    NULO1    CONTINUE TO OUTPUT NULLS
01857A 0D7E 39                RTS             RETURN TO CALLER
01858                   *
01859                   *
01860                   *
01861                   *
01862                   *DUPLICATE CONTENTS OF DISK DRIVE 0 TO DISK DRIVE 1  EACH TRACK
01863                   *IS COPIED IN 3 DISK REVOLUTIONS, STORED IN 3 REVOLUTIONS AND
01864                   *CHECK READ FOR CRC IN 2 REVOLUTIONS. BUFFER USED IS TRKBF
01865                   *
01866          0D7F  A DUP    EQU    *
01867A 0D7F 86 01    A        LDAA   #$1      START ON SECTOR #1
01868A 0D81 97 07    A        STAA   ISCTR    FOR THE INPUT FILE
01869A 0D83 7F 0006  A        CLR    ITRK     START ON TRACK 0
01870A 0D86 86 41    A        LDAA   #$41     SELECT UNIT 1, SECTOR 1
01871A 0D88 97 0C    A        STAA   OSCTR    FOR THE OUTPUT FILE
01872A 0D8A 7F 000B  A        CLR    OTRK     START ON TRACK 0
01873A 0D8D 96 07    A DUP1   LDAA   ISCTR    PICK UP INITIAL INPUT SECTOR
01874A 0D8F BD EA22  A        JSR    XUSP     TRANSMIT PHYSICAL <UNIT><SECTOR> TO DISK
```

```
01875A 0D92 BD 0468  A        JSR    DSKRD    IF DISK IS READY, THEN
01876A 0D95 96 06    A        LDAA   ITRK     PICK UP INITIAL INPUT TRACK
01877A 0D97 BD EA2B  A        JSR    SEEK     SEEK DISK 0 TO TRACK 0
01878A 0D9A CE 1200  A        LDX    #TRKBF   DEFINE START OF TRACK BUFFER
01879A 0D9D DF CA    A        STX    BGBFP    IN THE TRACK BUFFER POINTER
01880A 0D9F 96 07    A DUP2   LDAA   ISCTR    PICK UP CURRENT SECTOR FOR INPUT
01881A 0DA1 BD EA22  A        JSR    XUSP     TRANSMIT PHYSICAL <UNIT><SECTOR> TO DISK
01882A 0DA4 DE CA    A        LDX    BGBFP    PICK UP THE CURRENT TRACK BUFFER POINTER
01883A 0DA6 BD 0E31  A        JSR    DUSRD    READ A SECTOR INTO THE TRACK BUFFER
01884A 0DA9 DF CA    A        STX    BGBFP    SAVE THE CURRENT TRACK BUFFER POINTER
01885A 0DAB 86 03    A        LDAA   #$3      INTERLEAVE SECTORS BY 3 (ALLOW PROCESSING TIME)
01886A 0DAD 9B 07    A        ADDA   ISCTR    TO AVOID A 1 REVOLUTION WAIT BETWEEN SECTORS
01887A 0DAF 97 07    A        STAA   ISCTR    (MAINTAIN MINIMUM DISK ACCESS DELAY TIME)
01888A 0DB1 C6 02    A        LDAB   #$2      ASSUME REV 1, START REV 2 @ SECTOR 2
01889A 0DB3 81 1C    A        CMPA   #28      IF NEXT SECTOR IS 28, END OF REV 1
01890A 0DB5 27 12 0DC9        BEQ    DUP3     THEN START REV 2
01891A 0DB7 5C                INCB            ELSE ASSUME REV 2, START REV 3 @ SECTOR 3
01892A 0DB8 81 1D    A        CMPA   #29      IF NEXT SECTOR IS 29, END OF REV 2
01893A 0DBA 27 0D 0DC9        BEQ    DUP3     THEN START REV 3
01894A 0DBC 81 1B    A        CMPA   #27      IF NEXT SECTOR IS 27, END OF REV 3
01895A 0DBE 26 DF 0D9F        BNE    DUP2     THEN
01896A 0DC0 7C 0006  A        INC    ITRK     TAKE THE NEXT TRACK
01897A 0DC3 86 01    A        LDAA   #$1      START ON SECTOR 1
01898A 0DC5 97 07    A        STAA   ISCTR    OF THAT TRACK
01899A 0DC7 20 04 0DCD        BRA    DUP4     WRITE THE TRACK JUST READ
01900                  *
01901A 0DC9 D7 07    A DUP3   STAB   ISCTR    START REV 2 OR 3
01902A 0DCB 20 D2 0D9F        BRA    DUP2     READ THE INTERLEAVED SECTORS
01903                  *
01904A 0DCD 86 05    A DUP4   LDAA   #$5      DEFINE A WRITE RETRY COUNT
01905A 0DCF 97 C9    A        STAA   WRTRY    5 ATTEMPTS TO WRITE THE TRACK
01906A 0DD1 CE 1200  A DUP5   LDX    #TRKBF   START THE BUFFER POINTER
01907A 0DD4 DF CA    A        STX    BGBFP    AT THE BEGINNING OF THE TRACK BUFFER
01908A 0DD6 96 0C    A        LDAA   OSCTR    PICK UP INITIAL <UNIT><SECTOR>
01909A 0DD8 BD EA22  A        JSR    XUSP     TRANSMIT PHYSICAL <UNIT><SECTOR> TO DISK
01910A 0DDB BD 0468  A        JSR    DSKRD    IF DISK IS READY, THEN
01911A 0DDE 96 0B    A        LDAA   OTRK     PICK UP INITIAL TRACK
01912A 0DE0 BD EA2B  A        JSR    SEEK     SEEK DRIVE 1 TO TRACK 0, SECTOR 1
01913A 0DE3 96 0C    A DUP6   LDAA   OSCTR    PICK UP CURRENT <UNIT><SECTOR>
01914A 0DE5 BD EA22  A        JSR    XUSP     TRANSMIT PHYSICAL <UNIT><SECTOR> TO DISK
01915A 0DE8 DE CA    A        LDX    BGBFP    PICK UP THE CURRENT TRACK BUFFER POINTER
01916A 0DEA BD 0E83  A        JSR    DUWTS    WRITE A SECTOR TO DISK
01917A 0DED DF CA    A        STX    BGBFP    SAVE THE CURRENT TRACK BUFFER POINTER
01918A 0DEF 86 03    A        LDAA   #$3      INTERLEAVE SECTORS BY 3 (ALLOW PROCESSING TIME)
01919A 0DF1 9B 0C    A        ADDA   OSCTR    TO AVOID A 1 REVOLUTION WAIT BETWEEN SECTORS
01920A 0DF3 97 0C    A        STAA   OSCTR    (MAINTAIN MINIMUM DISK ACCESS DELAY TIME)
01921A 0DF5 C6 42    A        LDAB   #$42     ASSUME REV 1, START REV 2 @ SECTOR 2
01922A 0DF7 81 5C    A        CMPA   #28+$40  IF NEXT SECTOR IS 28, END OF REV 1
01923A 0DF9 27 0F 0E0A        BEQ    DUP7     THEN START REV 2
01924A 0DFB 5C                INCB            ELSE ASSUME REV 2, START REV 3 @ SECTOR 3
01925A 0DFC 81 5D    A        CMPA   #29+$40  IF NEXT SECTOR IS 29, END OF REV 2
01926A 0DFE 27 0A 0E0A        BEQ    DUP7     THEN START REV 3
01927A 0E00 81 5B    A        CMPA   #27+$40  IF NEXT SECTOR IS 27, END OF REV 3
01928A 0E02 26 DF 0DE3        BNE    DUP6     THEN
01929A 0E04 86 41    A        LDAA   #$41     START <UNIT><SECTOR> AT
01930A 0E06 97 0C    A        STAA   OSCTR    DRIVE 1, SECTOR 0 AGAIN
01931A 0E08 20 04 0E0E        BRA    DUP8     CHECK CRC ON TRACK JUST WRITTEN
01932                  *
```

