```
^R cp/m source

.TITLE CPM

.=$2900             ;STANDARD 16K START ADDRESS

.UNIT.=4            ;WE TELL USER WHAT THE DEFAULT UNIT IS, HERE.
.BDOS.=5            ;VECTOR ADDRESS FOR BDOS CALLS
.CFCB.=$5C          ;WE BUILD AN FCB FOR USER FROM CMD STRING, HERE

;ASCII CHARACTERS

CH.NUL=0            ;NULL FOR ASCIZ
CH.ATN=$03          ;CONTROL C FOR ATTENTION
CH.TAB=$09          ;HORIZ TAB CHAR
CH.LF=$0A           ;LINEFEED
CH.CR=$0D           ;CARRIAGE RETURN
CH.XOF=$13          ;STOP OUTPUT
CH.EOF=$1A          ;END OF FILE (^Z)
CH.CTL=$20          ;CONTROL CHARS ARE THOSE LESS THAN THIS
CH.SPC=$20          ;SPACE
CH.NUM=$23          ;NUMBERSIGN
CH.DOL=$24          ;DOLLARSIGN FOR STRING TERMINATOR
CH.CLN=$3A          ;COLON
CH.PRM=$3E          ;PROMPT CHARACTER
CH.QM=$3F           ;WILDCARD CHAR, QUESTIONMARK
CH.A=$41            ;CHARACTER CAPITAL A
CH.UPA=$5E          ;UPARROW FOR TYPING CTL CHARS
CH.LCA=^O141
CH.LCZ=^O172        ;LOWER CASE A,Z
CH.RUB=$7F          ;RUBOUT

CH.MSK=$7F          ;MASK FOR 7-BIT CHARACTER

;CORE ADDRESSES

DEFBFR=$0080            ;DEFAULT DMA ADDR BUFFER
.TPA.=$0100             ;TRANSIENT PROGRAM AREA
.CCP.=$2900             ;CONSOLE COMMAND PROCESSOR

;COMMANDS TO BDOS

BF.TYI=1            ;TYPE IN A CHAR
BF.TYO=2            ;TYPE OUT A CHAR
BF.RDR=3            ;GET CHAR FROM READER
BF.PCH=4            ;SEND CHAR TO PUNCH
BF.LPT=5            ;SEND CHAR TO LISTING DEVICE
BF.ADR=6            ;GET ADDR OF BDOS
BF.GST=7            ;GET I/O STATUS BYTE
BF.SST=8            ;SET STATUS BYTE
BF.SOU=9            ;STRING OUTPUT
BF.SIN=10           ;STRING INPUT
BF.KBS=11           ;KEYBOARD STATUS CHECK

;FOLLOWING ARE THE DISK ONES

BF.LHD=12           ;LIFT HEAD
BF.INI=13           ;INIT BDOS, SET DMA TO 80, LOG IN DSK A
BF.LOG=14           ;LOG IN DISK N
```

```
BF.CLS=16       ;CLOSE FILE
BF.FND=17       ;FIND FILE
BF.FNX=18       ;FIND NEXT FILE
BF.DEL=19       ;DELETE FILE
BF.RED=20       ;READ NEXT RECORD
BF.WRT=21       ;WRITE NEXT RECORD
BF.CRE=22       ;CREATE NEW FILE
BF.REN=23       ;RENAME FILE
BF.ILV=24       ;INTERROGATE LOGIN VECTOR
BF.ILD=25       ;INTERROGATE LOGGED DRIVE NUMBER
BF.DMA=26       ;SET DMA ADDRESS
BF.IAV=27       ;INTERROGATE ALLOCATION VECTOR FOR CURRENT DISK
^L
;LAYOUT OF AN FCB (FILE CONTROL BLOCK)

FCB.UN=$00      ;UNIT NUMBER, IF FORCED
FCB.NM=$01      ;NAME, 8 CHARS
FCB.EX=$09      ;EXTENSION, 3 CHARS
FCB.RC=$0F      ;RECORD COUNT
FCB.MP=$10      ;MAP AREA
FCB.NR=$20      ;NUMBER OF RECORD

NDISKS=4        ;MAX NUMBER OF DISKS ON SYSTEM

XCLUST=$100     ;CLUSTERS ON A DISK, ROUNDED UP
NALLOV=XCLUST/8 ;BYTES OF ALLOC VECTOR FOR A DISK. (8 BITS/BYTE)
NNAMCH=8        ;CHARACTERS IN A FILE NAME
NEXTCH=3        ;CHARACTERS IN A FILENAME EXTENSION
NSECCH=$80      ;CHARACTERS IN A SECTOR
^L

CCP:    JMP     CCP000          ;MAIN ENTRY TO CCP

        JMP     CCPREE          ;RESTART, CLEARING CMD BUFFER

;TEXT BUFFER USED FOR COMMAND TYPEIN.
; IT IS PRE-LOADED WITH A COPYRIGHT STATEMENT

NCMDBF=^D128            ;SIZE OF BUFFER


CMDBUF: .BYTE   NCMDBF-1        ;MAXIMUM COUNT OF CHARS IN BUFFER
CBUFCT: .BYTE   2               ;CURRENT COUNT IN BUFFER
CBUFTX: .ASCII /                /       ;SIXTEEN SPACES
        .ASCII /COPYRIGHT (C) 1978, DIGITAL RESEARCH  /
        .BLKB   NCMDBF+CBUFTX-.

CMDPTR: .ADDR   CBUFTX          ;PARSER SCAN POINTER

ERRPTR: .BLKW 1                 ;POINTER TO TYPE OUT BAD PART IF ERROR

CCPTYO: MOV E,  A               ;CHAR TO TYPE
        MVI C,  BF.TYO          ;BDOS TYO FUNCTION
        JMP     .BDOS.          ;CALL BDOS

TYOSBC: PUSH    B               ;SAME, BUT SAVING BC PAIR
        CALL    CCPTYO          ;CALL TYO ROUTINE
        POP     B               ;RESTORE BC PAIR
        RET
```

```
TCRLF:  MVI A,  CH.CR           ;TYPE CARR RET, LINEFEED
        CALL    CCPTYO
        MVI A,  CH.LF           ;LINEFEED
        JMP     CCPTYO

CRSOUT: PUSH    B               ;SAVE POINTER TO ASCIZ
        CALL    TCRLF           ;FIRST A CRLF
        POP     H               ;RESTORE POINTER TO ASCIZ
CCPSOU: MOV A,  M               ;GET A CHAR FROM STRING
        ORA     A               ;IS IT A NULL?
        RZ                      ;IF SO, END OF STRING
        INX     H               ;NOT NULL, STEP PAST IT
        PUSH    H               ;SAVE FOR NEXT LOOP
        CALL    CCPTYO          ;TYPE THIS CHAR
        POP     H               ;POINT TO NEXT ONE
        JMP     CCPSOU          ;LOOP FOR MORE

INIDOS: MVI C,  BF.INI          ;FUNCTION = INIT BDOS
        JMP     .BDOS.

