```
OS SERIES 9
GEOFF COX
 ************************************************************************
 *                                                                      *
 *       OSBYTE 140  *TAPE                                              *
 *       selects TAPE filing system                                     *
 *                                                                      *
 *                                                                      *
 *       OSBYTE 141  *ROM                                               *
 *       selects ROM filing system                                      *
 *                                                                      *
 ************************************************************************


 F135    EOR     #&8C    ;if its *TAPE A=0 *ROM A=1
 F137    ASL             ;double it
 F138    STA     &0247   ;store it in filing system flag store
 F13B    CPX     #&03    ;if X>=3 C set X=3 Z set
 F13D    JMP     &F14B   ;

 ******** set cassette options ***************************************
         ;called after BREAK etc
         ;lower nybble sets sequential access
         ;upper sets load and save options

         ;0000   Ignore errors,          no messages
         ;0001   Abort if error,         no messages
         ;0010   Retry after error,      no messages
         ;1000   Ignore error            short messages
         ;1001   Abort if error          short messages
         ;1010   Retry after error       short messages
         ;1100   Ignore error            long messages
         ;1101   Abort if error          long messages
         ;1110   Retry after error       long messages


 F140    PHP             ;save flags
 F141    LDA     #&A1    ;set sequential access abort if error, no messages
 F143    STA     &E3     ;set load/save retry if error, short messages
 F145    LDA     #&19    ;set interblock gap
 F147    STA     &03D1   ;and store it
 F14A    PLP             ;get back flags

 F14B    PHP             ;push flags
 F14C    LDA     #&06    ;get close files command to FSCV
 F14E    JSR     &E031   ;and gosub OSFSC
 F151    LDX     #&06    ;
 F153    PLP             ;get back flags
 F154    BEQ     &F157   ;if Z set earlier
 F156    DEX             ;do not decrement X
 F157    STX     &C6     ;set current baud rate X=5 300 baud X=6 1200 baud

 ********* reset FILEV,ARGSV,BGETV,BPUTV,GBPBV,FINDV,FSCV *****************
 **********   to F27D, F18E, F4C9, F529, FFA6, F3CA, F1B1 *****************

 F159    LDX     #&0E    ;RESET VECTORS FOR FILE RELATED OPERATIONS
 F15B    LDA     &D951,X ;
 F15E    STA     &0211,X ;
 F161    DEX             ;
 F162    BNE     &F15B   ;

 F164    STX     &C2     ;&C2=0 PROGRESS FLAG
 F166    LDX     #&0F    ;set X to make Rom service call &F claim vectors!
```

```
*************************************************************************
*                                                                       *
*       OSBYTE 143                                                      *
*       Pass service commands to sideways Roms                          *
*                                                                       *
*************************************************************************
                ;on entry X=command number

F168    LDA     &F4     ;get current Rom number
F16A    PHA             ;store it
F16B    TXA             ;command in A
F16C    LDX     #&0F    ;set X=15

                        ;send commands loop
F16E    INC     &02A1,X ;read bit 7 on rom map (no rom has type 254 &FE)
F171    DEC     &02A1,X ;
F174    BPL     &F183   ;if not set (+ve result)
F176    STX     &F4     ;else store rom number in &F4
F178    STX     &FE30   ;switch in paged ROM
F17B    JSR     &8003   ;and jump tp service entry
F17E    TAX             ;on exit put A in X
F17F    BEQ     &F186   ;if 0 (command recognised by ROM) reset roms & exit
F181    LDX     &F4     ;else point to next lower rom
F183    DEX             ;
F184    BPL     &F16E   ;and go round loop again

F186    PLA             ;get back original Rom number
F187    STA     &F4     ;store it in Ram copy
F189    STA     &FE30   ;select original page
F18C    TXA             ;put X back in A
F18D    RTS             ;and return



*************************************************************************
*                                                                       *
*       OSARGS (default) entry point
*
*       on entry A determines action                                    *
*       A=0     save block of memory as a file                          *
*       A=1     write catalogue info for existing file                  *
*       A=2     write load address only for existing file               *
*       A=3     write execution address only for existing file          *
*       A=4     write attributes only for existing file                 *
*       A=5     Read catalogue info, return file type in A              *
*       A=6     Delete named file                                       *
*       A=&FF   load the named file if lo byte of Exec address=0 use    *
*               address in parameter block else files own load address  *
*       X,Y     point to parameter block                                *
*       bytes   0,1 filename address, 2-5 load,6-9 exec,A-D length or   *
*               start of data for save, 0E End address /attributes      *
*************************************************************************

F18E    ORA     #&00    ;is A=00
F190    BNE     &F1A2   ;if not return
F192    CPY     #&00    ;is Y=0
F194    BNE     &F1A2   ;if not return
F196    LDA     &C6     ;else get current baud rate and zero bit 2
F198    AND     #&FB    ;C6=5 becomes 1, 6 becomes 2
F19A    ORA     &0247   ;if cassette selected A=0 else A=2
F19D    ASL             ;multiply by 2
F19E    ORA     &0247   ;Or it again
F1A1    LSR             ;divide by 2
F1A2    RTS             ;return cassette =0
```

```
*************************************************************************
*                                                                       *
*       FSC     VECTOR  TABLE                                           *
*                                                                       *
*************************************************************************

F1A3    DW      4C,F5   ; *OPT          (F54C)
F1A5    DW      1D,F6   ; check EOF     (F61D)
F1A7    DW      04,F3   ; */            (F304)
F1A9    DW      0F,E3   ; Bad Command   (E30F) if roms and FS don't want it
F1AB    DW      04,F3   ; *RUN          (F304)
F1AD    DW      2A,F3   ; *CAT          (F32A)
F1AF    DW      74,E2   ; osbyte 77     (E274)


*************************************************************************
*       Filing System control entry     OSFSC                           *
*       Entry via 021E FSCV                                             *
*       A= index 0 to 7                                                 *
*************************************************************************
        ;on entry A is reason code
        ;A=0    A *OPT command has been used X & Y are the 2 parameters
        ;A=1    EOF is being checked, on entry  X=File handle
                                        on Exit X=FF = EOF exists else 00
        ;A=2    A */ command has been used *RUN the file
        ;A=3    An unrecognised OS command has ben used X,Y point at command
        ;A=4    A *RUN command has been used X,Y point at filename
        ;A=5    A *CAT cammand has been issued X,Y point to rest of command
        ;A=6    New filing system about to take over close *SPOOL &*EXEC files
        ;A=7    Return in X and Y lowest and highest file handle used