```
01933A 0E0A D7 0C    A DUP7   STAB   OSCTR    START REV 2 OR 3
01934A 0E0C 20 D5 0DE3        BRA    DUP6     WRITE INTERLEAVED SECTORS
01935                 *
01936A 0E0E BD 0EA0  A DUP8   JSR    DUCRC    CHECK READ FOR CRC, IF FAILURE
01937A 0E11 27 0F 0E22        BEQ    DUP9     THEN
01938A 0E13 86 41    A        LDAA   #$41     START <UNIT><SECTOR> AT
01939A 0E15 97 0C    A        STAA   OSCTR    DRIVE 1, SECTOR 0 AGAIN
01940A 0E17 BD EA38  A        JSR    RFLAG    RESET ERROR FLAG ON DISK I/F
01941A 0E1A 7A 00C9  A        DEC    WRTRY    DECREMENT WRITE RETRY COUNTER, IF TRIES RUN OUT
01942A 0E1D 26 B2 0DD1        BNE    DUP5     THEN (ELSE TRY TO WRITE THE TRACK AGAIN)
01943A 0E1F 7E 042E  A        JMP    MEDER    PRINT "MEDIA ERROR" & WAIT FOR NEW COMMAND
01944                 *
01945A 0E22 7C 000B  A DUP9   INC    OTRK     TAKE THE NEXT TRACK
01946A 0E25 96 0B    A        LDAA   OTRK     IF THE NEXT TRACK
01947A 0E27 81 4D    A        CMPA   #77      IS (LAST TRACK)+1
01948A 0E29 26 03 0E2E        BNE    DUPA     THEN
01949A 0E2B 7E 016E  A        JMP    WTCMD    WAIT FOR NEXT COMMAND
01950                 *
01951A 0E2E 7E 0D8D  A DUPA   JMP    DUP1     ELSE START ON THE NEXT TRACK
01952                 *
01953                 *
01954                 *READ A DISK SECTOR TO TRACK INPUT BUFFER,  IF A READ ERROR
01955                 *PERSISTS, USE THE DATA READ ON THE FIFTH TRY
01956                 *
01957A 0E31 86 05    A DUSRD  LDAA   #$5      DEFINE A RETRY COUNT OF
01958A 0E33 B7 0155  A        STAA   TRYCT    5 TRIES TO READ DATA WITH ERRORS
01959A 0E36 86 02    A DUSR1  LDAA   #$2      DEFINE DISK READ COMMAND
01960A 0E38 BD EA3E  A        JSR    LOOP     ISSUE COMMAND TO DISK (LOOP ON BUSY)
01961A 0E3B B6 EC00  A        LDAA   DKDID    READ DISK STATUS
01962A 0E3E 84 08    A        ANDA   #$8      IF CRC ERROR
01963A 0E40 27 08 0E4A        BEQ    DUSR2    THEN
01964A 0E42 BD EA38  A        JSR    RFLAG    RESET ERROR FLAG ON DISK I/F
01965A 0E45 7A 0155  A        DEC    TRYCT    RECORD FAILED READ TRY, WHILE COUNT
01966A 0E48 26 EC 0E36        BNE    DUSR1    TRY AGAIN, ELSE
01967A 0E4A 86 04    A DUSR2  LDAA   #$4      DEFINE WRITE DATA COMMAND
01968A 0E4C A7 00    A        STAA   $0,X     ASSUME USABLE SECTOR (NOT "DD" MARK)
01969A 0E4E B6 EC00  A        LDAA   DKDID    IF THIS SECTOR READ AS "DD"
01970A 0E51 84 80    A        ANDA   #$80     INDICATED ON "DD" STATUS BIT
01971A 0E53 27 04 0E59        BEQ    DUSR3    THEN
01972A 0E55 86 0E    A        LDAA   #$E      DEFINE WRITE "DD" COMMAND
01973A 0E57 A7 00    A        STAA   $0,X     PASS COMMAND TO WRITE ROUTINE
01974A 0E59 08          DUSR3  INX            STEP OVER THE SECTOR COMMAND BYTE IN THE BUFFER
01975A 0E5A 86 80    A        LDAA   #128     DEFINE BYTE COUNT FOR A SECTOR = 128
01976A 0E5C B7 0155  A        STAA   BYTCT    INITIALIZE READ DATA BYTE COUNTER
01977A 0E5F C6 40    A        LDAB   #$40     DEFINE READ DATA COMMAND
01978A 0E61 86 3C    A DUSR4  LDAA   #$3C     DEFINE COMMAND CONTROL
01979A 0E63 B7 EC03  A        STAA   DKCOC    ISSUE COMMAND CONTROL
01980A 0E66 F7 EC02  A        STAB   DKCOD    ISSUE COMMAND DEFINED IN B REG (READ DATA)
01981A 0E69 B6 EC00  A        LDAA   DKDID    LOAD DATA READ FROM DISK BUFFER
01982A 0E6C A7 00    A        STAA   $0,X     STORE DATA IN TRKBF
01983A 0E6E 08                INX            TAKE THE NEXT LOCATION IN TRKBF
01984A 0E6F 86 2C    A        LDAA   #$2C     DEFINE RESET DRIVE COMMAND
01985A 0E71 B7 EC03  A        STAA   DKCOC    ISSUE RESET DRIVE
01986A 0E74 F7 EC02  A        STAB   DKCOD    ISSUE COMMAND DEFINED IN B REG (READ DATA)
01987A 0E77 7F EC02  A        CLR    DKCOD    CLEAR THE COMMAND REGISTER
01988A 0E7A 7A 0155  A        DEC    BYTCT    COUNT OFF THIS BYTE, WHILE 128 BYTES
01989A 0E7D 26 E2 0E61        BNE    DUSR4    CONTINUE
01990A 0E7F BD EA38  A        JSR    RFLAG    RESET ERROR FLAG ON DISK I/F
```

```
01991A 0E82 39               RTS             RETURN TO CALLER
01992                *
01993                *
01994                *WRITE A SECTOR TO DISK, USING DATA FROM THE CURRENT SECTOR
01995                *
01996A 0E83 A6 00    A DUWTS  LDAA   $0,X     PICK UP SECTOR WRITE CONTROL BYTE
01997A 0E85 36               PSHA            SAVE IT TO USE AS A WRITE COMMAND AFTER BUFFER LOAD
01998A 0E86 08               INX             STEP OVER THE SECTOR WRITE CONTROL BYTE
01999A 0E87 86 80    A       LDAA   #128     DEFINE 128 BYTES IN A SECTOR
02000A 0E89 B7 0155  A       STAA   BYTCT    FOR A WRITE BYTE COUNTER
02001A 0E8C C6 30    A       LDAB   #$30     DEFINE A WRITE DISK COMMAND
02002A 0E8E A6 00    A DUWT1  LDAA   $0,X     PICK UP CURRENT DATA BYTE FROM BUFFER
02003A 0E90 B7 EC06  A       STAA   DKDOD    OUTPUT IT TO THE DISK
02004A 0E93 F7 EC02  A       STAB   DKCOD    STROBE THE DATA IN TO THE DISK BUFFER
02005A 0E96 08               INX             TAKE THE NEXT DATA BYTE
02006A 0E97 7A 0155  A       DEC    BYTCT    WHILE 128 BYTES
02007A 0E9A 26 F2 0E8E       BNE    DUWT1    CONTINUE
02008A 0E9C 32               PULA            GET SECTOR WRITE CONTROL BYTE
02009A 0E9D 7E EA3E  A       JMP    LOOP     ISSUE COMMAND TO DISK (LOOP ON BUSY)
02010                *AND RETURN TO CALLER