LOGDSK: MOV E,  A               ;THIS IS THE DESIRED DISK
        MVI C,  BF.LOG          ;LOG IN THIS DISK NUMBER
        JMP     .BDOS.

OPNFIL: MVI C,  BF.OPN          ;OPEN A FILE
        CALL    .BDOS.
        STA     FCBNUM          ;SAVE THE BYTE ADDR IN DIR
        INR     A               ;IF WAS $FF, SET Z FLAG
        RET                     ;RETURN Z FLAG TO CALLER

OPNCMD: XRA     A               ;START OF FILE
        STA     CMDFCB+FCB.NR
        LXI D,  CMDFCB          ;POINT TO FILE BLOCK
        JMP     OPNFIL          ;OPEN THE FILE

CLSFIL: MVI C,  BF.CLS          ;CLOSE FILE
        CALL    .BDOS.
        STA     FCBNUM                          ; 29D7  32 C8 30
        INR     A                               ; 29DA  3C
        RET                                     ; 29DB  C9

FNDFIL: MVI C,  BF.FND          ;FIND FILE
        CALL    .BDOS.
        STA     FCBNUM                          ; 29E1  32 C8 30
        INR     A                               ; 29E4  3C
        RET                                     ; 29E5  C9

FNDNXT: MVI C,  BF.FNX          ;FIND NEXT FILE IN WILDCARD GROUP
        CALL    .BDOS.
        STA     FCBNUM                          ; 29EB  32 C8 30
        INR     A               ;RETURN Z FLAG IF NO MORE.
        RET

FNDCFL: LXI D,  CMDFCB                          ; 29F0  11 A7 30
        JMP     FNDFIL          ;FIND FILE

DELFIL: MVI C,  BF.DEL          ;DELETE FILE
        JMP     .BDOS.

READFL: MVI C,  BF.RED          ;READ NEXT RECORD
```

```
        CALL    .BDOS.
        ORA     A
        RET

RECMD:  LXI D,  CMDFCB          ;FCB OF CURRENT COMMAND
        JMP     READFL          ;READ ANOTHER SECTOR

WRTFL:  MVI C,  BF.WRT          ;WRITE NEXT RECORD
        CALL    .BDOS.
        ORA     A
        RET

CREATF: MVI C,  BF.CRE          ;CREATE FILE
        CALL    .BDOS.
        STA     FCBNUM
        RET

RENMF:  MVI C,  BF.REN          ;RENAME FILE
        JMP     .BDOS.

UPPER:  CPI     CH.LCA          ;RANGE CHECK. LOWER CASE A?
        RC                      ;RETURN IF SMALLER
        CPI     CH.LCZ+1        ;LOWER CASE Z?
        RNC                     ;RETURN IF BIGGER
        ANI     $5F             ;MAKE UPPER CASE
        RET

; CODE TO GET A COMMAND, FROM TOP LEVEL COMMAND LOOP.

COLECT: LDA     SUBFLG          ;SUBMIT FILE IN PROGRESS?
        ORA     A
        JZ      COLEC2          ;JUMP IF NOT.
        LDA     CMDUNI          ;LOGGED DISK AT COMMAND LEVEL
        ORA     A
        MVI A,  0               ;SELECT "A"
        CNZ     LOGDSK          ;IF NOT DISK A, SWITCH TO A.
        LDA     FCBSUB+FCB.RC   ;RECORD COUNT
        DCR     A               ;LESS ONE
        STA     FCBSUB+FCB.NR
        LXI D,  FCBSUB          ;FCB OF SUBMIT NAME
        CALL    READFL          ;READ ANOTHER SECTOR
        JNZ     COLEC2          ;IF ERROR ON READ, STOP.
        LXI D,  CBUFCT          ;MOVE BUFFER TO HERE
        LXI H,  DEFBFR          ;MOVE BUFFER FROM HERE
        MVI B,  NSECCH          ;MOVE THIS MANY BYTES
        CALL    MOVNCH          ;MOVE STRING
                                ; 2A51  21 95 30
        LXI H,  FCBSUB+FCB.RC
                                ; 2A54  35
        DCR     M
        LXI D,  FCBSUB          ;FCB FOR SUBMIT NAME
        CALL    CLSFIL          ;CLOSE FILE
                                ; 2A5B  CA 77 2A
        JZ      COLEC2
        LDA     CMDUNI          ;COMMAND LEVEL DRIVE
        ORA     A               ;SEE IF DRIVE A
        CNZ     LOGDSK          ;IF NOT, SWITCH TO WHATEVER IT IS
        LXI H,  CBUFTX          ;POINT TO THE TEXT
        CALL    CCPSOU          ;SHOW THE SUB TEXT
        CALL    CHKKBS          ;ANYTHING TYPED BY OPERATOR?
        JZ      COLEC3          ;IF NOT, USE SUBMIT STUFF
        CALL    DELSUB          ;IF SO, DELETE SUB FILE.
        JMP     PROMPT          ;AND GO TO CMD PROMPT
```

```
COLEC2: CALL    DELSUB          ;DELETE ANY SUB FILE
        MVI C,  BF.SIN          ;STRING INPUT FROM TTY
        LXI D,  CMDBUF          ;HERE'S WHERE WE WANT TEXT
        CALL    .BDOS.          ;DO TEXTI
COLEC3: LXI H,  CBUFCT          ;SEE HOW MANY CHARS WE GOT
        MOV B,  M               ;HERE'S THE COUNT
COLEC4: INX     H               ;LOOP THRU TEXT, RAISING IT.
        MOV A,  B               ;COUNT LEFT
        ORA     A               ;ANY LEFT?
        JZ      COLEC5          ;QUIT WHEN ALL DONE
        MOV A,  M               ;GET A CHAR
        CALL    RAISE           ;IF LOWER CASE ALPHA, RAISE IT
        MOV M,  A               ;PUT UPPER CASE IN BUFFER
        DCR     B               ;COUNT THRU BUFFER
        JMP     COLEC4          ;LOOP

COLEC5: MOV M,  A               ;MAKE IT ASCIZ - NULL AFTER TEXT
        LXI H,  CBUFTX          ;POINT AT START OF TEXT
        SHLD    CMDPTR          ;SCAN STARTS HERE
        RET                     ;LINE READY FOR PROCESSING

CHKKBS: MVI C,  BF.KBS          ;CHECK KEYBOARD STATUS
        CALL    .BDOS.
        ORA     A
        RZ                      ;NOTHING TYPED
        MVI C,  BF.TYI          ;GET THE TYPED CHAR
        CALL    .BDOS.