        ;A=8    OS about to process *command

;A=7 and A=8 are handled by current filing system by changing FSCV

F1B1    CMP     #&07    ;if A>6
F1B3    BCS     &F1A2   ;goto F1A2 (RTS)
F1B5    STX     &BC     ;else save X
F1B7    ASL             ;A=A*2
F1B8    TAX             ;X=A to get offset
F1B9    LDA     &F1A4,X ;get hi byte of address
F1BC    PHA             ;push it
F1BD    LDA     &F1A3,X ;get lo byte of address
F1C0    PHA             ;push it
F1C1    LDX     &BC     ;get back X
F1C3    RTS             ;this now jumps to the address got from the table +1
                        ;the next RTS takes us back to CLI


*************************************************************************
*                                                                       *
*       LOAD FILE                                                       *
*                                                                       *
*************************************************************************

F1C4    PHP             ;save flags on stack
F1C5    PHA             ;save A on stack
F1C6    JSR     &FB27   ;Set cassette optionsinto (BB),set C7=6
                        ;claim serial system for cassette
F1C9    LDA     &03C2   ;execution address LO
F1CC    PHA             ;save A on stack
F1CD    JSR     &F631   ;search for file
F1D0    PLA             ;get back A
F1D1    BEQ     &F1ED   ;if A=0 F1ED
F1D3    LDX     #&03    ;else X=3
F1D5    LDA     #&FF    ;A=&FF
```

```
F1D7    PHA             ;save A on stack
F1D8    LDA     &03BE,X ;get load address
F1DB    STA     &B0,X   ;store it as current load address
F1DD    PLA             ;get back A
F1DE    AND     &B0,X   ;
F1E0    DEX             ;X=X-1
F1E1    BPL     &F1D7   ;until all 4 bytes copied

F1E3    CMP     #&FF    ;if all bytes contain don't contain &FF
F1E5    BNE     &F1ED   ;continue
F1E7    JSR     &FAE8   ;else sound bell, reset ACIA & motor off
F1EA    JMP     &E267   ;'Bad Address' error

F1ED    LDA     &03CA   ;block flag
F1F0    LSR             ;set carry from bit 0
F1F1    PLA             ;get back A
F1F2    BEQ     &F202   ;if A=0 F202
F1F4    BCC     &F209   ;if carry clear F209


*************** LOCKED FILE ROUTINE ************************************

F1F6    JSR     &FAF2   ;enable second processor and reset serial system

F1F9    BRK             ;
F1FA    DB      &E5     ;error number
F1FC    'Locked'        ;
F201    BRK             ;


F202    BCC     &F209   ;if carry clear F209
F204    LDA     #&03    ;else A=3
F206    STA     &0258   ;store to cause ESCAPE disable and memory
                        ;clear on break

F209    LDA     #&30    ;
F20B    AND     &BB     ;current OPTions
F20D    BEQ     &F213   ;if options and #&30 =0 ignore error condityion is set
F20F    LDA     &C1     ;else get checksum result
F211    BNE     &F21D   ;and if not 0 F21D

F213    TYA             ;A=Y
F214    PHA             ;save A on stack
F215    JSR     &FBBB   ;read from second processor if present
F218    PLA             ;get back A
F219    TAY             ;Y=A
F21A    JSR     &F7D5   ;reset flags and check block length
F21D    JSR     &F9B4   ;load file from tape
F220    BNE     &F255   ;if not found return to search
F222    JSR     &FB69   ;increment current block number
F225    BIT     &03CA   ;block flag
F228    BMI     &F232   ;if bit 7=1 then this is last block so F232
F22A    JSR     &F96A   ;else increment current load address
F22D    JSR     &F77B   ;read block header
F230    BNE     &F209   ;and goto F209

******** store data in OSFILE parameter block ***************************

F232    LDY     #&0A    ;Y=&0A
F234    LDA     &CC     ;file length counter lo
F236    STA     (&C8),Y ;OSFILE parameter  block
F238    INY             ;Y=Y+1
F239    LDA     &CD     ;file length counter hi
F23B    STA     (&C8),Y ;OSFILE parameter  block
F23D    LDA     #&00    ;A=0
F23F    INY             ;Y=Y+1
```

```
F240    STA     (&C8),Y ;OSFILE parameter  block
F242    INY             ;Y=Y+1
F243    STA     (&C8),Y ;OSFILE parameter  block
F245    PLP             ;get back flags
F246    JSR     &FAE8   ;bell, reset ACIA & motor
F249    BIT     &BA     ;current block flag
F24B    BMI     &F254   ;return
F24D    PHP             ;save flags on stack
F24E    JSR     &FA46   ; print message following call (in this case NEWLINE!)
F251    DB      &0D,&00 ;message
F254    RTS             ;return
        ;
************RETRY AFTER A FAILURE ROUTINE ******************************

F255    JSR     &F637   ;search for a specified block
F258    BNE     &F209   ;goto F209


*********** Read Filename using Command Line Interpreter ***************

;filename pointed to by X and Y

F25A    STX     &F2     ;OS filename/command line pointer lo
F25C    STY     &F3     ;OS filename/command line pointer
F25E    LDY     #&00    ;Y=0
F260    JSR     &EA1D   ;initialise string
F263    LDX     #&00    ;X=0
F265    JSR     &EA2F   ;GSREAD call
F268    BCS     &F277   ;if end of character string F277
F26A    BEQ     &F274   ;if 0 found F274
F26C    STA     &03D2,X ;else store character in CFS filename area
F26F    INX             ;X=X+1
F270    CPX     #&0B    ;if X<>11
F272    BNE     &F265   ;then read next
F274    JMP     &EA8F   ;else Bad String error


************* terminate Filename ***************************************

F277    LDA     #&00    ;terminate filename with 0
F279    STA     &03D2,X ;
F27C    RTS             ;return



*************************************************************************
*                                                                       *
*       OSFILE ENTRY                                                    *
*                                                                       *
*************************************************************************

;parameter block located by XY
;0/1    Address of Filename terminated by &0D
;2/4    Load Address of File
;6/9    Execution Address of File
;A/D    Start address of data for write operations or length of file
;       for read operations
;E/11   End address of Data; i.e. byte AFTER last byte to be written
;       or file attributes
;
;On Entry action is determined by value in A
;
;A=0    Save section of memory as named file, write catalogue information
```

```
;A=1    Write catalogue information for named file
;A=2    Write the LOAD       address (only) for the named File
;A=3    Write the EXECUTION  address (only) for the named File
;A=4    Write the ATTRIBUTES for the named File
;A=5    Read the named files catalogue information Place file type in A
;A=6    Delete the named file
;A=&FF  Load the named file and read its catalogue information

F27D    PHA             ;save A on stack
F27E    STX     &C8     ;osfile block pointer lo
F280    STY     &C9     ;osfile block pointer hi
F282    LDY     #&00    ;Y=0
F284    LDA     (&C8),Y ;OSFILE parameter  block
F286    TAX             ;X=A
F287    INY             ;Y=Y+1
F288    LDA     (&C8),Y ;OSFILE parameter  block
F28A    TAY             ;Y=A
F28B    JSR     &F25A   ;get filename from BUFFER
F28E    LDY     #&02    ;Y=2

F290    LDA     (&C8),Y ;copy parameters to Cassette block at 3BE/C5
F292    STA     &03BC,Y ;from LOAD and EXEC address
F295    STA     &00AE,Y ;make second copy at B0-B8
F298    INY             ;Y=Y+1
F299    CPY     #&0A    ;until Y=10
F29B    BNE     &F290   ;

F29D    PLA             ;get back A
F29E    BEQ     &F2A7   ;if A=0 F2A7
F2A0    CMP     #&FF    ;else if A<>&FF
F2A2    BNE     &F254   ;RETURN as cassette has no other options
F2A4    JMP     &F1C4   ;load file


************** Save a file ********************************************

F2A7    STA     &03C6   ;zero block number
F2AA    STA     &03C7   ;zero block number hi