02012                *READ A TRACK & CHECK FOR CRC ERRORS
02013                *
02014A 0EA0 96 0C    A DUCRC  LDAA   OSCTR    PICK UP CURRENT SECTOR
02015A 0EA2 BD EA22  A       JSR    XUSP     TRANSMIT PHYSICAL <UNIT><SECTOR> TO DISK
02016A 0EA5 86 06    A       LDAA   #$6      DEFINE READ DISK FOR CRC COMMAND
02017A 0EA7 BD EA3E  A       JSR    LOOP     ISSUE COMMAND TO DISK (LOOP ON BUSY)
02018A 0EAA B6 EC00  A       LDAA   DKDID    READ DISK STATUS
02019A 0EAD 84 08    A       ANDA   #$8      IF CRC ERROR NOT ON
02020A 0EAF 26 19 0ECA       BNE    DUCR2    THEN (FAILURE EXIT, Z = 0)
02021A 0EB1 86 02    A       LDAA   #$2      INTERLEAVE SECTORS BY TWO DURING
02022A 0EB3 9B 0C    A       ADDA   OSCTR    CRC READ PASS (LESS CPU TIME)
02023A 0EB5 97 0C    A       STAA   OSCTR    DEFINE NEXT SECTOR
02024A 0EB7 81 5B    A       CMPA   #27+$40  IF END OR PAST PASS 1
02025A 0EB9 2B E5 0EA0       BMI    DUCRC    THEN
02026A 0EBB 81 5C    A       CMPA   #28+$40  IF NOT END OF PASS 2
02027A 0EBD 27 06 0EC5       BEQ    DUCR1    THEN
02028A 0EBF 86 42    A       LDAA   #$42     DEFINE SECTOR 2 ON UNIT 1
02029A 0EC1 97 0C    A       STAA   OSCTR    FOR PASS 2
02030A 0EC3 20 DB 0EA0       BRA    DUCRC    BEGIN PASS 2 (READ INTERLEAVED SECTORS)
02031                *
02032A 0EC5 86 41    A DUCR1  LDAA   #$41     DEFINE SECTOR 1 ON UNIT 1
02033A 0EC7 97 0C    A       STAA   OSCTR    FOR NEXT FUNCTION
02034A 0EC9 4F               CLRA            SET Z = 1 (SUCCESS EXIT)
02035A 0ECA 39         DUCR2  RTS             RETURN TO CALLER
02036                *
02037                *
02038                *
02039A 0ECB    0001  A NULCT  RMB    1
02040                *
02041                *
02042                *THIS FUNCTION IS A GENERAL FILE DUPLICATE OR MERGE FACILITY
02043                *
02044          0ECC  A MERGE  EQU    *
02045A 0ECC BD 0BD8  A       JSR    SVIOF    CREATE A NAMED FCB, ATTR = FF, SIZE = FREE SPACE
02046A 0ECF BD 0360  A MERG1  JSR    PRSFL    PARSE THE NEXT FILE NAME & UNIT I/D
```

```
02047A 0ED2 96 C0    A        LDAA   FLNM0    IF THE FILE IS DEFINED
02048A 0ED4 81 20    A        CMPA   #SPC     (FIRST SPACE IS OVERWRITTEN)
02049A 0ED6 27 0E 0EE6        BEQ    MERG3    THEN
02050A 0ED8 BD 052C  A        JSR    DFINF    LOOK FOR THIS FILE
02051A 0EDB 4D                TSTA            IF IT DOES NOT EXIST
02052A 0EDC 27 03 0EE1        BEQ    MERG2    THEN
02053A 0EDE 7E 06E6  A        JMP    NOSFL    PRINT "NO SUCH FILE" & WAIT FOR NEW COMMAND
02054                 *
02055A 0EE1 BD 0EE9  A MERG2  JSR    MOVFL    MOVE SOURCE FILE TO DESTINATION FILE
02056A 0EE4 20 E9 0ECF        BRA    MERG1    LOOK FOR MORE SOURCE FILES
02057                 *
02058A 0EE6 7E 0692  A MERG3  JMP    FLCLS    CLOSE THE OUPUT FILE & WAIT FOR THE NEXT COMMAND
02059                 *
02060                 *
02061                 *MOVE AN INPUT FILE TO A OUTPUT FILE, VIA A LARGE BUFFER AREA
02062                 *
02063A 0EE9 7F 0158  A MOVFL  CLR    EOFFL    CLEAR "END OF FILE" FLAG
02064A 0EEC CE 1200  A MOVF1  LDX    #BGBUF   SET THE POINTER AT THE START
02065A 0EEF DF CA    A MOVF2  STX    BGBFP    OF THE LARGE BUFFER AREA
02066A 0EF1 BD E809  A        JSR    XRI      GET AN INPUT BYTE FROM THE DISK
02067A 0EF4 25 23 0F19        BCS    MOVF5    IF NOT EOF, THEN
02068A 0EF6 DE CA    A        LDX    BGBFP    LOAD THE CURRENT BUFFER POINTER
02069A 0EF8 A7 00    A        STAA   $0,X     STORE THE DATA IN THE BUFFER AREA
02070A 0EFA 08                INX             TAKE THE NEXT BUFFER LOCATION
02071A 0EFB 8C 1EFE  A        CPX    #BGBFE   IF END OF THE BUFFER
02072A 0EFE 26 EF 0EEF        BNE    MOVF2    THEN
02073                 *
02074A 0F00 DF CA    A        STX    BGBFP    SAVE THE POINTER AS A LIMIT
02075A 0F02 CE 1200  A MOVF3  LDX    #BGBUF   START AT THE BEGINNING OF THE BUFFER
02076A 0F05 DF CC    A MOVF4  STX    BFOP2    USING AN OUTPUT POINTER LOCATION
02077A 0F07 A6 00    A        LDAA   $0,X     GET THE CURRENT DATA BYTE
02078A 0F09 BD E80C  A        JSR    XWRT     WRITE THE DATA TO DISK
02079A 0F0C DE CC    A        LDX    BFOP2    LOAD THE OUTPUT POINTER
02080A 0F0E 08                INX             TAKE THE NEXT BUFFER LOCATION
02081A 0F0F 9C CA    A        CPX    BGBFP    IF REACHED PREVIOUS END POINT
02082A 0F11 26 F2 0F05        BNE    MOVF4    THEN
02083A 0F13 7D 0158  A        TST    EOFFL    IF THE END OF FILE HAS BEEN REACHED
02084A 0F16 27 D4 0EEC        BEQ    MOVF1    THEN
02085A 0F18 39                RTS             FILE IS MOVED, RETURN TO CALLER
02086                 *
02087A 0F19 7C 0158  A MOVF5  INC    EOFFL    RECORD END OF FILE FOUND
02088A 0F1C 20 E4 0F02        BRA    MOVF3    START WRITING IT
```

```
02090                   *
02091                   *
02092                   *LOCAL VARIABLE STORAGE FOR THE PURGE FUNCTION
02093                   *
02094A 0F1E    0001  A PUUID  RMB    1        SAVE LOCATION FOR DISK UNIT I/D BEING PURGED
02095A 0F1F    0002  A PUFCP  RMB    2        FILE CONTROL BLOCK STRING POINTER (FILE BEING PURGED)
02096A 0F21    0001  A PUSEC  RMB    1        CURRENT SECTOR BEING PURGED <UNIT><SECTOR>
02097A 0F22    0002  A PUFMP  RMB    2        FILE CONTROL BLOCK STRING POINTER (FILE BEING MOVED)
02098A 0F24    0001  A PUSEM  RMB    1        CURRENT SECTOR BEING MOVED <UNIT><SECTOR>
02099                   *
02100                   *
02101                   *USER PROMPT MESSAGE STRINGS FOR THE PURGE FUNCTION
02102                   *
02103A 0F25    0A    A PERMM  FCB    $0A
02104A 0F26    20    A        FCC    / PERMANENT/
02105A 0F30    07    A PURGM  FCB    $07
02106A 0F31    20    A        FCC    / PURGED/
02107A 0F38    0C    A PKDSM  FCB    $0C
02108A 0F39    50    A        FCC    /PACKING DISK/
```