        ORA     A               ;MAKE NON-ZERO FLAG (UNLESS NULL)
        RET

LIFTHD: MVI C,  BF.LHD          ;LIFT HEAD (NOP)
        JMP     .BDOS.

GETLDN: MVI C,  BF.ILD          ;GET LOGGED DISK NUMBER
        JMP     .BDOS.

;UNUSED ROUTINE TO MOVE (C) CHARS FROM (DE) TO (HL)

DOBLT:  LDAX    D
        MOV M,  A
        INX     D
        INX     H
        DCR     C
        JNZ     DOBLT
        RET

SETADR: MVI C,  BF.DMA          ;SET DMA ADDRESS
        JMP     .BDOS.                          ; 2AC0  C3 05 00

DELSUB: LXI H,  SUBFLG          ;WAS SUB FILE BEING DONE?
        MOV A,  M
        ORA     A
        RZ                      ;RETURN IF NOT
        MVI M,  0               ;IT WAS, BUT NO MORE.
        XRA     A               ;CHOOSE DRIVE A
        CALL    LOGDSK          ;LOG IT IN
        LXI D,  FCBSUB          ;SUBMIT FILE'S FCB
```

```
        JMP     LOGDSK          ;AND RETURN

CHKSER: LXI D,  CCPSER          ;MAKE SURE SERIALS MATCH
        LXI H,  DOSSER          ; BETWEEN CCP AND BDOS
        MVI B,  SERSIZ          ;CHECK THIS MANY CHARS
CHKSRL: LDAX    D
        CMP     M
        JNZ     SERERR          ;IF NOT, GO STEP ON SELF
        INX     D
        INX     H
        DCR     B
        JNZ     CHKSRL          ;LOOP THRU ALL OF SERIAL
        RET                     ;IF MATCHES, RETURN OK

;REJECT COMMAND

CMDREJ: CALL    TCRLF           ;NEW LINE
        LHLD    ERRPTR          ;TYPE OUT THE BAD GUY
CMDRJL: MOV A,  M               ; ..
        CPI     CH.SPC
        JZ      CMDRJ1          ;QUIT ON SPACE
        ORA     A
        JZ      CMDRJ1          ;QUIT AT END OF TEXT
        PUSH    H
        CALL    CCPTYO          ;TYPE CHAR OF BAD CMD
        POP     H
        INX     H
        JMP     CMDRJL          ;LOOP TIL NULL

CMDRJ1: MVI A,  '?              ;STICK "?" ON END
        CALL    CCPTYO
        CALL    TCRLF
        CALL    DELSUB          ;DELETE SUB FILE, IF ANY
        JMP     PROMPT          ;RESTART CCP
^L
;CHECK FOR TERMINATORS
; RETURN "Z" IF LEGAL SEPARATOR, ABORT ON CONTROLS.

CHKTRM: LDAX    D
        ORA     A
        RZ                      ;RETURN Z IF NULL
        CPI     CH.CTL          ;IF CONTROLS,
        JC      CMDREJ          ; ABORT
        RZ                      ;SPACE IS OK, THOUGH
        CPI     '=              ;SO ARE THE FOLLOWING
        RZ
CPI     '_
        RZ
CPI     '.
        RZ
CPI     CH.CLN          ;":"
        RZ
CPI     ';
        RZ
CPI     '[
        RZ
CPI     ']
        RZ
RET                     ;IF NONE OF THOSE, RETURN NON-ZERO
```

```
NOTSPC: LDAX    D               ;RETURN "Z" ON NULL
        ORA     A
        RZ
CPI     $20             ;SCAN FOR NON-SPACE
        RNZ
INX     D
        JMP     NOTSPC

HL=+A:  ADD     L               ;0A + HL => HL
        MOV L,  A
        RNC
INR     H
        RET
^L
;ROUTINE TO SCAN A WORD OR FILENAME

PARSE:  MVI A,  0               ;START AT BEGINNING TO BUILD WORD
PARSE1: LXI H,  CMDFCB          ;MAY BE A FILENAME, OR A VERB, ETC
        CALL    HL=+A           ;ADD A TO HL
        PUSH    H               ;SAVE TWO COPIES
        PUSH    H
        XRA     A               ;ASSUME NOT A DISK SPEC
        STA     PRSUNI
        LHLD    CMDPTR          ;COMMAND POINTER
        XCHG                    ;TO DE
        CALL    NOTSPC          ;SCAN FOR NON-SPACE
        XCHG                    ;BACK TO HL
        SHLD    ERRPTR          ;IF ERROR, WILL RETYPE FROM HERE
        XCHG                                            ; 2B5D  EB
        POP     H                                       ; 2B5E  E1
        LDAX    D               ;GET BACK THE NON-SPACE
        ORA     A               ;WAS NON-SPACE A NULL?
        JZ      PARSE2          ;JUMP IF SO
        SBI     CH=A-1
        MOV B,  A               ;MAY BE A DRIVE NAME
        INX     D               ;SEE IF A COLON IS NEXT
        LDAX    D
        CPI     CH=CLN          ;COLON?
        JZ      PARSE3          ;IF SO, THIS WAS A DRIVE
        DCX     D               ;WASN'T. BACK BEFORE COLON
PARSE2: LDA     CMDUNI          ;PLUG DEFAULT IN
        MOV M,  A
        JMP     PARSE4

PARSE3: MOV A,  B               ;WE HAD A <UNIT>:
        STA     PRSUNI          ;SO PLUG IT IN
        MOV M,  B               ; ..
        INX     D               ;STEP PAST UNIT AND COLON
PARSE4: MVI B,  NNAMCH
PARSE5: CALL    CHKTRM          ;LOOK FOR SPECIFIC TERMINATORS
        JZ      PARSE9          ;END OF WORD IF SO
        INX     H               ;NOT ONE OF THOSE. STEP PAST IT

        CPI     '*              ;WILDCARD?
        JNZ     PARSE6          ;JUMP IF NOT
        MVI M,  '?              ;YES, MAKE A WILD CHAR
        JMP     PARSE7          ;AND DON'T STEP PAST IT.

PARSE6: MOV M,  A               ;STORE CHAR FROM TEXT INPUT
```

```
PARSE7: DCR     B               ;COUNT CHARS IN A NAME
        JNZ     PARSE5          ;SOME LEFT, LOOP.
PARSE8: CALL    CHKTRM          ;SKIP TRAILING NON-TERMS
        JZ      PARS10          ;QUIT AT TERM
        INX     D               ;NON-TERM. SKIP IT.
        JMP     PARSE8

PARSE9: INX     H               ;FILL REST OF NAME WITH BLANKS
        MVI M,  CH.SPC
        DCR     B               ;COUNT THRU EIGHT CHARS
        JNZ     PARSE9          ;TILL ALL FILLED
PARS10: MVI B,  NEXTCH          ;CHARS IN AN EXTENSION
        CPI     '.              ;WAS TERM A DOT?