F2AD    LDA     (&C8),Y ;OSFILE parameter  block
F2AF    STA     &00A6,Y ;store to Zero page copy (&B0 to &B7)
F2B2    INY             ;data start and data end address
F2B3    CPY     #&12    ;until Y=18
F2B5    BNE     &F2AD   ;
F2B7    TXA             ;A=X
F2B8    BEQ     &F274   ;if X=0 no filename found so B274 else BAD STRING error

F2BA    JSR     &FB27   ;Set cassette option sinto (BB),set C7=6
                        ;claim serial system for cassette
F2BD    JSR     &F934   ;prompt to start recording

F2C0    LDA     #&00    ;A=0
F2C2    JSR     &FBBD   ;enable 2nd proc. if present and set up osfile block
F2C5    JSR     &FBE2   ;set up CFS for write operation
F2C8    SEC             ;set carry flag
F2C9    LDX     #&FD    ;X=&FD

F2CB    LDA     &FFB7,X ;set 03C8/A block length and block flag
F2CE    SBC     &FFB3,X ;to B4/6-B0/2 the number of pages (blocks) to be
                        ;saved
F2D1    STA     &02CB,X ;
F2D4    INX             ;X=X+1
F2D5    BNE     &F2CB   ;

F2D7    TAY             ;Y=A
F2D8    BNE     &F2E8   ;if last byte is non zero F2E8 else
```

```
F2DA    CPX     &03C8   ;compare X with block length
F2DD    LDA     #&01    ;A=1
F2DF    SBC     &03C9   ;subtract block length hi
F2E2    BCC     &F2E8   ;if carry clear F2E8

F2E4    LDX     #&80    ;X=&80
F2E6    BNE     &F2F0   ;jump F2F0

F2E8    LDA     #&01    ;A=1
F2EA    STA     &03C9   ;block length hi
F2ED    STX     &03C8   ;block length
F2F0    STX     &03CA   ;block flag
F2F3    JSR     &F7EC   ;write block to Tape
F2F6    BMI     &F341   ;return if negative
F2F8    JSR     &F96A   ;increment current load address
F2FB    INC     &03C6   ;block number
F2FE    BNE     &F2C8   ;if not 0 loop back again else
F300    INC     &03C7   ;block number hi
F303    BNE     &F2C8   ;and loop back again


*************************************************************************
*                                                                       *
*       *RUN    ENTRY                                                   *
*                                                                       *
*************************************************************************


F305    JSR     &F25A   ;get filename from BUFFER
F308    LDX     #&FF    ;X=&FF
F30A    STX     &03C2   ;execution address
F30D    JSR     &F1C4   ;load file
F310    BIT     &027A   ;&FF if tube present
F313    BPL     &F31F   ;so if not present F31F else
F315    LDA     &03C4   ;execution address extend
F318    AND     &03C5   ;execution address extend
F31B    CMP     #&FF    ;if they are NOT both &FF i.e.for base processor
F31D    BNE     &F322   ;F322 else
F31F    JMP     (&03C2) ; RUN file

F322    LDX     #&C2    ;point to execution address
F324    LDY     #&03    ;(&3C2)
F326    LDA     #&04    ;Tube call 4
F328    JMP     &FBC7   ;and issue to Tube to run file




*************************************************************************
*                                                                       *
*       *CAT    ENTRY                                                   *
*                                                                       *
*************************************************************************

        ;CASSETTE OPTIONS in &E2

        ;bit 0  input file open
        ;bit 1  output file open
        ;bit 2,4,5  not used
        ;bit 3  current CATalogue status
        ;bit 6  EOF reached
        ;bit 7  EOF warning given
```

```
F32B    LDA     #&08    ;A=8
F32D    JSR     &F344   ;set status bits from A
F330    JSR     &FB27   ;Set cassette options into (BB),set C7=6
                        ;claim serial system for cassette
F333    LDA     #&00    ;A=0
F335    JSR     &F348   ;read data from CFS/RFS
F338    JSR     &FAFC   ;perform read
F33B    LDA     #&F7    ;A=&F7
F33D    AND     &E2     ;clear bit 3 of CFS status bit
F33F    STA     &E2     ;
F341    RTS             ;return
        ;

F342    LDA     #&40    ;set bit 6 of E2 cassette options
F344    ORA     &E2     ;
F346    BNE     &F33F   ;and Jump F33F


********** search routine ***********************************************

F348    PHA             ;save A on stack
F349    LDA     &0247   ;filing system flag 0=CFS 2=RFS
F34C    BEQ     &F359   ;if CFS F359 else
F34E    JSR     &EE13   ;set current Filing System ROM/PHROM
F351    JSR     &EE18   ;get byte from data Romcheck type
F354    BCC     &F359   ;if carry clear F359 else
F356    CLV             ;clear overflow flag
F357    BVC     &F39A   ;JUMP F39A


********** cassette routine*********************************************

F359    JSR     &F77B   ;read block header
F35C    LDA     &03C6   ;block number
F35F    STA     &B4     ;current block no. lo
F361    LDA     &03C7   ;block number hi
F364    STA     &B5     ;current block no. hi
F366    LDX     #&FF    ;X=&FF
F368    STX     &03DF   ;copy of last read block flag
F36B    INX             ;X=X+1
F36C    STX     &BA     ;current block flag
F36E    BEQ     &F376   ;if  0 F376

F370    JSR     &FB69   ;inc. current block no.
F373    JSR     &F77B   ;read block header
F376    LDA     &0247   ;get filing system flag 0=CFS 2=RFS
F379    BEQ     &F37D   ;if CFS F37D
F37B    BVC     &F39A   ;if V clear F39A
F37D    PLA             ;get back A
F37E    PHA             ;save A on stack
F37F    BEQ     &F3AE   ;if A=0 F3AE
F381    JSR     &FA72   ;else check filename header block matches searched Fn
F384    BNE     &F39C   ;if so F39C
F386    LDA     #&30    ;else A=30 to clear all but bits 4/5 of current OPTions
F388    AND     &BB     ;current OPTions
F38A    BEQ     &F39A   ;if 0 F39A else

F38C    LDA     &03C6   ;block number
F38F    CMP     &B6     ;next block no. lo
F391    BNE     &F39C   ;
F393    LDA     &03C7   ;block number hi
F396    CMP     &B7     ;next block no. hi
F398    BNE     &F39C   ;
F39A    PLA             ;get back A
F39B    RTS             ;return
```

```
        ;
F39C    LDA     &0247   ;filing system flag 0=CFS 2=RFS
F39F    BEQ     &F3AE   ;if tape F3AE
F3A1    JSR     &EEAD   ;else set ROM displacement address

F3A4    LDA     #&FF    ;A=&FF
F3A6    STA     &03C6   ;block number
F3A9    STA     &03C7   ;block number hi
F3AC    BNE     &F370   ;jump F370

F3AE    BVC     &F3B5   ;if carry clear F3B5
F3B0    LDA     #&FF    ;A=&FF
F3B2    JSR     &F7D7   ;set flags
F3B5    LDX     #&00    ;X=0
F3B7    JSR     &F9D9   ;report 'DATA?'
F3BA    LDA     &0247   ;filing system flag 0=CFS 2=RFS
F3BD    BEQ     &F3C3   ;
F3BF    BIT     &BB     ;current OPTions
F3C1    BVC     &F3A1   ;long messages not required if BIT 6 =0
F3C3    BIT     &03CA   ;block flag
F3C6    BMI     &F3A4   ;if -ve F3A4
F3C8    BPL     &F370   ;else loop back and do it again



*************************************************************************
*                                                                       *
*       OSFIND  ENTRY                                                   *
*       file handling                                                   *
*                                                                       *
*************************************************************************
;on entry A determines Action Y may contain file handle or
;X/Y point to filename terminated by &0D in memory
;A=0    closes file in channel Y if Y=0 closes all files
;A=&40  open a file for input  (reading) X/Y points to filename
;A=&80  open a file for output (writing) X/Y points to filename
;A=&C0  open a file for input and output (random Access)
;ON EXIT Y=0 if no file found else Y=channel number in use for file

                        ;save A X and Y
F3CA    STA     &BC     ;file status or temporary store
F3CC    TXA             ;A=X
F3CD    PHA             ;save X on stack