```
02110                  *
02111                  *THIS FUNCTION PURGES UNWANTED FILES FROM THE DISK, AND
02112                  *PACKS THE REMAINING FILES, RECLAIMING THE SPACE
02113                  *
02114                  *PURGE IS DONE IN TWO PASSES
02115                  *
02116                  *                     PASS 1
02117                  *TEST VALIDITY OF INPUT FILE STRING, WRITE DELETED FILE
02118                  *MARKERS IN DIRECTORY FCB OF FILES TO BE PURGED.
02119                  *
02120                  *                     PASS 2
02121                  *SEARCH FOR DELETED FILE MARK (ATTRIBUTE = 80).  ON FINDING
02122                  *THE FIRST DELETED FILE MARK, SCAN FORWARD UNTIL A NON-
02123                  *DELETED FILE IS FOUND. THIS FILE IS THEN MOVED INTO THE
02124                  *AREA TO BE PURGED.  THIS PROCESS THEN CONTINUES TO THE END
02125                  *OF THE DIRECTORY. EACH TIME A FILE IS MOVED, THE DIRECTORY
02126                  *IS UPDATED.
02127                  *
02128                  *                    ERRORS
02129                  *SEVERAL ERRORS MAY OCCUR DURING THE PURGE PROCESS, SOME OF
02130                  *WHICH ARE FATAL TO ALL FILES FOLLOWING THE ERROR.  SOME
02131                  *POSSIBLE ERRORS MAY CAUSE DIRECTORY FAULTS.  PURGE SHOULD
02132                  *NOT BE USED ON A DISK THAT HAS NO BACKUP COPY.
02133                  *
02134           0F45  A PURGE  EQU    *
02135A 0F45 96 C5     A        LDAA   FLNM5    GET THE UNIT I/D TO BE PURGED
02136A 0F47 B7 0F1E   A        STAA   PUUID    AND DEFINE THE PURGE UID
02137A 0F4A BD 0360   A PURG1  JSR    PRSFL    PARSE THE NEXT FILE NAME
02138A 0F4D 96 C0     A        LDAA   FLNM0    GET THE FIRST CHARACTER
02139A 0F4F 81 20     A        CMPA   #SPC     IF IT IS A SPACE
02140A 0F51 26 03 0F56         BNE    PURG2    THEN
02141A 0F53 7E 0FF6   A        JMP    PKDSK    FILE INPUT IS COMPLETE
02142                  *
02143A 0F56 B6 0F1E   A PURG2  LDAA   PUUID    PICK UP THE PURGE UID
02144A 0F59 97 C5     A        STAA   FLNM5    AND DEFINE THIS FILE TO RESIDE ON PURGE UID DRIVE
02145A 0F5B BD 052C   A        JSR    DFINF    DEFINE THE INPUT FILE
02146A 0F5E 4D                 TSTA            IF SUCCESSFUL
02147A 0F5F 26 E9 0F4A         BNE    PURG1    THEN
02148A 0F61 A6 05     A        LDAA   $5,X     GET THE ATTRIBUTE ENTRY
02149A 0F63 81 01     A        CMPA   #$1      IF THE ATTRIBUTE IS NOT "PERMANENT" (01)
02150A 0F65 26 16 0F7D         BNE    PURG3    THEN
02151                  *
02152A 0F67 CE 00C0   A        LDX    #FLNM0   SET A POINTER TO THE FILE NAME
02153A 0F6A C6 05     A        LDAB   #$5      WITH A CHARACTER COUNT OF 5
02154A 0F6C BD 03B3   A        JSR    PRINS    PRINT OUT THE FILE NAME
02155A 0F6F CE 0F26   A        LDX    #PERMM+1 POINT TO MESSAGE STRING & SET CHAR
02156A 0F72 F6 0F25   A        LDAB   PERMM    COUNT FOR PERMANENT MESSAGE
02157A 0F75 BD 03B3   A        JSR    PRINS    PRINT " PERMANENT"
02158A 0F78 BD 03BD   A        JSR    CRLF     PRINT <CR><LF> FOR NEW LINE ON PRINTER
02159A 0F7B 20 CD 0F4A         BRA    PURG1    CONTINUE WITH NEXT FILE
02160                  *
02161A 0F7D 86 80     A PURG3  LDAA   #$80     DEFINE A "DELETED FILE" MARK
02162A 0F7F A7 05     A        STAA   $5,X     SET FILE ATTRIBUTE "DELETED" IN FCB
02163A 0F81 B6 0150   A        LDAA   USECT    PICK UP THE DIRECTORY <UNIT><SECTOR>
02164A 0F84 5F                 CLRB            SELECT TRACK 0 FOR DIRECTORY
02165A 0F85 CE 00D0   A        LDX    #SECB1   SET A POINTER TO THE SECTOR BUFFER
02166A 0F88 BD 0479   A        JSR    WRTSE    WRITE THIS SECTOR TO DISK
02167A 0F8B 4D                 TSTA            IF DIRECTORY SECTOR WRITE FAILURE
```

```
02168A 0F8C 27 03 0F91        BEQ    PURG4    THEN
02169A 0F8E 7E 042E  A        JMP    MEDER    PRINT "MEDIA ERROR" & WAIT FOR NEW COMMAND
02170                 *
02171A 0F91 CE 00C0  A PURG4  LDX    #FLNM0   SET A POINTER TO THE FILE NAME
02172A 0F94 C6 05    A        LDAB   #$5      WITH A CHARACTER COUNT OF 5
02173A 0F96 BD 03B3  A        JSR    PRINS    PRINT OUT THE FILE NAME
02174A 0F99 CE 0F31  A        LDX    #PURGM+1 POINT TO MESSAGE STRING & SET CHAR
02175A 0F9C F6 0F30  A        LDAB   PURGM    COUNT FOR PURGED MESSAGE
02176A 0F9F BD 03B3  A        JSR    PRINS    PRINT " PURGED"
02177A 0FA2 BD 03BD  A        JSR    CRLF     PRINT <CR><LF> FOR NEW LINE ON PRINTER
02178A 0FA5 20 A3 0F4A        BRA    PURG1    CONTINUE WITH NEXT FILE
02179                 *
02180                 *
02181A 0FA7 0C          DIRSE  CLC             CLEAR "C" BIT FOR SHIFTING
02182A 0FA8 B6 0F1E  A        LDAA   PUUID    PICK UP PURGE UNIT I/D
02183A 0FAB 46                RORA           SHIFT UID BITS INTO BITS 6 &7
02184A 0FAC 46                RORA           ROTATING THEM THROUGH THE C BIT
02185A 0FAD 46                RORA           BIT 7 BECOMES OLD 1, BIT 6 BECOMES OLD 0
02186A 0FAE 8A 04    A        ORAA   #$4      THE DIRECTORY STARTS AT SECTOR 4
02187A 0FB0 B7 0F24  A        STAA   PUSEM    DEFINE THE PURGE DIRECTORY <UNIT><SECTOR>
02188                 *
02189A 0FB3 CE 00D0  A DIRS1  LDX    #SECB1   LOAD POINTER FOR SECTOR BUFFER
02190A 0FB6 FF 0F22  A        STX    PUFMP    INITIALIZE THE SECTOR SCAN
02191                 *
02192A 0FB9 B6 0F24  A DIRS2  LDAA   PUSEM    GET CURRENT <UNIT><SECTOR>
02193A 0FBC 84 3F    A        ANDA   #$3F     MASK TO SECTOR ONLY (STRIP OFF UID)
02194A 0FBE 81 1B    A        CMPA   #27      IF THIS IS THE END OF TRACK (26 SECTORS/TRACK)
02195A 0FC0 26 03 0FC5        BNE    DIRS3    THEN
02196A 0FC2 86 FF    A        LDAA   #$FF     FLAG END OF DIRECTORY
02197A 0FC4 39                RTS            AND RETURN TO CALLER
02198                 *
02199A 0FC5 CE 00D0  A DIRS3  LDX    #SECB1   START AT THE BEGINNING OF THE SECTOR BUFFER
02200A 0FC8 B6 0F24  A        LDAA   PUSEM    GET CURRENT <UNIT><SECTOR>
02201A 0FCB 5F                CLRB           SELECT TRACK 0 FOR THE DIRECTORY
02202A 0FCC BD 0401  A        JSR    RED11    READ IN THE DIRECTORY SECTOR
02203A 0FCF 4D                TSTA           IF A "DD" MARK WAS FOUND IN THIS SECTOR
02204A 0FD0 27 05 0FD7        BEQ    DIRS5    THEN
02205A 0FD2 7C 0F24  A DIRS4  INC    PUSEM    TAKE THE NEXT SECTOR
02206A 0FD5 20 DC 0FB3        BRA    DIRS1    AND TRY IT
02207                 *
02208A 0FD7 FE 0F22  A DIRS5  LDX    PUFMP    USE THE CURRENT SECTOR POINTER
02209A 0FDA A6 05    A        LDAA   $5,X     READ THE ATTRIBUTE VALUE FROM THE CURRENT FCB
02210A 0FDC 39                RTS            AND RETURN TO CALLER
02211                 *
02212                 *
02213                 *ADVANCE BUFFER POINTER BY 11 TO NEXT FCB.
02214                 *WHEN NEEDED, GET NEXT DIRECTORY SECTOR
02215                 *
02216A 0FDD 86 0B    A NXFCB  LDAA   #11      DEFINE 11 BYTES IN A FILE CONTROL BLOCK
02217A 0FDF 5F                CLRB           WITH A HIGH BYTE OF 0
02218A 0FE0 BB 0F23  A        ADDA   PUFMP+LBYT ADD 11 TO LOW BYTE OF POINTER
02219A 0FE3 F9 0F22  A        ADCB   PUFMP    ADD IN THE CARRY (IF ANY) TO THE HIGH BYTE
02220A 0FE6 B7 0F23  A        STAA   PUFMP+LBYT STORE THE LOW POINTER BYTE
02221A 0FE9 F7 0F22  A        STAB   PUFMP    AND THE HIGH POINTER BYTE (POINTER IS NOW +11)
02222A 0FEC FE 0F22  A        LDX    PUFMP    USE THE NEW POINTER VALUE (NEXT FCB)
02223A 0FEF 8C 0149  A        CPX    #SECB1+121 IF THERE ARE NO MORE FCB'S IN THIS SECTOR
02224A 0FF2 26 C5 0FB9        BNE    DIRS2    THEN
02225A 0FF4 20 DC 0FD2        BRA    DIRS4    TAKE THE NEXT SECTOR & FIRST FCB IN THAT SECTOR
```