        JNZ     PARS15          ;JUMP IF NOT
        INX     D               ;YES. PASS IT.
PARS11: CALL    CHKTRM          ;CHECK CHARS IN EXTENSION
        JZ      PARS15          ;GO IF TERMINATOR
        INX     H               ;STEP DESTINATION
        CPI     '*              ;WILD CARD EXTENSION?
        JNZ     PARS12          ;JUMP IF NOT
        MVI M,  '?              ;YES, MAKE WILD CHAR
        JMP     PARS13          ;AND DON'T SCAN PAST IT

PARS12: MOV M,  A               ;COPY CHAR FROM SOURCE
        INX     D               ;STEP THRU EXTENSION
PARS13: DCR     B               ;COUNT EXTENSION
        JNZ     PARS11          ;LOOP TILL DONE
PARS14: CALL    CHKTRM          ;SKIP TRAILING NON-TERMS
        JZ      PARS16          ;JUMP ON TERM
        INX     D               ;STEP SOURCE
        JMP     PARS14          ;LOOK FOR TERM

PARS15: INX     H               ;STEP DESTINATION
        MVI M,  CH.SPC          ;PAD WITH SPACES
        DCR     B               ;COUNT THRU EXTENSION
        JNZ     PARS15          ;FILL WHOLE EXT
PARS16: MVI B,  $03             ;FILL MORE OF SPARE BYTES OF FCB
PARS17: INX     H               ;HERE
        MVI M,  0               ;WITH ZEROS
        DCR     B               ;COUNT THEM
        JNZ     PARS17          ;LOOP FOR THREE SPARE BYTES
        XCHG                    ;GET HOW MUCH READ FROM TEXT
        SHLD    CMDPTR          ;REMEMBER FOR NEXT SCAN
        POP     H               ;BACK TO START OF FILENAME
        LXI B,  NNAMCH+NEXTCH   ;C GETS # TO SCAN, B GETS ANSWER
PARS18: INX     H               ;LOOK FOR WILD CARDS IN FILENAME
        MOV A,  M               ;GET A FILENAME CHAR
        CPI     CH.QM           ;WILD?
        JNZ     PARS19          ;IF NOT, DON'T COUNT IT
        INR     B               ;YES, COUNT IT.
PARS19: DCR     C               ;SCAN NAME AND EXT
        JNZ     PARS18
        MOV A,  B               ;NUMBER OF WILD CHARS
        ORA     A               ;RETURN Z FLAG AS FAST CHECK
        RET

;CCP COMMAND TABLE

CCPCTB: .ASCII /DIR /
NCMPCH=.-CCPCTB                 ;CHARS IN A COMMAND NAME
```

```
        .ASCII /ERA /
        .ASCII /TYPE/
        .ASCII /SAVE/
        .ASCII /REN /
△ NCCPC=<.-CCPCTB>/NCCPCH          ;COMMANDS IN TABLE

△ CCPSER: .BYTE 0,$0E,0,0,1,2      ;SERIAL NUMBER OF COPY OF CCP
△ SERSIZ=.-CCPSER                  ;SIZE OF SERIAL NUMBER

;HERE TO LOOK UP A COMMAND VERB

FINDCM: LXI H, CCPCTB              ;POINT TO LEGAL COMMANDS
        MVI C, 0                   ;THIS WILL BE INDEX INTO TABLE
FINDC1: MOV A, C
        CPI  NCCPC                 ;TRIED THEM ALL?
        RNC                        ;GIVE UP IF SO
        LXI D, CMDNAM              ;LOOK AT USER'S STRING
        MVI B, NCCPCH              ;COMMAND IS THIS LONG
FINDC2: LDAX D
        CMP  M                     ;CHECK A CHARACTER
        JNZ  FINDC3                ;NOT THIS COMMAND
        INX  D                     ;STEP THRU REAL AND TEST STRINGS
        INX  H
        DCR  B                     ;COUNT LENGTH OF COMMAND
        JNZ  FINDC2                ;LOOP FOR WHOLE COMMAND
        LDAX D                     ;CHECK USER'S CHAR AFTER COMMAND
        CPI  CH.SPC                ;FOR SPACE
        JNZ  FINDC4               9;JUMP IF NOT SPACE
        MOV A, C                   ;RETURN TABLE INDEX
        RET

FINDC3: INX  H                     ;NOT THIS COMMAND. SKIP REST OF IT.
        DCR  B
        JNZ  FINDC3
FINDC4: INR  C                     ;TRY NEXT COMMAND
        JMP  FINDC1

                                                    ; 2C3A  B7
CPNSUB: ORA  A                                      ; 2C3B  C2 4C 2C
        JNZ  CPNSB1
        LXI D, FCBSUB              ;SUBMIT COMMAND'S FILENAME
        CALL CPNFIL                ;TRY TO FIND IT
        JZ   CPNSB1                ;IF NOT THERE, SKIP.