F3CE    TYA             ;A=Y
F3CF    PHA             ;save Y on stack
F3D0    LDA     &BC     ;file status or temporary store
F3D2    BNE     &F3F2   ;if A is non zero open a file via F3F2


************ close a file ***********************************************

F3D4    TYA             ;A=Y
F3D5    BNE     &F3E3   ;if A<> 0 close specified file else close them all
F3D7    JSR     &E275   ;close all files via OSBYTE 77
F3DA    JSR     &F478   ;tidy up
F3DD    LSR     &E2     ;CFS status byte is shifted left and right to zero
F3DF    ASL     &E2     ;bit 0
F3E1    BCC     &F3EF   ;and if carry clear no input file was open so F3EF

F3E3    LSR             ;A contains file handle so shift bit 0 into carry
F3E4    BCS     &F3DD   ;if carry set close input file
F3E6    LSR             ;else shift bit 1 into carry
F3E7    BCS     &F3EC   ;if carry set close output file
F3E9    JMP     &FBB1   ;else report 'Channel Error' as CFS can only support
                        ;1 input and 1 output file
```

```
F3EC    JSR     &F478   ;tidy up
F3EF    JMP     &F471   ;and exit


************ OPEN A FILE ***********************************************

F3F2    JSR     &F25A   ;get filename from BUFFER
F3F5    BIT     &BC     ;file status or temporary store
F3F7    BVC     &F436   ;check A at input if bit 6 not set its an output file

********* Input files +*********************************************

F3F9    LDA     #&00    ;else its an input file
F3FB    STA     &039E   ;BGET buffer offset for next byte
F3FE    STA     &03DD   ;Expected BGET file block number lo
F401    STA     &03DE   ;expected BGET file block number hi
F404    LDA     #&3E    ;A=&3E
F406    JSR     &F33D   ;CFS status =CFS status AND A
F409    JSR     &FB1A   ;claim serial system and set OPTions
F40C    PHP             ;save flags on stack
F40D    JSR     &F631   ;search for file
F410    JSR     &F6B4   ;check protection bit of block status and respond
F413    PLP             ;get back flags
F414    LDX     #&FF    ;X=&FF increment to 0 on next instruction

F416    INX             ;X=X+1
F417    LDA     &03B2,X ;get file name and
F41A    STA     &03A7,X ;store as BGET filename
F41D    BNE     &F416   ;until end of filename

F41F    LDA     #&01    ;A=1 to show file open
F421    JSR     &F344   ;set status bits from A
F424    LDA     &02EA   ;CFS currently resident file block length lo
F427    ORA     &02EB   ;CFS currently resident file block length hi
F42A    BNE     &F42F   ;if block length is 0
F42C    JSR     &F342   ;set CFS status bit 3 (EOF reached)
                        ;else
F42F    LDA     #&01    ;A=1
F431    ORA     &0247   ;filing system flag 0=CFS 2=RFS
F434    BNE     &F46F   ;and exit after restoring registers


****************** open an output file*********************************

F436    TXA             ;A=X

F437    BNE     &F43C   ;if X=0 then zero length filename so
F439    JMP     &EA8F   ;Bad String error

F43C    LDX     #&FF    ;X=&FF
F43E    INX             ;X=X+1
                        ;copy sought filename to header block
F43F    LDA     &03D2,X ;sought filename
F442    STA     &0380,X ;BPUT file header block
F445    BNE     &F43E   ;until A=0 end of filename
F447    LDA     #&FF    ;A=&FF
F449    LDX     #&08    ;X=8

F44B    STA     &038B,X ;set 38C/93 to &FF
F44E    DEX             ;X=X-1
F44F    BNE     &F44B   ;

F451    TXA             ;A=X=0
F452    LDX     #&14    ;X=14
```

```
F454    STA     &0380,X ;BPUT file header block
F457    INX             ;X=X+1
F458    CPX     #&1E    ;this zeros 394/D
F45A    BNE     &F454   ;

F45C    ROL     &0397   ;
F45F    JSR     &FB27   ;Set cassette optionsinto (BB),set C7=6
                        ;claim serial system for cassette
F462    JSR     &F934   ;prompt to start recording
F465    JSR     &FAF2   ;enable second processor and reset serial system
F468    LDA     #&02    ;A=2
F46A    JSR     &F344   ;set status bits from A
F46D    LDA     #&02    ;A=2
F46F    STA     &BC     ;file status or temporary store
F471    PLA             ;get back A
F472    TAY             ;Y=A
F473    PLA             ;get back A
F474    TAX             ;X=A
F475    LDA     &BC     ;file status or temporary store
F477    RTS             ;return
        ;

F478    LDA     #&02    ;A=2 clearing all but bit 1 of status byte
F47A    AND     &E2     ;CFS status byte, with output file open
F47C    BEQ     &F477   ;if file not open then exit
F47E    LDA     #&00    ;else A=0
F480    STA     &0397   ;setting block length to current value of BPUT offset
F483    LDA     #&80    ;A=&80
F485    LDX     &039D   ;get BPUT buffer ofset
F488    STX     &0396   ;setting block length to current value of BPUT offset
F48B    STA     &0398   ;mark current block as last
F48E    JSR     &F496   ;save block to tape
F491    LDA     #&FD    ;A=&FD
F493    JMP     &F33D   ;CFS status =CFS status AND A


********** SAVE BLOCK TO TAPE ******************************************

F496    JSR     &FB1A   ;claim serial system and set OPTions

F499    LDX     #&11    ;X=11
F49B    LDA     &038C,X ;copy header block from 38C-39D
F49E    STA     &03BE,X ;to 3BE/DF
F4A1    DEX             ;X=X-1
F4A2    BPL     &F49B   ;
                        ;X=&FF
F4A4    STX     &B2     ;current load address high word
F4A6    STX     &B3     ;current load address high word
F4A8    INX             ;X=X+1, (X=0)
F4A9    STX     &B0     ;current load address lo byte set to &00
F4AB    LDA     #&09    ;A=9 to set current load address at &900
F4AD    STA     &B1     ;current load address
F4AF    LDX     #&7F    ;X=&7F
F4B1    JSR     &FB81   ;copy from 301/C+X to 3D2/C sought filename
F4B4    STA     &03DF   ;copy of last read block flag
F4B7    JSR     &FB8E   ;switch Motor On
F4BA    JSR     &FBE2   ;set up CFS for write operation
F4BD    JSR     &F7EC   ;write block to Tape
F4C0    INC     &0394   ;block number lo
F4C3    BNE     &F4C8   ;
F4C5    INC     &0395   ;block number hi
F4C8    RTS             ;return
```

```
*************************************************************************
*                                                                       *
*                                                                       *
*       OSBGET  get a byte from a file                                  *
*                                                                       *
*                                                                       *
*************************************************************************
        ;on ENTRY       Y contains channel number
        ;on EXIT        X and Y are preserved C=0 indicates valid character
        ;               A contains character (or error) A=&FE End Of File

                        ;push X and Y
F4C9    TXA             ;A=X
F4CA    PHA             ;save A on stack
F4CB    TYA             ;A=Y
F4CC    PHA             ;save A on stack
F4CD    LDA     #&01    ;A=1
F4CF    JSR     &FB9C   ;check conditions for OSBGET are OK
F4D2    LDA     &E2     ;CFS status byte
F4D4    ASL             ;shift bit 7 into carry (EOF warning given)
F4D5    BCS     &F523   ;if carry set F523
F4D7    ASL             ;shift bit 6 into carry
F4D8    BCC     &F4E3   ;if clear EOF not reached F4E3
F4DA    LDA     #&80    ;else A=&80 setting bit 7 of status byte EOF warning
F4DC    JSR     &F344   ;set status bits from A
F4DF    LDA     #&FE    ;A=&FE
F4E1    BCS     &F51B   ;if carry set F51B

F4E3    LDX     &039E   ;BGET buffer offset for next byte
F4E6    INX             ;X=X+1
F4E7    CPX     &02EA   ;CFS currently resident file block length lo
F4EA    BNE     &F516   ;read a byte
                        ;else
F4EC    BIT     &02EC   ;block flag of currently resident block
F4EF    BMI     &F513   ;if bit 7=1 this is the last block so F513 else