```
02226                   *
02227                   *
02228                   *SEARCH DIRECTORY FOR FILES TO BE PURGED (ATTRIBUTE = $80)
02229                   *
02230A 0FF6 CE 0F39  A PKDSK  LDX    #PKDSM+1 LOAD POINTER TO MESSAGE STRING
02231A 0FF9 F6 0F38  A        LDAB   PKDSM    LOAD THE CHARACTER COUNT
02232A 0FFC BD 03B3  A        JSR    PRINS    PRINT "PACKING DISK"
02233A 0FFF BD 03BD  A        JSR    CRLF     PRINT <CR><LF> FOR NEW LINE
02234A 1002 BD 0FA7  A        JSR    DIRSE    READ & SET UP THE FIRST DIRECTORY SECTOR
02235                   *
02236A 1005 81 FF    A PKDS1  CMPA   #$FF     IF THIS IS THE END OF THE DIRECTORY TRACK
02237A 1007 26 01 100A        BNE    PKDS2    THEN
02238A 1009 39                RTS             RETURN
02239A 100A 81 80    A PKDS2  CMPA   #$80     IF ATTRIBUTE OF CURRENT FCB IS NOT "DELETED FILE" ($80
02240A 100C 27 05 1013        BEQ    PURFL    THEN
02241A 100E BD 0FDD  A        JSR    NXFCB    LOOK AT THE NEXT FCB
02242A 1011 20 F2 1005        BRA    PKDS1    AND CONTINUE
02243                   *
02244                   *PURGE THIS FILE, MOVE THE FILES, UPDATE THE DIRECTORY
02245                   *
02246A 1013 FE 0F22  A PURFL  LDX    PUFMP    GET POINTER TO CURRENT FCB
02247A 1016 A6 06    A        LDAA   $6,X     GET THE STARTING TRACK ADDRESS OF FILE TO BE PURGED
02248A 1018 97 0B    A        STAA   OTRK     DEFINE IT AS START OF AN OUTPUT FILE
02249A 101A B6 0F1E  A        LDAA   PUUID    GET PURGE UNIT NUMBER
02250A 101D 97 02    A        STAA   OUNIT    DEFINE OUTPUT FILE ON THAT UNIT
02251A 101F 0C                CLC             CLEAR THE "C" BIT FOR SHIFTING
02252A 1020 46                RORA            SHIFT THE UID INTO BITS 6 & 7
02253A 1021 46                RORA            SHIFTING THROUGH THE CARRY BIT
02254A 1022 46                RORA            BIT 7 BECOMES OLD 1, BIT 6 BECOMES OLD 0
02255A 1023 AA 07    A        ORAA   $7,X     INCLUDE THE START SECTOR OF THE FILE TO BE PURGED
02256A 1025 97 0C    A        STAA   OSCTR    DEFINE AS THE OUTPUT FILE <UNIT><SECTOR>
02257A 1027 7F 000D  A        CLR    OCNTR    CLEAR THE BYTE COUNTER FOR THE OUTPUT FILE
02258A 102A CE 07B7  A        LDX    #1975    DEFINE A FILE SIZE OF ALL USER SPACE ON DISK
02259A 102D DF 09    A        STX    OSIZE    AS THE OUTPUT FILE SIZE (SHUT OFF FILE SIZE MONITOR)
02260A 102F DF C7    A        STX    FILRQ    REQUEST A FILE SIZE OF ALL USER SPACE
02261A 1031 FE 0F22  A        LDX    PUFMP    GET POINTER TO CURRENT FCB
02262A 1034 FF 0F1F  A        STX    PUFCP    SAVE IT FOR LATER USE
02263A 1037 B6 0F24  A        LDAA   PUSEM    GET THE CURRENT DIRECTORY <UNIT><SECTOR>
02264A 103A B7 0F21  A        STAA   PUSEC    SAVE IT FOR USE LATER
02265                   *
02266A 103D BD 0FDD  A PURF1  JSR    NXFCB    TAKE THE NEXT FILE CONTROL BLOCK
02267A 1040 81 FF    A        CMPA   #$FF     IF THIS IS NOT THE END OF THE DIRECTORY TRACK
02268A 1042 27 36 107A        BEQ    PURF3    AND
02269A 1044 81 80    A        CMPA   #$80     IF THIS IS NOT ANOTHER DELETED FILE
02270A 1046 27 F5 103D        BEQ    PURF1    THEN
02271                   *
02272A 1048 FE 0F22  A        LDX    PUFMP    POINT TO THE CURRENT FILE CONTROL BLOCK
02273A 104B A6 06    A        LDAA   $6,X     PICK UP THIS FILE'S STARTING TRACK ADDRESS
02274A 104D 97 06    A        STAA   ITRK     DEFINE THIS AS AN INPUT FILE TRACK START ADDRESS
02275A 104F A6 08    A        LDAA   $8,X     PICK UP THIS FILE'S LENGTH (HIGH BYTE)
02276A 1051 97 04    A        STAA   ISIZE    DEFINE AS INPUT FILE SIZE (HIGH BYTE)
02277A 1053 A6 09    A        LDAA   $9,X     PICK UP THIS FILE'S LENGTH (LOW BYTE)
02278A 1055 97 05    A        STAA   ISIZE+LBYT DEFINE AS INPUT FILE SIZE (LOW BYTE)
02279A 1057 7F 0008  A        CLR    ICNTR    CLEAR THE INPUT FILE BYTE COUNTER
02280A 105A B6 0F1E  A        LDAA   PUUID    GET PURGE UNIT NUMBER
02281A 105D 0C                CLC             CLEAR THE "C" BIT FOR SHIFTING
02282A 105E 46                RORA            SHIFT THE UID INTO BITS 6 & 7
02283A 105F 46                RORA            ROTATING THEM THROUGH THE CARRY BIT
```