        MVI A, $FF                 ;FLAG DOING SUB FILE
        JMP  CPNSB2

CPNSB1: XRA  A                     ;NOT DOING SUB FILE
CPNSB2: STA  SUBFLG                ;STORE WHETHER SUBMIT IN PROGRESS OR NOT
        RET

                                                    ; 2C51  AF
CCPREE: XRA  A                                      ; 2C52  32 07 29
        STA  CBUFCT

;JUMP HERE FROM BASE OF CCP

CCP000: LXI SP, CCPSTK             ;SET UP STACK FOR CCP
                                                    ; 2C58  C5
        PUSH B                                      ; 2C59  C5
        PUSH B
        CALL INIDOS                ;INIT BDOS
                                                    ; 2C5D  C1
        POP  B                                      ; 2C5E  F5
        PUSH PSW
```

```
                MOV A, C                ;THIS DISK NUMBER
                CALL    LOGDSK          ;LOG IT IN AND SELECT IT
                POP     PSW                                     ; 2C63  F1
                POP     B                                       ; 2C64  C1
                INR     A                                       ; 2C65  3C
                ORA     C                                       ; 2C66  B1
                CALL    OPNSUB          ;OPEN SUB FILE, IF EXISTS
                                                                ; 2C6A  3A 07 29
                LDA     CBUFCT                                  ; 2C6D  B7
                ORA     A                                       ; 2C6E  C2 87 2C
                JNZ     NPROMT
PROMPT: LXI SP, CCPSTK          ;INIT STACK                     ; 2C74  CD 98 29
                CALL    TCRLF
                CALL    GETLDN          ;INTERROGATE DRIVE NUMBER
                ADI     CH.A            ;MAKE IT A LETTER, A-D
                CALL    CCPTYO
                MVI A, CH.PRM           ;PROMPT CHAR
                CALL    CCPTYO
                CALL    COLECT          ;GO COLLECT A COMMAND
NPROMT: LXI D, DEFBFR           ;DEFAULT LOW CORE BUFFER
                CALL    SETADR          ;SET DMA ADDRESS
                CALL    GETLDN          ;INTERROGATE DRIVE NUMBER
                STA     CMDUNI          ;SAVE COMMAND LEVEL DISK
                CALL    PARSE                                   ; 2C93  CD 44 2B
                CNZ     CMDREJ                                  ; 2C96  C4 EF 2A
                LDA     PRSUNI                                  ; 2C99  3A CA 32
                ORA     A                                       ; 2C9C  B7
                JNZ     ZNOTFD          ;NO SUCH COMMAND
                CALL    FINDCM                                  ; 2CA0  CD 10 2C
                LXI H, CDSPTB                                   ; 2CA3  21 B0 2C
                MOV E, A                                        ; 2CA6  5F
                MVI D, 0                                        ; 2CA7  16 00
                DAD     D                                       ; 2CA9  19
                DAD     D                                       ; 2CAA  19
                MOV A, M                                        ; 2CAB  7E
                INX     H                                       ; 2CAC  23
                MOV H, M                                        ; 2CAD  66
                MOV L, A                                        ; 2CAE  6F
                PCHL                                            ; 2CAF  E9

;DISPATCH TABLE FOR TYPED COMMANDS

CDSPTB: .ADDR   ZDIR            ;DIRECTORY COMMAND
        .ADDR   ZERA            ;ERASE COMMAND
        .ADDR   ZTYP            ;TYPE COMMAND
        .ADDR   ZSAV            ;SAVE COMMAND
        .ADDR   ZREN            ;RENAME COMMAND
        .ADDR   ZNOTFD          ;NO SUCH COMMAND

;HERE ONLY IF SERIAL NUMBER MISMATCHES

SERERR: LXI H, $76F3            ;MAKE A DI, HLT PAIR
        SHLD    CCP             ;PUT THEM AT START ADDRESS OF CCP
        LXI H, CCP              ;GO DO THEM
        PCHL                    ;WHEN SERIAL NUMBERS DONT MATCH

RRDERR: LXI B, MRDERR           ;TYPE READ ERROR MESSAGE
        JMP     CRSOUT

MRDERR: .ASCIZ /READ ERROR/
```

```
RNTFND: LXI B,  MNTFND          ;TYPE NOT FOUND MESSAGE
        JMP     CRSOUT

MNTFND: .ASCIZ /NOT FOUND/

GETNUM: LXI H,  CMDNAM          ;POINT AT WORD WHICH MAY BE NUMBER
        LXI B,  $000B           ;11 CHARS, ANS IN B STARTS AT 0
GETNML: MOV A,  M               ;GET WHAT MAY BE A DIGIT
        CPI     CH.SPC          ;END OF WORD?
        JZ      GETNM1          ;JUMP IF SO, CHECK TRAILING CHARS
        INX     H               ;NO. STEP PAST IT
        SUI     '0              ;SEE IF IT'S A DECIMAL DIGIT
        CPI     10              ;0-9?
        JNC     CMDREJ          ;ERROR IF NOT
        MOV D,  A               ;GOOD. KEEP A COPY
        MOV A,  B               ;ACCUMULATE DECIMAL NUMBER
        ANI     $E0             ;PRE-CHECK FOR OVERFLOW
        JNZ     CMDREJ          ;BAD IF SO
        MOV A,  B               ;OK, MULTIPLY PARTIAL BY 10
        RLC                     ;THREE SHIFTS FOR EIGHT
        RLC
RLC
ADD     B                       ;ADD ONE MAKES NINE
        JC      CMDREJ
        ADD     B               ;ONE MORE MAKES TEN
        JC      CMDREJ          ;QUIT IF OVERFLOW
        ADD     D               ;ADD IN NEW DIGIT
        JC      CMDREJ          ;AGAIN, ERR IF OVERFLOW
        MOV B,  A               ;NEW PARTIAL ANSWER
        DCR     C               ;COUNT CHARS IN SOURCE LINE
        JNZ     GETNML          ;LOOP IF MORE TO GO
        RET

;MAKE SURE END OF WORD AND NO MORE BUT SPACE FILLS

GETNM1: MOV A,  M               ;ALL SPACES?
        CPI     CH.SPC
        JNZ     CMDREJ          ;NOT A SPACE
        INX     H
        DCR     C               ;COUNT AND SKIP ANOTHER
        JNZ     GETNM1          ;LOOP IF MORE
        MOV A,  B
        RET
^L
MOV3CH: MVI B,  $03             ;MOVE 3 CHARS
MOVNCH: MOV A,  M               ;ENTER HERE TO MOVE (B) CHARS
        STAX    D               ;FROM (HL) TO (DE)
        INX     H
        INX     D
        DCR     B
        JNZ     MOVNCH
        RET

;GET A CHAR FROM DISK DIRECTORY AT (A) + (C) INTO THE BUFFER

GTDRCH: LXI H,  DEFBFR
        ADD     C               ;CURRENT CHAR + FCB BASE
        CALL    HL..A           ;0A + HL => HL
        MOV A,  M               ;GET THE CHAR
        RET
```

```
;ROUTINES TO SWITCH TO/FROM DISK NAMED IN A COMMAND, FROM TOP-LEVEL ONE.

TMPSEL: XRA     A               ;CLEAR DISK NUMBER BYTE IN FCB
        STA     CMDFCB
TMPSL2: LDA     PRSUNI          ;DISK SPECIFIED?
        ORA     A
        RZ                      ;DONE IF NOT
        DCR     A               ;YES, CONVERT TO 0-3
        LXI H,  CMDUNI          ;WAS IT THE DEFAULT?
        CMP     M               ;SAME AS TOP LEVEL?
        RZ                      ;IF SO, DON'T RE-LOG IT
        JMP     LOGDSK          ;DIFFERENT. GO LOG IT.

;ROUTINE TO SWITCH TO COMMAND LEVEL DISK.

CMDSEL: LDA     PRSUNI          ;WAS THERE ONE IN COMMAND?
        ORA     A               ; ..
        RZ                      ;RETURN IF NOT
        DCR     A               ;YES. CONVERT TO 0-3 FORM
        LXI H,  CMDUNI                                  ; 2D51  21 C9 30
        CMP     M               ;IS IT SAME AS TOP LEVEL ONE?
        RZ                      ;IF SO, NO PROBLEM
        LDA     CMDUNI          ;IF DIFFERENT, SWITCH BACK TO THIS
        JMP     LOGDSK          ;GO LOG IT

;DIRECTORY LISTING COMMAND. DIR, OR DIR AFN

DIR:    CALL    PARSE           ;GET THE TEXT ARGUMENT
        CALL    TMPSEL          ;SWITCH TO REQUESTED DISK
        LXI H,  CMDNAM          ;POINT TO ARGUMENT
        MOV A,  M               ;WAS IT BLANK?