F4F1    LDA     &02ED   ;last character of currently resident block
F4F4    PHA             ;save A on stack
F4F5    JSR     &FB1A   ;claim serial system and set OPTions
F4F8    PHP             ;save flags on stack
F4F9    JSR     &F6AC   ;read in a new block
F4FC    PLP             ;get back flags
F4FD    PLA             ;get back A
F4FE    STA     &BC     ;file status or temporary store
F500    CLC             ;clear carry flag
F501    BIT     &02EC   ;block flag of currently resident block
F504    BPL     &F51D   ;if not last block (bit 7=0)
F506    LDA     &02EA   ;CFS currently resident file block length lo
F509    ORA     &02EB   ;CFS currently resident file block length hi
F50C    BNE     &F51D   ;if block size not 0 F51D else
F50E    JSR     &F342   ;set CFS status bit 6 (EOF reached)
F511    BNE     &F51D   ;goto F51D

F513    JSR     &F342   ;set CFS status bit 6 (EOF reached)
F516    DEX             ;X=X-1
F517    CLC             ;clear carry flag
F518    LDA     &0A00,X ;read byte from cassette buffer

F51B    STA     &BC     ;file status or temporary store
F51D    INC     &039E   ;BGET buffer offset for next byte
F520    JMP     &F471   ;exit via F471
```

```
F523    BRK             ;
F524    DB      &DF     ;error number
F525    DB      'EOF'   ;
F528    BRK             ;


*************************************************************************
*                                                                       *
*                                                                       *
*       OSBPUT  WRITE A BYTE TO FILE                                    *
*                                                                       *
*                                                                       *
*************************************************************************
;ON ENTRY  Y contains channel number A contains byte to be written

F529    STA     &C4     ;store A in temorary store
F52B    TXA             ;and stack X and Y
F52C    PHA             ;save  on stack
F52D    TYA             ;A=Y
F52E    PHA             ;save on stack

F52F    LDA     #&02    ;A=2
F531    JSR     &FB9C   ;check conditions necessary for OSBPUT are OK
F534    LDX     &039D   ;BPUT buffer offset for next byte
F537    LDA     &C4     ;get back original value of A
F539    STA     &0900,X ;Cassette buffer
F53C    INX             ;X=X+1
F53D    BNE     &F545   ;if not 0 F545, otherwise buffer is full so
F53F    JSR     &F496   ;save block to tape
F542    JSR     &FAF2   ;enable second processor and reset serial system
F545    INC     &039D   ;BPUT buffer offset for next byte
F548    LDA     &C4     ;get back A
F54A    JMP     &F46F   ;and exit


*************************************************************************
*                                                                       *
*                                                                       *
*       OSBYTE 139      Select file options                             *
*                                                                       *
*                                                                       *
*************************************************************************
;ON ENTRY  Y contains option value  X contains option No. see *OPT X,Y
;this applies largely to CFS LOAD SAVE CAT and RUN
;X=1    is message switch
;       Y=0     no messages
;       Y=1     short messages
;       Y=2     gives detailed information on load and execution addresses

;X=2    is error handling
;       Y=0     ignore errors
;       Y=1     prompt for a retry
;       Y=2     abort operation

;X=3    is interblock gap for BPUT# and PRINT#
;       Y=0-127 set gap in 1/10ths Second
;       Y > 127 use default values


F54D    TXA             ;A=X
F54E    BEQ     &F57E   ;if A=0 F57E
F550    CPX     #&03    ;if X=3
F552    BEQ     &F573   ;F573 to set interblock gap
F554    CPY     #&03    ;else if Y>2 then BAD COMMAND error
```

```
F556    BCS     &F55E   ;
F558    DEX             ;X=X-1
F559    BEQ     &F561   ;i.e. if X=1 F561 message control
F55B    DEX             ;X=X-1
F55C    BEQ     &F568   ;i.e. if X=2 F568 error response
F55E    JMP     &E310   ;else E310 to issue Bad Command error


********** message control ********************************************

F561    LDA     #&33    ;to set lower two bits of each nybble as mask
F563    INY             ;Y=Y+1
F564    INY             ;Y=Y+1
F565    INY             ;Y=Y+1
F566    BNE     &F56A   ;goto F56A



********** error response *********************************************

F568    LDA     #&CC    ;setting top two bits of each nybble as mask
F56A    INY             ;Y=Y+1
F56B    AND     &E3     ;clear lower two bits of each nybble
F56D    ORA     &F581,Y ;or with table value
F570    STA     &E3     ;store it in &E3
F572    RTS             ;return

        ;setting of &E3
        ;
        ;lower nybble sets LOAD options
        ;upper sets SAVE options

        ;0000   Ignore errors,          no messages
        ;0001   Abort if error,         no messages
        ;0010   Retry after error,      no messages
        ;1000   Ignore error            short messages
        ;1001   Abort if error          short messages
        ;1010   Retry after error       short messages
        ;1100   Ignore error            long messages
        ;1101   Abort if error          long messages
        ;1110   Retry after error       long messages


**********set interblock gap ******************************************

F573    TYA             ;A=Y
F574    BMI     &F578   ;if Y>127 use default values
F576    BNE     &F57A   ;if Y<>0 skip next instruction
F578    LDA     #&19    ;else A=&19

F57A    STA     &03D1   ;sequential block gap
F57D    RTS             ;return
        ;
F57E    TAY             ;Y=A
F57F    BEQ     &F56D   ;jump to F56D



********** DEFAULT OPT VALUES TABLE **********************************

F581    DB      &A1     ;%1010 0001
F582    DB      &00     ;%0000 0000
F583    DB      &22     ;%0010 0010
```

```
F584    DB      &11     ;%0001 0001
F585    DB      &00     ;%0000 0000
F586    DB      &88     ;%1000 1000
F587    DB      &CC     ;%1100 1100

F588    DEC     &C0     ;filing system buffer flag
F58A    LDA     &0247   ;filing system flag 0=CFS 2=RFS
F58D    BEQ     &F596   ;if CFS F596

F58F    JSR     &EE51   ;read RFS data rom or Phrom
F592    TAY             ;Y=A
F593    CLC             ;clear carry flag
F594    BCC     &F5B0   ;jump to F5B0

F596    LDA     &FE08   ;ACIA status register
F599    PHA             ;save A on stack
F59A    AND     #&02    ;clear all but bits 0,1 A=(0-3)
F59C    BEQ     &F5A9   ;if 0 F5A9 transmit data register full or RDR empty
F59E    LDY     &CA     ;
F5A0    BEQ     &F5A9   ;
F5A2    PLA             ;get back A
F5A3    LDA     &BD     ;character temporary storage
F5A5    STA     &FE09   ;ACIA transmit data register
F5A8    RTS             ;return
        ;
F5A9    LDY     &FE09   ;read ACIA recieve data register
F5AC    PLA             ;get back A
F5AD    LSR             ;bit 2 to carry (data carrier detect)
F5AE    LSR             ;
F5AF    LSR             ;

F5B0    LDX     &C2     ;progress flag
F5B2    BEQ     &F61D   ;if &C2=0 exit
F5B4    DEX             ;X=X-1
F5B5    BNE     &F5BD   ;if &C2>1 then F5BD
F5B7    BCC     &F61D   ;else if carrier tone from cassette detected exit

F5B9    LDY     #&02    ;Y=2

F5BB    BNE     &F61B   ;
F5BD    DEX             ;X=X-1
F5BE    BNE     &F5D3   ;if &C2>2
F5C0    BCS     &F61D   ;if carrier tone from cassette not detected  exit
F5C2    TYA             ;A=Y
F5C3    JSR     &FB78   ;set (BE/C0) to 0
F5C6    LDY     #&03    ;Y=3
F5C8    CMP     #&2A    ;is A= to synchronising byte &2A?