```
02284A 1060 46               RORA           BIT 7 BECOMES OLD 1, BIT 6 BECOMES OLD 0
02285A 1061 AA 07    A       ORAA   $7,X    INCLUDE SECTOR TO MAKE <UNIT><SECTOR>
02286A 1063 4A               DECA           MINUS ONE
02287A 1064 97 07    A       STAA   ISCTR   AS THE STARTING SECTOR FOR AN INPUT FILE
02288A 1066 0C               CLC            CLEAR THE "C" BIT FOR SHIFTING
02289A 1067 49               ROLA           SHIFT UID BACK INTO BITS 1 & 0
02290A 1068 49               ROLA           SECTOR NUMBER NOW OCCUPIES BITS 3 - 7
02291A 1069 49               ROLA           (SECTOR NUMBER IS * 8)
02292A 106A 97 03    A       STAA   IUNIT   SAVE THIS AS INPUT UNIT <SECTOR><UNIT>
02293A 106C BD 0EE9  A       JSR    MOVFL   MOVE THE INPUT FILE INTO SPACE CREATED BY PURGING
02294A 106F 7D 000D  A PURF2 TST    OCNTR   IF OUTPUT SECTOR NOT COMPLETE
02295A 1072 27 09 107D       BEQ    PURF4   THEN
02296A 1074 4F               CLRA           DEFINE A NULL
02297A 1075 BD E80C  A       JSR    XWRT    PAD THE REMAINDER OF THIS SECTOR WITH NULLS
02298A 1078 20 F5 106F       BRA    PURF2   WHILE SECTOR, CONTINUE
02299                *
02300A 107A 7E 1132  A PURF3 JMP    WTEOD   CLOSE END OF DIRECTORY (BRANCH RANGE EXTENSION)
02301                *
02302                *
02303                *UPDATE THE FCB NOW RELATING TO THE FILE JUST MOVED
02304                *
02305A 107D FE 0F22  A PURF4 LDX    PUFMP   SET THE POINTER TO THE FCB OF THE FILE JUST MOVED
02306A 1080 A6 05    A       LDAA   $5,X    GET FILE ATTRIBUTE
02307A 1082 36               PSHA           AND SAVE IT
02308A 1083 A6 04    A       LDAA   $4,X    GET NAME CHAR #5
02309A 1085 36               PSHA           AND SAVE IT
02310A 1086 A6 03    A       LDAA   $3,X    GET NAME CHAR #4
02311A 1088 36               PSHA           AND SAVE IT
02312A 1089 A6 02    A       LDAA   $2,X    GET NAME CHAR #3
02313A 108B 36               PSHA           AND SAVE IT
02314A 108C A6 01    A       LDAA   $1,X    GET NAME CHAR #2
02315A 108E 36               PSHA           AND SAVE IT
02316A 108F A6 00    A       LDAA   $0,X    GET NAME CHAR #1
02317A 1091 36               PSHA           AND SAVE IT
02318                *
02319A 1092 D6 C7    A       LDAB   FILRQ   GET THE CURRENT "MAXIMUM USER DISK SPACE"
02320A 1094 96 C8    A       LDAA   FILRQ+LBYT (IT STARTED OUT AS 1975 SECTORS)
02321A 1096 90 0A    A       SUBA   OSIZE+LBYT SUBTRACT THE SECTORS NOT USED DURING
02322A 1098 D2 09    A       SBCB   OSIZE   WRITE, GIVING THIS FILE'S SECTOR COUNT
02323A 109A 8B 01    A       ADDA   #$1     OR SIZE, AFTER ALLOWING ONE MORE SECTOR
02324A 109C C9 00    A       ADCB   #$0     WITH PROPAGATED CARRY
02325A 109E 37               PSHB           SAVE FILE SIZE (HIGH)
02326A 109F 36               PSHA           SAVE FILE SIZE (LOW)
02327                *
02328A 10A0 CE 00D0  A       LDX    #SECB1  POINT TO THE DIRECTORY SECTOR BUFFER
02329A 10A3 B6 0F21  A       LDAA   PUSEC   GET <UNIT><SECTOR> FOR PURGED FILE FCB
02330A 10A6 5F               CLRB           USE TRACK 0 FOR DIRECTORY READ
02331A 10A7 BD 0401  A       JSR    RED11   READ THE DIRECTORY SECTOR
02332                *
02333                *AN ERROR CHECK HAS BEEN MISSED HERE!!!!
02334                *
02335A 10AA FE 0F1F  A       LDX    PUFCP   POINT TO THE CURRENT PURGED FILE FCB
02336A 10AD 32               PULA           GET LOW BYTE OF MOVED FILE SIZE
02337A 10AE A7 09    A       STAA   $9,X     STORE IT AS FCB LENGTH (LOW)
02338A 10B0 32               PULA           GET HIGH BYTE OF MOVED FILE SIZE
02339A 10B1 A7 08    A       STAA   $8,X     STORE IT AS FCB LENGTH (HIGH)
02340A 10B3 32               PULA           GET THE FIRST CHAR OF THE MOVED FILE NAME
02341A 10B4 A7 00    A       STAA   $0,X     STORE IT AS FCB NAME, CHAR 1
```