        CPI     CH.SPC
        JNZ     ZDIR2           ;NO. DIR OF SPECIFIC FILES
        MVI B,  NNAMCH+NEXTCH   ;YES. PLUG IN ALL WILD CARDS
ZDIR1:  MVI M,  CH.QM           ;FILL FCB NAME AND EXT
        INX     H
        DCR     B               ;COUNT THEM
        JNZ     ZDIR1           ;LOOP FOR WHOLE NAME/EXT
ZDIR2:  CALL    FNDCFL          ;FIND FILE
        CZ      RNTFND          ;IF NONE AT ALL, TYPE "NOT FOUND"
ZDIR3:  JZ      ZDIR7           ;JUMP IF NONE NOUND
        CALL    TCRLF           ;NEXT LINE
        CALL    GETLDN          ;FIND LOGGED DISK NUMBER
        ADI     CH.A            ;MAKE "A"-"D"
        CALL    CCPTYO          ;DISK NAME
        MVI A,  CH.CLN          ;COLON
        CALL    CCPTYO
        MVI A,  CH.SPC          ;SPACE FOR SEPARATOR
        CALL    CCPTYO
        LDA     FCBNUM          ;COMPUTE BYTE ADDR IN SECTOR
        RAL                     ; FOR THIS FCB, FROM FCB NUMBER
        RAL
        RAL
        ANI     $60             ;32 BYTES PER FCB, FOUR FCB PER SECTOR
        MOV C,  A               ;HOLD HERE
        MVI B,  $01             ;FIRST CHAR OF FILENAME
ZDIR4:  MOV A,  B               ;CHARACTER NUMBER
        CALL    GTDRCH          ;GET DEFBFR+C+A TO A
        CPI     CH.SPC          ;TRAILING SPACE IN NAME?
```

```
        JNZ     ZDIR5           ;JUMP IF NOT
        MVI A,  $09             ;IF SO, SKIP TO EXTENSION
        CALL    GTDRCH          ;GET EXTENSION'S FIRST CHAR
        MVI A,  CH.SPC          ;IS EXT BLANK?
        CMP     M               ; ..
        JZ      ZDIR6           ;JUMP IF SO.
ZDIR5:  CALL    TYOSBC          ;TYO, SAVING BC
        INR     B               ;STEP TO NEXT CHAR OF NAME/EXT
        MOV A,  B
        CPI     $0C             ;END OF EXTENSION YET?
        JNC     ZDIR6           ;JUMP IF SO
        CPI     $09             ;END OF NAME?
        JNZ     ZDIR4           ;JUMP IF NOT
        MVI A,  CH.SPC          ;YES, SPACE BETWEEN NAME AND EXT
        CALL    TYOSBC          ;TYPE SPACE
        JMP     ZDIR4           ;START ON EXT

ZDIR6:  CALL    CHKKBS          ;CHECK FOR TYPEIN
        JNZ     ZDIR7           ;IF SO, BREAK OUT
        CALL    FNDNXT          ;FIND NEXT FILE IN * GROUP
        JMP     ZDIR3           ;GO LIST NEXT FILENAME

ZDIR7:  JMP     CMDDUN          ;ON TYPEIN, DONE WITH COMMAND

;ERASE FILE(S) COMMAND

ZERA:   CALL    PARSE           ;SCAN NAME(S)
        CPI     NNAMCH+NEXTCH   ;ALL FILES?
        JNZ     ZERA1           ;NOT ALL WILDCARDS
        LXI B,  ERAMSG          ;YES, BETTER CHECK THAT!
        CALL    CRSOUT          ;ASK OPERATOR
        CALL    COLECT          ;GET A Y OR N
        LXI H,  CBUFCT          ;HOW MANY CHARS IN ANSWER
        DCR     M               ;BETTER HAVE BEEN ONE
        JNZ     PROMPT          ;JUMP IF WASN'T ONE.
        INX     H               ;WAS ONE CHAR. GET IT.
        MOV A,  M               ; ..
        CPI     'Y              ;WAS IT A YES?
        JNZ     PROMPT          ;QUIT IF NOT.
        INX     H               ;REALLY WANTS ALL FILES DELETED
        SHLD    CMDPTR          ;POINT AT THE TRAILING SPACES
        LXI H,  CMDFCB          ;AND RE-BUILD THE "*.*"
        MVI M,  CH.QM           ;OUT OF ALL WILD-CARDS
        CALL    TMPSL2          ;SWITCH TO CORRECT DISK
        LXI D,  CMDFCB          ;POINT TO THE WILD NAME
        CALL    DELFIL          ;DO THE DELETES
        CALL    INIDOS          ;RE-INIT THE DOS
        JMP     CMDDUN          ;SCAN TO EOL AND RE-PROMPT

ZERA1:  CALL    TMPSEL          ;SWITCH TO REQUESTED DISK
        LXI D,  CMDFCB          ;POINT TO REQUESTED FILE NAME
        CALL    DELFIL          ;DO THE DELETES
        JMP     CMDDUN          ;SCAN TO EOL AND RE-PROMPT

ERAMSG: .ASCIZ  .ALL FILES (Y/N)?.      ;FOR VERIFYING *.*

;COMMAND TO TYPE OUT A FILE

ZTYP:   CALL    PARSE           ;GET THE FILE NAME TO TYPE
        JNZ     CMDBAD          ;CAN'T DO WILDCARDS
```

```
        CALL    TMPSEL          ;SWITCH TO REQUESTED DISK
        CALL    OPNCMD          ;OPEN THE FILE
        JZ      ZTYPF           ;JUMP IF NOT FOUND
        CALL    TCRLF           ;MOVE BELOW COMMAND LINE.
        LXI H,  TYPNCH          ;CHAR COUNT IN SECTOR (USED CT)
        MVI M,  $FF             ;MAKE HUGE
ZTYP1:  LXI H,  TYPNCH          ;HOW MANY CHARS USED
        MOV A,  M               ;READ THIS MANY?
        CPI     NSECCH          ;WHOLE SECTOR?
        JC      ZTYP2           ;JUMP IF STILL OK
        PUSH    H               ;NEED TO READ A SECTOR. SAVE HL
        CALL    REDCMD          ;READ NEXT SECTOR
        POP     H               ;RESTORE STACK / HL
        JNZ     TYEOFQ          ;JUMP IF EOF OR ERR.
        XRA     A               ;CLEAR COUNT OF TYPED CHARS IN SECTOR
        MOV M,  A               ; ..
ZTYP2:  INR     M               ;COUNT A USED CHAR
        LXI H,  DEFBFR          ;POINT INTO BUFFER
        CALL    HL..A           ;ADD IN USED COUNT
        MOV A,  M               ;GET THE CHAR AT THAT POINT
        CPI     CH.EOF          ;END OF FILE?
        JZ      CMDDUN
        CALL    CCPTYC          ;NO, TYPE THE CHAR.