F5CA    BEQ     &F61B   ;if so F61B
F5CC    JSR     &FB50   ;control cassette system
F5CF    LDY     #&01    ;Y=1
F5D1    BNE     &F61B   ;goto F61B
F5D3    DEX             ;X=X-1
F5D4    BNE     &F5E2   ;if &C2>3
F5D6    BCS     &F5DC   ;
F5D8    STY     &BD     ;get character read into Y
F5DA    BEQ     &F61D   ;if 0 exit via F61D
F5DC    LDA     #&80    ;else A=&80
F5DE    STA     &C0     ;filing system buffer flag
F5E0    BNE     &F61D   ;and exit

F5E2    DEX             ;X=X-1
F5E3    BNE     &F60E   ;if &C2>4 F60E
F5E5    BCS     &F616   ;if carry set F616
F5E7    TYA             ;A=Y
F5E8    JSR     &F7B0   ;perform CRC
F5EB    LDY     &BC     ;file status or temporary store
```

```
F5ED    INC     &BC     ;file status or temporary store
F5EF    BIT     &BD     ;if bit 7 set this is the last byte read
F5F1    BMI     &F600   ;so F600
F5F3    JSR     &FBD3   ;check if second processor file test tube prescence
F5F6    BEQ     &F5FD   ;if return with A=0 F5FD
F5F8    STX     &FEE5   ;Tube FIFO3
F5FB    BNE     &F600   ;

F5FD    TXA             ;A=X restore value
F5FE    STA     (&B0),Y ;store to current load address
F600    INY             ;Y=Y+1
F601    CPY     &03C8   ;block length
F604    BNE     &F61D   ;exit
F606    LDA     #&01    ;A=1
F608    STA     &BC     ;file status or temporary store
F60A    LDY     #&05    ;Y=5
F60C    BNE     &F61B   ;exit

F60E    TYA             ;A=Y
F60F    JSR     &F7B0   ;perform CRC
F612    DEC     &BC     ;file status or temporary store
F614    BPL     &F61D   ;exit

F616    JSR     &FB46   ;reset ACIA
F619    LDY     #&00    ;Y=0
F61B    STY     &C2     ;progress flag
F61D    RTS             ;return


*************************************************************************
*                                                                       *
*       OSBYTE 127 check for end of file                                *
*                                                                       *
*************************************************************************
        ;
F61E    PHA             ;save A on stack
F61F    TYA             ;A=Y
F620    PHA             ;save Y on stack
F621    TXA             ;A=X to put X into Y
F622    TAY             ;Y=A
F623    LDA     #&03    ;A=3
F625    JSR     &FB9C   ;confirm file is open
F628    LDA     &E2     ;CFS status byte
F62A    AND     #&40    ;
F62C    TAX             ;X=A
F62D    PLA             ;get back A
F62E    TAY             ;Y=A
F62F    PLA             ;get back A
F630    RTS             ;return
        ;
F631    LDA     #&00    ;A=0
F633    STA     &B4     ;current block no. lo
F635    STA     &B5     ;current block no. hi
F637    LDA     &B4     ;current block no. lo
F639    PHA             ;save A on stack
F63A    STA     &B6     ;next block no. lo
F63C    LDA     &B5     ;current block no. hi
F63E    PHA             ;save A on stack
F63F    STA     &B7     ;next block no. hi
F641    JSR     &FA46   ;print message following call

F644    DB      'Searching';
F64C    DB      &0D     ;newline
F64E    BRK             ;

F64F    LDA     #&FF    ;A=&FF
```

```
F651    JSR     &F348   ;read data from CFS/RFS
F654    PLA             ;get back A
F655    STA     &B5     ;current block no. hi
F657    PLA             ;get back A
F658    STA     &B4     ;current block no. lo
F65A    LDA     &B6     ;next block no. lo
F65C    ORA     &B7     ;next block no. hi
F65E    BNE     &F66D   ;
F660    STA     &B4     ;current block no. lo
F662    STA     &B5     ;current block no. hi
F664    LDA     &C1     ;checksum result
F666    BNE     &F66D   ;
F668    LDX     #&B1    ;current load address
F66A    JSR     &FB81   ;copy from 301/C+X to 3D2/C sought filename
F66D    LDA     &0247   ;filing system flag 0=CFS 2=RFS
F670    BEQ     &F685   ;if cassette F685
F672    BVS     &F685   ;

F674    BRK             ;
F675    DB      &D6             ;Error number
F676    DB      'File Not found'
F684    BRK             ;

F685    LDY     #&FF    ;Y=&FF
F687    STY     &03DF   ;copy of last read block flag
F68A    RTS             ;return
        ;

********        * EXEC  0 ***********************************************

F68B    LDA     #&00    ;A=0



**************************************************************************
*                                                                        *
*       * EXEC                                                           *
*                                                                        *
**************************************************************************

F68D    PHP             ;save flags on stack
F68E    STY     &E6     ;&E6=Y
F690    LDY     &0256   ;EXEC file handle
F693    STA     &0256   ;EXEC file handle
F696    BEQ     &F69B   ;if not 0 close file via OSFIND
F698    JSR     OSFIND  ;
F69B    LDY     &E6     ;else Y= original Y
F69D    PLP             ;get back flags
F69E    BEQ     &F6AB   ;if A=0 on entry exit else
F6A0    LDA     #&40    ;A=&40
F6A2    JSR     OSFIND  ;to open an input file
F6A5    TAY             ;Y=A
F6A6    BEQ     &F674   ;If Y=0 'File not found' else store
F6A8    STA     &0256   ;EXEC file handle
F6AB    RTS             ;return

******* read a block    ***********************************************

F6AC    LDX     #&A6    ;X=&A6
F6AE    JSR     &FB81   ;copy from 301/C+X to 3D2/C sought filename
F6B1    JSR     &F77B   ;read block header
F6B4    LDA     &03CA   ;block flag
F6B7    LSR             ;A=A/2 bit 0 into carry to check for locked file
F6B8    BCC     &F6BD   ;if not set then skip next instruction
F6BA    JMP     &F1F6   ;'locked' file routine
```

```
F6BD    LDA     &03DD   ;Expected BGET file block number lo
F6C0    STA     &B4     ; current block no. lo
F6C2    LDA     &03DE   ;expected BGET file block number hi
F6C5    STA     &B5     ;current block no. hi
F6C7    LDA     #&00    ;A=0
F6C9    STA     &B0     ;current load address
F6CB    LDA     #&0A    ;A=&A setting current load address to the CFS/RFS
F6CD    STA     &B1     ;current load address buffer at &A00
F6CF    LDA     #&FF    ;A=&FF to set other 2 bytes
F6D1    STA     &B2     ;current load address high word
F6D3    STA     &B3     ;current load address high word
F6D5    JSR     &F7D5   ;reset flags
F6D8    JSR     &F9B4   ;load file from tape
F6DB    BNE     &F702   ;if return non zero F702 else
F6DD    LDA     &0AFF   ;get last character from input buffer
F6E0    STA     &02ED   ;last character currently resident block
F6E3    JSR     &FB69   ;inc. current block no.
F6E6    STX     &03DD   ;expected BGET file block number lo
F6E9    STY     &03DE   ;expected BGET file block number hi
F6EC    LDX     #&02    ;X=2
F6EE    LDA     &03C8,X ;read bytes from block flag/block length
F6F1    STA     &02EA,X ;store into current values of above
F6F4    DEX             ;X=X-1
F6F5    BPL     &F6EE   ;until X=-1 (&FF)

F6F7    BIT     &02EC   ;block flag of currently resident block
F6FA    BPL     &F6FF   ;
F6FC    JSR     &F249   ;print newline if needed
F6FF    JMP     &FAF2   ;enable second processor and reset serial system
F702    JSR     &F637   ;search for a specified block
F705    BNE     &F6B4   ;if NE check for locked condition else
F707    CMP     #&2A    ;is it Synchronising byte &2A?
F709    BEQ     &F742   ;if so F742
F70B    CMP     #&23    ;else is it &23 (header substitute in ROM files)
F70D    BNE     &F71E   ;if not BAD ROM error

F70F    INC     &03C6   ;block number
F712    BNE     &F717   ;
F714    INC     &03C7   ;block number hi
F717    LDX     #&FF    ;X=&FF
F719    BIT     &D9B7   ;to set V & M
F71C    BNE     &F773   ;and jump (ALWAYS!!) to F773

F71E    LDA     #&F7    ;clear bit 3 of RFS status (current CAT status)
F720    JSR     &F33D   ;RFS status =RFS status AND A

F723    BRK             ;and cause error
F724    DB      &D7     ;error number
F725    DB      'Bad Rom'
F72C    BRK             ;


**********: pick up a header ******************************************

F72D    LDY     &FF     ;get ESCAPE flag
F72F    JSR     &FB90   ;switch Motor on
F732    LDA     #&01    ;A=1
F734    STA     &C2     ;progress flag
F736    JSR     &FB50   ;control serial system
F739    JSR     &F995   ;confirm ESC not set and CFS not executing
F73C    LDA     #&03    ;A=3
F73E    CMP     &C2     ;progress flag
F740    BNE     &F739   ;back until &C2=3

F742    LDY     #&00    ;Y=0
F744    JSR     &FB7C   ;zero checksum bytes
```

```
F747    JSR     &F797   ;get character from file and do CRC
F74A    BVC     &F766   ;if V clear on exit F766
F74C    STA     &03B2,Y ;else store
F74F    BEQ     &F757   ;or if A=0 F757
F751    INY             ;Y=Y+1
F752    CPY     #&0B    ;if Y<>&B
F754    BNE     &F747   ;go back for next character
F756    DEY             ;Y=Y-1

F757    LDX     #&0C    ;X=12
F759    JSR     &F797   ;get character from file and do CRC
F75C    BVC     &F766   ;if V clear on exit F766
F75E    STA     &03B2,X ;else store byte
F761    INX             ;X=X+1
F762    CPX     #&1F    ;if X<>31
F764    BNE     &F759   ;goto F759

F766    TYA             ;A=Y
F767    TAX             ;X=A
F768    LDA     #&00    ;A=0
F76A    STA     &03B2,Y ;store it
F76D    LDA     &BE     ;CRC workspace
F76F    ORA     &BF     ;CRC workspace
F771    STA     &C1     ;Checksum result
F773    JSR     &FB78   ;set (BE/C0) to 0
F776    STY     &C2     ;progress flag
F778    TXA             ;A=X
F779    BNE     &F7D4   ;
F77B    LDA     &0247   ;filing system flag 0=CFS 2=RFS
F77E    BEQ     &F72D   ;if cassette F72D
F780    JSR     &EE51   ;read RFS data rom or Phrom
F783    CMP     #&2B    ;is it ROM file terminator?