```
02342A 10B6 32               PULA           GET THE SECOND CHAR OF THE MOVED FILE NAME
02343A 10B7 A7 01    A       STAA   $1,X    STORE IT AS FCB NAME, CHAR 2
02344A 10B9 32               PULA           GET THE THIRD CHAR OF THE MOVED FILE NAME
02345A 10BA A7 02    A       STAA   $2,X    STORE IT AS FCB NAME, CHAR 3
02346A 10BC 32               PULA           GET THE FOURTH CHAR OF THE MOVED FILE NAME
02347A 10BD A7 03    A       STAA   $3,X    STORE IT AS FCB NAME, CHAR 4
02348A 10BF 32               PULA           GET THE FIFTH CHAR OF THE MOVED FILE NAME
02349A 10C0 A7 04    A       STAA   $4,X    STORE IT AS FCB NAME, CHAR 5
02350A 10C2 32               PULA           GET THE ATTRIBUTE CHARACTER OF THE MOVED FILE
02351A 10C3 A7 05    A       STAA   $5,X    STORE IT AS FCB ATTRIBUTE CHAR
02352                *
02353                *FCB IS NOW UPDATED (TRACK & SECTOR ARE KNOWN TO BE CORRECT)
02354                *
02355A 10C5 CE 00D0  A       LDX    #SECB1  POINT TO THE BEGINNING OF THE SECTOR BUFFER
02356A 10C8 B6 0F21  A       LDAA   PUSEC   GET THE <UNIT><SECTOR> FOR THIS FCB ENTRY
02357A 10CB 5F               CLRB           SELECT TRACK 0 FOR THE DIRECTORY
02358A 10CC BD 0479  A       JSR    WRTSE   WRITE THIS SECTOR TO DISK
02359A 10CF 4D               TSTA           IF THERE WAS A WRITE ERROR
02360A 10D0 27 03 10D5       BEQ    PURF5   THEN
02361A 10D2 7E 042E  A       JMP    MEDER   PRINT "MEDIA ERROR" & WAIT FOR NEW COMMAND
02362                *
02363A 10D5 B6 0F22  A PURF5 LDAA   PUFMP   GET POINTER TO CURRENT FCB (HIGH BYTE)
02364A 10D8 36               PSHA           AND SAVE IT
02365A 10D9 B6 0F23  A       LDAA   PUFMP+LBYT GET POINTER TO CURRENT FCB (LOW BYTE)
02366A 10DC 36               PSHA           AND SAVE IT
02367A 10DD B6 0F24  A       LDAA   PUSEM   GET CURRENT FCB <UNIT><SECTOR>
02368A 10E0 36               PSHA           AND SAVE IT
02369A 10E1 FE 0F1F  A       LDX    PUFCP   GET THE POINTER FOR FCB OF THE PURGED FILE
02370A 10E4 FF 0F22  A       STX    PUFMP   AND SAVE IT AS TARGET FOR FCB MOVE
02371A 10E7 B6 0F21  A       LDAA   PUSEC   GET THE <UNIT><SECTOR> FOR FCB OF THE PURGED FILE
02372A 10EA B7 0F24  A       STAA   PUSEM   AND SAVE IT AS TARGET FOR FCB MOVE
02373A 10ED BD 0FDD  A       JSR    NXFCB   GET THE NEXT FCB
02374A 10F0 FE 0F22  A       LDX    PUFMP   POINT TO IT IN THE SECTOR BUFFER
02375A 10F3 96 0B    A       LDAA   OTRK    GET THE LAST TRACK USED
02376A 10F5 A7 06    A       STAA   $6,X    SAVE IT AS START TRACK IN THE NEW FCB
02377A 10F7 96 0C    A       LDAA   OSCTR   GET THE LAST <UNIT><SECTOR> USED
02378A 10F9 84 3F    A       ANDA   #$3F    STRIP OFF THE UNIT ID BITS
02379A 10FB A7 07    A       STAA   $7,X    SAVE IT AS START SECTOR IN THE NEW FCB
02380A 10FD CE 00D0  A       LDX    #SECB1  START AT THE BEGINNING OF THE SECTOR BUFFER
02381A 1100 B6 0F24  A       LDAA   PUSEM   GET THE <UNIT><SECTOR> FOR THE CHANGED FCB
02382A 1103 5F               CLRB           SELECT TRACK 0 FOR THE DIRECTORY
02383A 1104 BD 0479  A       JSR    WRTSE   WRITE THIS SECTOR TO DISK
02384A 1107 4D               TSTA           IF THERE WAS A WRITE ERROR
02385A 1108 27 03 110D       BEQ    PURF7   THEN
02386A 110A 7E 042E  A PURF6 JMP    MEDER   PRINT "MEDIA ERROR" & WAIT FOR NEW COMMAND
02387A 110D CE 07B7  A PURF7 LDX    #1975   DEFINE MAXIMUM USER SPACE AVAILABLE ON DISK
02388A 1110 DF 09    A       STX    OSIZE   SET AN OUTPUT SIZE OF MAX. (SHUT OFF OUTPUT TEST)
02389A 1112 DF C7    A       STX    FILRQ   REQUEST A OUTPUT FILE OF MAXIMUM USER SPACE
02390A 1114 7F 000D  A       CLR    OCNTR   CLEAR THE OUTPUT FILE BYTE COUNTER
02391A 1117 FE 0F22  A       LDX    PUFMP   GO BACK TO THE PURGED FILE FCB
02392A 111A FF 0F1F  A       STX    PUFCP   IN THE SAVE LOCATION
02393A 111D B6 0F24  A       LDAA   PUSEM   GO BACK TO THE PURGED FILE <UNIT><SECTOR>
02394A 1120 B7 0F21  A       STAA   PUSEC   IN THE SAVE LOCATION
02395A 1123 32               PULA           GET CURRENT <UNIT><SECTOR> OFF STACK
02396A 1124 B7 0F24  A       STAA   PUSEM   AND RESTORE IT
02397A 1127 32               PULA           GET LOW BYTE OF CURRENT FCB PONTER OFF STACK
02398A 1128 B7 0F23  A       STAA   PUFMP+LBYT AND RESTORE IT
02399A 112B 32               PULA           GET HIGH BYTE OF CURRENT FCB POINTER OFF STACK
```

```
02400A 112C B7 0F22  A        STAA   PUFMP    AND RESTORE IT
02401A 112F 7E 103D  A        JMP    PURF1    CONTINUE THE PACKING PROCESS
02402                *
02403                *
02404                *THIS SUBROUTINE WRITES AN END-OF-DIRECTORY FCB ENTRY
02405                *
02406A 1132 CE 00D0  A WTEOD  LDX    #SECB1   POINT TO THE SECTOR BUFFER
02407A 1135 B6 0F21  A        LDAA   PUSEC    GET THE CURRENT <UNIT><SECTOR>
02408A 1138 5F               CLRB            SELECT TRACK 0 FOR THE DIRECTORY
02409A 1139 BD 0401  A        JSR    RED11    READ THIS SECTOR FROM THE DISK
02410                *
02411                *AN ERROR CHECK IS MISSING HERE!!!
02412                *
02413A 113C FE 0F1F  A        LDX    PUFCP    POINT TO THE "NEXT FCB" DEFINED AT LAST MOVE
02414A 113F 86 2A    A        LDAA   #'*      DEFINE ASCII CHARACTER "*"
02415A 1141 A7 00    A        STAA   $0,X     DEFINE THE NAME
02416A 1143 A7 01    A        STAA   $1,X     OF THE EOD
02417A 1145 A7 02    A        STAA   $2,X     FILE TO BE
02418A 1147 A7 03    A        STAA   $3,X     "*****"
02419A 1149 A7 04    A        STAA   $4,X     AS AN IDENTIFIER
02420A 114B 86 FF    A        LDAA   #$FF     DEFINE EOD ATTRIBUTE
02421A 114D A7 05    A        STAA   $5,X     AS NEW FCB ATTRIBUTE
02422A 114F 96 0B    A        LDAA   OTRK     TAKE THE LAST TRACK WRITTEN
02423A 1151 A7 06    A        STAA   $6,X     AS THE START TRACK FOR THIS FCB
02424A 1153 96 0C    A        LDAA   OSCTR    TAKE THE LAST <UNIT><SECTOR> WRITTEN
02425A 1155 84 3F    A        ANDA   #$3F     STRIP OFF UNIT ID BITS
02426A 1157 A7 07    A        STAA   $7,X     AND USE AS START SECTOR FOR THIS FCB
02427A 1159 CE 00D0  A        LDX    #SECB1   POINT TO THE BEGINNING OF THE SECTOR BUFFER
02428A 115C B6 0F21  A        LDAA   PUSEC    GET <UNIT><SECTOR> FOR THIS FCB
02429A 115F 5F               CLRB            SELECT TRACK 0 FOR THE DIRECTORY
02430A 1160 BD 0479  A        JSR    WRTSE    WRITE THIS SECTOR TO DISK
02431A 1163 4D               TSTA            IF THERE WAS NO WRITE ERROR
02432A 1164 26 A4 110A       BNE    PURF6    THEN
02433A 1166 39               RTS             RETURN TO CALLER
02434                *
02435                *
02436                **TRACK BUFFER & LARGE BUFFER ALLOCATION BEGINS AFTER EDOS
02437                *
02438A 1200                  ORG    $1200
02439                *
02440          1200  A TRKBF  EQU    *        DISK TRACK BUFFER START ADDRESS
02441A 1200    0CFE  A BGBUF  RMB    3326     LARGE DATA BUFFER
02442          1EFE  A BGBFE  EQU    *        END OF ALLOCATION FOR LARGE DATA BUFFER
02443                *
02444                        END
TOTAL ERRORS 00000--00000



   FCF4 ACIAC  00062*01682 01712
   FCF5 ACIAR  00063*00441 01698 01715 01716
   FCF4 ACIAS  00061*00438 01688
   FCF5 ACIAT  00064*
   015B AERFL  00139*01286 01313 01321 01335
   0159 APARS  00137*01450 01465
   0931 ASM    00219 01237*
   0963 ASM1   01259 01261*01337
   097F ASM2   01243 01247 01275*
```