        CALL    CHKKBS          ;CHECK THE KEYBOARD
        JNZ     CMDDUN          ;IF TYPEIN, QUIT.
        JMP     ZTYP1           ;LOOP, TYPE MORE.

TYEOFQ: DCR     A               ;WAS IT ERR OR EOF?
        JZ      CMDDUN          ;IF EOF, THAT'S ALL.
        CALL    RRDERR          ;ERROR. SAY "READ ERROR"
ZTYPF:  CALL    DFLSEL          ;SWITCH BACK TO DEFAULT UNIT
        JMP     CMDREJ          ;AND ERROR PATH BACK TO TOP

;
;SAVE COMMAND

ZSAV:   CALL    PARSE           ;SCAN A SIZE NUMBER
        LDA     PRSUNI          ;DISK SPECIFIED IN SIZE??
        ORA     A               ; ..
        JNZ     CMDREJ          ;IF SO, THAT'S WRONG.
        CALL    GETNUM          ;GET SIZE FROM COMMAND TEXT
        PUSH    PSW             ;SAVE THE COUNT
        CALL    PARSE           ;NOW GET THE FILENAME
        JNZ     CMDREJ          ;NO WILDCARDS ALLOWED
        CALL    TMPSEL          ;SWITCH TO REQUESTED DISK
        LXI D,  CMDFCB          ;DELETE ANY PREVIOUS COPY
        PUSH    D               ;SAVE ADDR OF CMDFCB
        CALL    DELFIL          ;DELETE
        POP     D               ;POINT AT NAME AGAIN
        CALL    CREATF          ;CREATE NEW COPY
        CALL    OPNCMD          ;OPEN NEW COPY
        JZ      ZSAV2           ;JUMP IF NO SPACE.
        POP     PSW             ;NUMBER OF PAGES TO SAVE
        MOV L,  A               ;CONVERT TO SECTORS
        MVI H,  0               ;WHICH MAY BE 9 BITS
        DAD     H               ;BY DOUBLING (NOT PARAMETERIZABLE?)
        LXI D,  .TPA.           ;START AT BASE OF TRANSIENT AREA
ZSAV1:  MOV A,  H               ;SEE IF DONE ALL SECTORS
        ORA     L               ; ..
        JZ      ZSAV3           ;IF ZERO, DONE.
        DCX     H               ;COUNT ANOTHER SECTOR SAVED
```

```
        PUSH    H               ;SAVE COUNT
        LXI H,  NSECCH          ;ADD ONE SECTOR
        DAD     D               ;TO CURRENT PLACE IN CORE
        PUSH    H               ;SAVE WHERE TO DUMP FROM
        CALL    SETADR          ;SET DMA ADDRESS TO THERE
        LXI D,  CMDFCB          ;FILE WE ARE WRITING
        CALL    WRITFL          ;WRITE ANOTHER RECORD
        POP     D               ;RESTORE CURRENT ADDRESS
        POP     H               ;AND COUNT OF SECTORS LEFT TO DO
        JNZ     ZSAV2           ;JUMP IF ERROR ON WRITE
        JMP     ZSAV1           ;ELSE GO SAVE SOME MORE

ZSAV2:  LXI B,  ZSAVM1          ;NO SPACE ERROR MESSAGE
        CALL    CRSOUT          ;TYPE IT
ZSAV3:  LXI D,  CMDFCB          ;POINT AT FILE
        CALL    CLSFIL          ;CLOSE IT, DO WRITE ALLOC BYTES
        JNZ     ZSAV4           ;JUMP IF OK
        LXI B,  ZSAVM2          ;ERROR ON CLOSE.
        CALL    CRSOUT          ;SAY SO
ZSAV4:  JMP     CMDDUN          ;END OF SAVE COMMAND

ZSAVM1: .ASCIZ /NO SPACE/

ZSAVM2: .ASCIZ /CANNOT CLOSE/
^L
;RENAME COMMAND - REN NEW=OLD

ZREN:   CALL    PARSE           ;GET NEW FILE NAME
        JNZ     CMDREJ          ;ERROR IF WILDCARDS
        LDA     PRSUNI          ;ANY DISK?
        PUSH    PSW             ;SAVE FOR LATER CHECK
        CALL    TMPSEL          ;SWITCH TO REQUESTED DISK
        CALL    FNDCFL          ;FIND NEW FILE
        JNZ     ZREN5           ;JUMP IF FOUND (SHOULDN'T)
        LXI H,  CMDFCB          ;COPY OUT THE FILE NAME
        LXI D,  CMFCB2          ;TO HERE
        MVI B,  $10             ;THIS MANY CHARACTERS
        CALL    MOVNCH          ;MOVE THAT TEXT
        LHLD    CMDPTR          ;RESTART SCAN
        XCHG
CALL    NOTSPC                  ;SCAN FOR A NON-SPACE
        CPI     '=              ;LEGAL SEPARATORS
        JZ      ZREN1           ;OK
        CPI     '_              ;OTHER LEGAL SEPARATOR
        JNZ     ZREN4           ;ERROR IF NOT
ZREN1:  XCHG
INX     H                       ;STEP PAST THE "="
        SHLD    CMDPTR          ;RE-SCAN FROM THIS POINT
        CALL    PARSE           ;GET OLD FILE NAME
        JNZ     ZREN4           ;NO STARS ALLOWED
        POP     PSW             ;NEW DISK NUMBER
        MOV B,  A               ;COMPARE WITH OLD FILE'S DISK
        LXI H,  PRSUNI          ;DISK IN SECOND NAME
        MOV A,  M               ;WAS THERE ONE?
        ORA     A               ; ..
        JZ      ZREN2           ;JUMP IF NOT
        CMP     B               ;YES. SAME AS OLD FILE?
        MOV M,  B               ;AND UPDATE IT
        JNZ     ZREN4           ;CAN'T RENAME ACROSS DISKS
ZREN2:  MOV M,  B               ;THIS IS THE ONE.
```

```
        XRA     A               ;CLEAR DISK BYTE IN NAME
        STA     CMDFCB          ; ..
        CALL    FNDCFL          ;FIND OLD FILE.
        JZ      ZREN3           ;IF NOT FOUND, QUIT
        LXI D,  CMDFCB
        CALL    RENAMF          ;DO THE RENAME!
        JMP     CMDDUN          ;DONE

ZREN2:  CALL    RNTFND          ;OLD FILE NOT FOUND MSG
        JMP     CMDDUN

ZREN3:  CALL    DFLSEL          ;SWITCH BACK TO DEFAULT UNIT
        JMP     CMDREJ

ZREN4:  LXI B,  FEXMSG          ;NEW FILE ALREADY EXISTS MSG
        CALL    CRSOUT
        JMP     CMDDUN          ;AND NO MORE TO DO

FEXMSG: .ASCIZ /FILE EXISTS/
;L
ZNOCMD: CALL    CHKSER          ;NO SUCH CMD. IS VERSION OF SYSTEM OK?