F785    BNE     &F707   ;if not F707

********* terminator found ***********************************************

F787    LDA     #&08    ;A=8 isolating bit 3 CAT status
F789    AND     &E2     ;CFS status byte
F78B    BEQ     &F790   ;if clera skip next instruction
F78D    JSR     &F24D   ;print CR if CFS not operational
F790    JSR     &EE18   ;get byte from data Rom
F793    BCC     &F780   ;if carry set F780
F795    CLV             ;clear overflow flag
F796    RTS             ;return



***************  get character from file and do CRC  ******************
        ;
F797    LDA     &0247   ;filing system flag 0=CFS 2=RFS
F79A    BEQ     &F7AD   ;if cassette F7AD
F79C    TXA             ;A=X to save X and Y
F79D    PHA             ;save X on stack
F79E    TYA             ;A=Y
F79F    PHA             ;save Y on stack
F7A0    JSR     &EE51   ;read RFS data rom or Phrom
F7A3    STA     &BD     ;put it in temporary storage
F7A5    LDA     #&FF    ;A=&FF
F7A7    STA     &C0     ;filing system buffer flag
F7A9    PLA             ;get back Y
F7AA    TAY             ;Y=A
F7AB    PLA             ;get back X
F7AC    TAX             ;X=A
F7AD    JSR     &F884   ;check for Escape and loop till bit 7 of FS buffer
                        ;flag=1
```

```
************************* perform CRC *********************************

F7B0    PHP             ;save flags on stack
F7B1    PHA             ;save A on stack
F7B2    SEC             ;set carry flag
F7B3    ROR     &CB     ;CRC Bit counter
F7B5    EOR     &BF     ;CRC workspace
F7B7    STA     &BF     ;CRC workspace
F7B9    LDA     &BF     ;CRC workspace
F7BB    ROL             ;A=A*2 C=bit 7
F7BC    BCC     &F7CA   ;
F7BE    ROR             ;A=A/2
F7BF    EOR     #&08    ;
F7C1    STA     &BF     ;CRC workspace
F7C3    LDA     &BE     ;CRC workspace
F7C5    EOR     #&10    ;
F7C7    STA     &BE     ;CRC workspace
F7C9    SEC             ;set carry flag

F7CA    ROL     &BE     ;CRC workspace
F7CC    ROL     &BF     ;CRC workspace
F7CE    LSR     &CB     ;CRC Bit counter
F7D0    BNE     &F7B9   ;
F7D2    PLA             ;get back A
F7D3    PLP             ;get back flags
F7D4    RTS             ;return
        ;


F7D5    LDA     #&00    ;A=0
F7D7    STA     &BD     ;&BD=character temporary storage buffer=0
F7D9    LDX     #&00    ;X=0
F7DB    STX     &BC     ;file status or temporary store
F7DD    BVC     &F7E9   ;
F7DF    LDA     &03C8   ;block length
F7E2    ORA     &03C9   ;block length hi
F7E5    BEQ     &F7E9   ;if 0 F7E9

F7E7    LDX     #&04    ;else X=4
F7E9    STX     &C2     ;filename length/progress flag
F7EB    RTS             ;return


*************** SAVE A BLOCK ***************************************

F7EC    PHP             ;save flags on stack
F7ED    LDX     #&03    ;X=3
F7EF    LDA     #&00    ;A=0
F7F1    STA     &03CB,X ;clear 03CB/E (RFS EOF+1?)
F7F4    DEX             ;X=X-1
F7F5    BPL     &F7F1   ;

F7F7    LDA     &03C6   ;block number
F7FA    ORA     &03C7   ;block number hi
F7FD    BNE     &F804   ;if block =0 F804 else
F7FF    JSR     &F892   ;generate a 5 second delay
F802    BEQ     &F807   ;goto F807


F804    JSR     &F896   ;generate delay set by interblock gap
F807    LDA     #&2A    ;A=&2A
F809    STA     &BD     ;store it in temporary file
F80B    JSR     &FB78   ;set (BE/C0) to 0
F80E    JSR     &FB4A   ;set ACIA control register
```

```
F811    JSR     &F884   ;check for Escape and loop till bit 7 of FS buffer
                        ;flag=1
F814    DEY             ;Y=Y-1
F815    INY             ;Y=Y+1
F816    LDA     &03D2,Y ;move sought filename
F819    STA     &03B2,Y ;into filename block
F81C    JSR     &F875   ;transfer byte to CFS and do CRC
F81F    BNE     &F815   ;if filename not complet then do it again


******: deal with rest of header ***************************************

F821    LDX     #&0C    ;X=12
F823    LDA     &03B2,X ;get filename byte
F826    JSR     &F875   ;transfer byte to CFS and do CRC
F829    INX             ;X=X+1
F82A    CPX     #&1D    ;until X=29
F82C    BNE     &F823   ;

F82E    JSR     &F87B   ;save checksum to TAPE reset buffer flag
F831    LDA     &03C8   ;block length
F834    ORA     &03C9   ;block length hi
F837    BEQ     &F855   ;if 0 F855

F839    LDY     #&00    ;else Y=0
F83B    JSR     &FB7C   ;zero checksum bytes
F83E    LDA     (&B0),Y ;get a data byte
F840    JSR     &FBD3   ;check if second processor file test tube prescence
F843    BEQ     &F848   ;if not F848 else

F845    LDX     &FEE5   ;Tube FIFO3

F848    TXA             ;A=X
F849    JSR     &F875   ;transfer byte to CFS and do CRC
F84C    INY             ;Y=Y+1
F84D    CPY     &03C8   ;block length
F850    BNE     &F83E   ;
F852    JSR     &F87B   ;save checksum to TAPE reset buffer flag
F855    JSR     &F884   ;check for Escape and loop till bit 7 of FS buffer
                        ;flag=1
F858    JSR     &F884   ;check for Escape and loop till bit 7 of FS buffer
                        ;flag=1
F85B    JSR     &FB46   ;reset ACIA

F85E    LDA     #&01    ;A=1
F860    JSR     &F898   ;generate 0.1 * A second delay
F863    PLP             ;get back flags
F864    JSR     &F8B9   ;update block flag, PRINT filename (& address if reqd)
F867    BIT     &03CA   ;block flag
F86A    BPL     &F874   ;is this last block (bit 7 set)?