```
08B1 DIR8   01170*01212
02A4 DIRHD  00290*01143 01144
0FB3 DIRS1  02189*02206
0FB9 DIRS2  02192*02224
0FC5 DIRS3  02195 02199*
0FD2 DIRS4  02205*02225
0FD7 DIRS5  02204 02208*
0FA7 DIRSE  02181*02234
EC03 DKCOC  00055*00502 00508 01979 01985
EC02 DKCOD  00054*00504 00510 00511 00550 01117 01980 01986 01987 02004
EC01 DKDIC  00053*
EC00 DKDID  00052*00481 00484 00505 00523 00559 01961 01969 01981 02018
EC07 DKDOC  00059*
EC06 DKDOD  00058*00548 01115 02003
02C0 DKNRM  00292*00527 00528
0010 DLE    00011*01816 01834
0B8A DOOF1  01463 01550*
0B91 DOOF2  01553*01566
0B9C DOOF3  01557 01560*
0BA3 DOOF4  01561 01564*
0B81 DOOFL  01542*01601
0470 DSKR1  00525 00527*
0468 DSKRD  00474 00523*00539 01875 01910
0EC5 DUCR1  02027 02032*
0ECA DUCR2  02020 02035*
0EA0 DUCRC  01936 02014*02025 02030
0CC6 DUMP1  01748 01750*
0D11 DUMR1  01791*01798
0D22 DUMR2  01792 01801*
0D0C DUMRD  01776 01788*
0D4C DUMW1  01825*01830
0D31 DUMWT  01779 01782 01814*
0292 DUNMM  00286*01075 01076
0D7F DUP    00233 01866*
0D8D DUP1   01873*01951
0D9F DUP2   01880*01895 01902
0DC9 DUP3   01890 01893 01901*
0DCD DUP4   01899 01904*
0DD1 DUP5   01906*01942
0DE3 DUP6   01913*01928 01934
0E0A DUP7   01923 01926 01933*
0E0E DUP8   01931 01936*
0E22 DUP9   01937 01945*
0E2E DUPA   01948 01951*
0811 DUPNM  01021 01075*
0E36 DUSR1  01959*01966
0E4A DUSR2  01963 01967*
0E59 DUSR3  01971 01974*
0E61 DUSR4  01978*01989
0E31 DUSRD  01883 01957*
0E8E DUWT1  02002*02007
0E83 DUWTS  01916 01996*
03CA ECHO1  00412 00422*
03DA ECHO2  00421 00426 00431*
03C1 ECHOC  00315 00418*
0B77 EDINF  01489 01535*
0B12 EDIT   00221 01485*
0B21 EDIT1  01492*01526
```

```
   0B57 EDIT2  01488 01518*
   E824 EDITR  00033*00913 01513 01514
   0B7E EDNOF  01533 01537 01540*01545
   0158 EOFFL  00136*01775 01777 01807 02063 02083 02087
   0183 ERRM   00178*00324 00367 00965 01083 01122 01275 01339
   080D ERTYP  01070 01073*01562
   001B ESC    00013*01645
   FF8A EXBST  00047*00907
   F564 EXBUG  00048*00225
   015C EXEC   00145 00158*
   0020 EXECS  00145*
   FF62 EXPUN  00046*01815 01838
   006B EXSTK  00093*00158 00166 00860
   0823 FERR1  01066 01083*
   085B FERR2  01087 01122*
   0278 FERRM  00178 00179 00282*
   00C6 FILAT  00110*00756 00878 01061 01554
   00C7 FILRQ  00111*00735 00736 00758 00759 00768 00772 00889 00890 01067 01552 01565 02260 02319 02320 02389
   0B12 FLABE  01375 01482*
   0B00 FLABF  01355 01481*
   069B FLCL1  00863*00892
   069E FLCL2  00862 00865*00869
   06A9 FLCL3  00866 00871*
   0692 FLCLS  00146 00859*01621 02058
   00C0 FLNM0  00103*00184 00346 00354 00602 00745 00872 00956 01033 01238 01262 01292 01299 01306 01315 01352 01437
               01486 01493 01531 01543 02047 02138 02152 02171
   00C1 FLNM1  00104*00187 00605 00747 01037 01356 01382
   00C2 FLNM2  00105*00190 00608 00749 00873 01039 01360 01384
   00C3 FLNM3  00106*00193 00611 00751 00874 01041 01386
   00C4 FLNM4  00107*00196 00614 00753 01043
   00C5 FLNM5  00108*00353 00363 00369 00575 00647 00685 00876 01014 01018 01086 01092 01093 01109 01135 01261 01441
               01492 01572 02135 02144
   04C9 FNDF1  00581*00592 00629
   04E4 FNDF2  00590 00593*
   04E9 FNDF3  00595*00627
   0515 FNDF4  00600 00603 00606 00609 00612 00615 00619*
   04BE FNDFL  00574*00635 00671 00705 00879 00951 01019 01026
   0514 FNDFX  00584 00598 00617*
   0C34 GEND1  01672*01728
   0C45 GEND2  01673 01680*
   0C52 GEND3  01678 01685*
   0C55 GEND4  01686*01704
   0C5A GEND5  01688*01692
   0C69 GEND6  01690 01697*
   0C79 GEND7  01695 01701 01706*
   0C98 GEND8  01723*01732
   0CAC GEND9  01730 01734*01738
   0CAD GENDA  01671 01736*
   0CB0 GENDB  01737*01740
   0C2E GENDV  01620 01642 01669*
   0C0F GENS1  01649*01653
   0C1A GENS2  01650 01655*
   0B47 GEPR1  01507 01509*
   0B54 GEPR2  01512 01514*
   0B3A GEPRO  01273 01503*
   03DB GETCH  00418 00438*00440
   0100 HBYT   00022*
   00C9 HDIGT  00112*00450 00456 00992 01003
```

```
   096A TCHR0  01256 01264*01282
   0CD2 TDMP1  01755*01760 01772
   0CE0 TDMP2  01756 01762*
   0CF0 TDMP3  01758 01769*
   0CBA TDUMP  00243 01743*
   FFFF TIMAX  00025*01687
   000F TISZE  00085*01587
   000E TITRK  00084*01585
   0156 TIUNT  00134*01509 01589
   0159 TKBFP  00138*01284 01323 01327 01381 01408 01424
   1200 TRKBF  01322 01329 01685 01721 01788 01824 01878 01906 02440*
   018E TRYCM  00177 00183*
   0155 TRYCT  00133*00478 00488 00543 00563 01958 01965
   0BE3 TTAPE  00250 01618*
   0BEC TXGEN  00260 01634*
   038B UIDER  00362 00367*
   00C9 UNUMB  00113*01015 01017
   0334 USCM1  00315*00326 00338
   0349 USCM2  00323 00325*
   0350 USCM3  00317 00333*
   035D USCM4  00319 00340*
   0327 USCMD  00167 00310*00334
   0150 USECT  00126*00580 00581 00586 00591 00628 00792 00797 00798 00808 00837 00843 00850 01046 01099 01141 01152
               01159 01165 01213 01656 02163
   0D6D WAIT1  01843*01844
   00C9 WRTRY  00115*01905 01941
   048F WRTS1  00547*00553
   049D WRTS2  00555*00564
   04BD WRTS3  00561 00568*
   0479 WRTSE  00536*00794 00845 01048 01332 02166 02358 02383 02430
   016E WTCMD  00164*00181 00200 00340 00863 00940 01071 01157 01767 01783 01949
   1132 WTEOD  02300 02406*
   0609 WTND1  00770 00776*00784
   0616 WTND2  00780 00783*
   0619 WTND3  00777 00785*
   0626 WTND4  00791*00801
   0642 WTND5  00796 00806*01103 01662
   064C WTND6  00789 00813*
   0657 WTND7  00809 00819*
   065D WTND8  00823*00826
   0673 WTND9  00836*00851
   0680 WTNDA  00840 00843*
   068D WTNDB  00847 00850*
   05E1 WTNDX  00756*00774 00891
   E809 XRI    00030*00932 01755 01791 02066
   EA03 XUS    00034*00473 00538
   EA22 XUSP   00035*01874 01881 01909 01914 02015
   E80C XWRT   00031*00868 01644 01646 01648 01652 01731 01739 02078 02297
   0A92 ZERFL  01295 01302 01423*
```