F86C    PHP             ;save flags on stack
F86D    JSR     &F892   ;generate a 5 second delay
F870    JSR     &F246   ;sound bell and abort
F873    PLP             ;get back flags
F874    RTS             ;return


****************** transfer byte to CFS and do CRC **********************
        ;
F875    JSR     &F882   ;save byte to buffer, transfer to CFS & reset flag
F878    JMP     &F7B0   ;perform CRC


***************** save checksum to TAPE reset buffer flag ***************

F87B    LDA     &BF     ;CRC workspace
```

```
F87D    JSR     &F882   ;save byte to buffer, transfer to CFS & reset flag
F880    LDA     &BE     ;CRC workspace


************** save byte to buffer, transfer to CFS & reset flag ********

F882    STA     &BD     ;store A in temporary buffer


***** check for Escape and loop till bit 7 of FS buffer flag=1 ***********

F884    JSR     &F995   ;confirm ESC not set and CFS not executing
F887    BIT     &C0     ;filing system buffer flag
F889    BPL     &F884   ;loop until bit 7 of &C0 is set

F88B    LDA     #&00    ;A=0
F88D    STA     &C0     ;filing system buffer flag
F88F    LDA     &BD     ;get temporary store byte
F891    RTS             ;return
        ;


****************** generate a 5 second delay  ***************************

F892    LDA     #&32    ;A=50
F894    BNE     &F898   ;generate delay 100ms *A (5 seconds)



*************** generate delay set by interblock gap ********************

F896    LDA     &C7     ;get current interblock flag


*************** generate delay ***************************************

F898    LDX     #&05    ;X=5
F89A    STA     &0240   ;CFS timeout counter
F89D    JSR     &F995   ;confirm ESC not set and CFS not executing
F8A0    BIT     &0240   ;CFS timeout counter (decremented each 20ms)
F8A3    BPL     &F89D   ;if +ve F89D
F8A5    DEX             ;X=X-1
F8A6    BNE     &F89A   ;
F8A8    RTS             ;return
        ;


************: generate screen reports **********************************

F8A9    LDA     &03C6   ;block number
F8AC    ORA     &03C7   ;block number hi
F8AF    BEQ     &F8B6   ;if 0 F8B6
F8B1    BIT     &03DF   ;copy of last read block flag
F8B4    BPL     &F8B9   ;update block flag, PRINT filename (& address if reqd)
F8B6    JSR     &F249   ;print newline if needed


************** update block flag, PRINT filename (& address if reqd) ****

F8B9    LDY     #&00    ;Y=0
F8BB    STY     &BA     ;current block flag
F8BD    LDA     &03CA   ;block flag
F8C0    STA     &03DF   ;copy of last read block flag
F8C3    JSR     &E7DC   ;check if free to print message
F8C6    BEQ     &F933   ;if A=0 on return Cassette system is busy
F8C8    LDA     #&0D    ;else A=&0D :carriage return
```

```
F8CA    JSR     OSWRCH  ;print it (note no linefeed as its via OSWRCH)
F8CD    LDA     &03B2,Y ;get byte form filename
F8D0    BEQ     &F8E2   ;if 0 filename is ended
F8D2    CMP     #&20    ;if <SPACE
F8D4    BCC     &F8DA   ;F8DA
F8D6    CMP     #&7F    ;if less than DELETE
F8D8    BCC     &F8DC   ;its a printable character for F8DC else

******************Control characters in RFS/CFS filename ******************

F8DA    LDA     #&3F    ;else A='?'
F8DC    JSR     OSWRCH  ;and print it

F8DF    INY             ;Y=Y+1
F8E0    BNE     &F8CD   ;back to get rest of filename

***************** end of filename ************************************

F8E2    LDA     &0247   ;filing system flag 0=CFS 2=RFS
F8E5    BEQ     &F8EB   ;if cassette F8EB
F8E7    BIT     &BB     ;test current OPTions
F8E9    BVC     &F933   ;if bit 6 clear no,long messages needed F933
F8EB    JSR     &F991   ;print a space
F8EE    INY             ;Y=Y+1
F8EF    CPY     #&0B    ;if Y<11 then
F8F1    BCC     &F8E2   ;loop again to fill out filename with spaces

F8F3    LDA     &03C6   ;block number
F8F6    TAX             ;X=A
F8F7    JSR     &F97A   ;print ASCII equivalent of hex byte
F8FA    BIT     &03CA   ;block flag
F8FD    BPL     &F933   ;if not end of file return
F8FF    TXA             ;A=X
F900    CLC             ;clear carry flag
F901    ADC     &03C9   ;block length hi
F904    STA     &CD     ;file length counter hi
F906    JSR     &F975   ;print space + ASCII equivalent of hex byte
F909    LDA     &03C8   ;block length
F90C    STA     &CC     ;file length counter lo
F90E    JSR     &F97A   ;print ASCII equivalent of hex byte
F911    BIT     &BB     ;current OPTions
F913    BVC     &F933   ;if bit 6 clear no long messages required so F933

F915    LDX     #&04    ;X=4
F917    JSR     &F991   ;print a space
F91A    DEX             ;X=X-1
F91B    BNE     &F917   ;loop to print 4 spaces

F91D    LDX     #&0F    ;X=&0F to point to load address
F91F    JSR     &F927   ;print 4 bytes from CFS block header
F922    JSR     &F991   ;print a space
F925    LDX     #&13    ;X=&13 point to Execution address

************** print 4 bytes from CFS block header **********************

F927    LDY     #&04    ;loop pointer
F929    LDA     &03B2,X ;block header
F92C    JSR     &F97A   ;print ASCII equivalent of hex byte
F92F    DEX             ;X=X-1
F930    DEY             ;Y=Y-1
F931    BNE     &F929   ;

F933    RTS             ;return
```

```
        ;
********** print prompt for SAVE on TAPE ******************************

F934    LDA     &0247   ;filing system flag 0=CFS 2=RFS
F937    BEQ     &F93C   ;if cassette F93C
F939    JMP     &E310   ;else 'Bad Command error message'
F93C    JSR     &FB8E   ;switch Motor On
F93F    JSR     &FBE2   ;set up CFS for write operation
F942    JSR     &E7DC   ;check if free to print message
F945    BEQ     &F933   ;if not exit else
F947    JSR     &FA46   ; print message following call

F94A    DB      'RECORD then RETURN';
F95C    BRK             ;

F95D    JSR     &F995   ;confirm CFS not operating, nor ESCAPE flag set


************ wait for RETURN key to be pressed ************************

F960    JSR     OSRDCH  ;wait for keypress
F963    CMP     #&0D    ;is it &0D (RETURN)
F965    BNE     &F95D   ;no then do it again

F967    JMP     OSNEWL  ;output Carriage RETURN and LINE FEED


************* increment current load address ***************************

F96A    INC     &B1     ;current load address
F96C    BNE     &F974   ;
F96E    INC     &B2     ;current load address high word
F970    BNE     &F974   ;
F972    INC     &B3     ;current load address high word
F974    RTS             ;return
        ;
************* print a space + ASCII equivalent of hex byte **************

F975    PHA             ;save A on stack
F976    JSR     &F991   ;print a space
F979    PLA             ;get back A


************** print ASCII equivalent of hex byte  **********************

F97A    PHA             ;save A on stack
F97B    LSR             ;/16 to put high nybble in lo
F97C    LSR             ;
F97D    LSR             ;
F97E    LSR             ;
F97F    JSR     &F983   ;print its ASCII equivalent
F982    PLA             ;get back A

F983    CLC             ;clear carry flag
F984    AND     #&0F    ;clear high nybble
F986    ADC     #&30    ;Add &30 to convert 0-9 to ASCII A-F to : ; < = > ?
F988    CMP     #&3A    ;if A< ASC(':')
F98A    BCC     &F98E   ;goto F98E
F98C    ADC     #&06    ;else add 7 to convert : ; < = > ? to A B C D E F

F98E    JMP     OSWRCH  ;print character and return
```

```
******************** print a space  ************************************

F991    LDA     #&20    ;A=' '
F993    BNE     &F98E   ;goto F98E to print it


******************** confirm CFS not operating, nor ESCAPE flag set *****

F995    PHP             ;save flags on stack
F996    BIT     &EB     ;CFS Active flag
F998    BMI     &F99E   ;
F99A    BIT     &FF     ;if ESCAPE condition
F99C    BMI     &F9A0   ;goto F9A0
F99E    PLP             ;get back flags
F99F    RTS             ;return
        ;


F9A0    JSR     &F33B   ;close input file
F9A3    JSR     &FAF2   ;enable second processor and reset serial system
F9A6    LDA     #&7E    ;A=&7E (126) Acknowledge ESCAPE
F9A8    JSR     OSBYTE  ;OSBYTE Call

F9AB    BRK             ;
F9AC    DB      &11     ;error 17
F9AD    DB      'Escape' ;
F9B3    BRK             ;
```