        LDA     CMDNAM          ;YES. CMD START WITH A SPACE?
        CPI     CH.SPC
        JNZ     RCOMQ           ;IF NOT BLANK COMMAND, JUMP
        LDA     PRSUNI          ;DISK DRIVE ALONE?
        ORA     A
        JZ      CMDDN1          ;IF NOT, PROBABLY BLANK LINE
        DCR     A               ;THERE WAS A DRIVE. CONVERT TO 0-3
        STA     CMDUNI          ;KEEP A COPY
        STA     .UNIT.          ;AND ONE FOR THE USER
        CALL    LOGDSK          ;GO LOG THAT DISK
        JMP     CMDDN1          ;CHECK FOR JUNK ON END OF LINE

RCOMQ:  LXI D,  CMDFCB+FCB.EX   ;FIRST CHAR AFTER 8-LTR NAME
        LDAX    D
        CPI     CH.SPC          ;SPACE?
        JNZ     CMDREJ          ;IF NOT, ERROR.
RCOM:   PUSH    D               ;SAVE POINTER TO EXTENSION
        CALL    TMPSEL          ;SWITCH TO REQUESTED DISK
        POP     D               ;EXTENSION AGAIN
        LXI H,  COMEXT          ;PLUG IN .COM EXTENSION
        CALL    MOV3CH          ;COPY IT INTO FCB
        CALL    OPNCMD          ;TRY TO OPEN SUCH A FILE
        JZ      RCOME           ;JUMP IF FAILED. RE-LOG CMD DISK
        LXI H,  .TPA.           ;BASE OF TRANSIENT AREA
RCOM1:  PUSH    H               ;SAVE CORE ADDR
        XCHG                    ;IN DE FOR SUBR ARG
        CALL    SETADR          ;SET DMA ADDRESS
        LXI D,  CMDFCB          ;THIS IS THE FILE
        CALL    READFL          ;READ ANOTHER SECTOR
        JNZ     RCOM2           ;IF EOF OR ERROR .
        POP     H               ;ADDRESS AGAIN
        LXI D,  NSECCH          ;SIZE OF SECTOR
        DAD     D               ;MOVE UP THAT MUCH
        LXI D,  CCP             ;BASE OF CCP
        MOV A,  L               ;MAKE SURE NOT OVERWRITING SELF
        SUB     E               ;16 BIT SUBTRACT
        MOV A,  H
        SBB     D
        JNC     RCOMX           ;JUMP IF TOO BIG
```

```
        JMP     RCOM1           ;ELSE READ SOME MORE

RCOM2:  POP     H               ;CLEAR ADDR FROM STACK
        DCR     A               ;BETTER BE EOF
        JNZ     RCOMX           ;IF NOT, ERROR
        CALL    DFLSEL          ;SWITCH BACK TO DEFAULT UNIT
        CALL    PARSE           ;SEE IF ANY TRAILING ARGS
        LXI H,  PRSUNI          ;GET DISK DRIVE NUMBER
        PUSH    H               ;SAVE A COPY OF ADDR
        MOV A,  M               ;PICK UP DRIVE
        STA     CMDFCB          ;PUT A COPY IN FCB OF FILE
        MVI A,  $10             ;MOVE DOWN IN THE FCB THIS FAR
        CALL    PARSE1          ;AND PARSE ANY ARGUMENTS
        POP     H               ;DRIVE ADDRESS AGAIN
        MOV A,  M               ;MAY HAVE BEEN A DRIVE IN THE ARG
        STA     CMFCB2
        XRA     A               ;START OF FILE
        STA     CMDFCB+FCS.NR
        LXI D,  .CFCB.          ;COMMAND FCB IN LOW CORE
        LXI H,  CMDFCB          ;POSSIBLE ARG OF FILENAME
        MVI B,  $21             ;THIS MANY CHARS FOR USER
        CALL    MOVNCH          ;COPY THEM
        LXI H,  CBUFTX          ;RE-SCAN TO GIVE USER TEXT
RCOM3:  MOV A,  M               ;LOOK FOR SPACE OR NULL
        ORA     A               ; ..
        JZ      RCOM4           ;QUIT ON NULL
        CPI     CH.SPC          ;QUIT ON SPACE
        JZ      RCOM4           ; ..
        INX     H               ;KEEP LOOKING
        JMP     RCOM3

RCOM4:  MVI B,  0               ;COUNT
        LXI D,  DEFBFR+1        ;COPY TRAILING STUFF FOR USER
RCOM5:  MOV A,  M               ;FROM COMMAND STRING
        STAX    D               ;TO USER COMM REGION
        ORA     A               ;QUIT ON NULL
        JZ      RCOM6           ; ..
        INR     B               ;COUNT
        INX     H               ;SOURCE
        INX     D               ;DEST
        JMP     RCOM5           ;AND LOOP

RCOM6:  MOV A,  B               ;HOW MANY CHARS WE COPIED
        STA     DEFBFR          ;TELL USER. (HE GETS NO NULL)
        CALL    TCRLF           ;BEFORE GOING TO USER, CRLF
        LDA     CMDUNI          ;TELL USER THE CURRENT DRIVE
        STA     .UNIT.          ; ..
        CALL    LIFTHD          ;LIFT HEAD (CCP) TO BDOS
        LXI D,  DEFBFR          ;PUT DMA BACK TO DEFAULT
        CALL    SETADR          ;SET DMA ADDRESS
        CALL    .TPA.           ;CALL THE USER PROGRAM
        LXI SP, CCPSTK          ;IN CASE IT STEPPED ON STACK
        LDA     CMDUNI          ;IN CASE IT CHANGED DISKS
        CALL    LOGDSK          ;RE-LOG THE DEFAULT
        JMP     PROMPT          ;AND GO GET ANOTHER COMMAND

RCOME:  CALL    DFLSEL          ;SWITCH BACK TO DEFAULT UNIT
        JMP     CMDREJ          ;ERROR RETURN

RCOMX:  LXI B,  LDERRM          ;LOAD ERROR COMMENT
```

```
        CALL    CRSOUT
        JMP     CMDDUN          ;ASK FOR ANOTHER COMMAND

LDERRM: .ASCIZ /LOAD ERROR/

COMEXT: .ASCII /COM/

CMDUN:  CALL    DFLSEL          ;SWITCH BACK TO DEFAULT UNIT
CMDUN1: CALL    PARSE           ;ANY MORE TEXT?
        LDA     CMDNAM          ;SHOULD BE JUST BLANKS
        SUI     CH.SPC          ; ..
        LXI H,  PRSUN1          ;MAYBE A DISK GOT PARSED
        ORA     M               ; ..
        JNZ     CMDREJ          ;IF EITHER, ERROR.
        JMP     PROMPT          ;OK, GET ANOTHER COMMAND

        .BLKB $10
CCPSTK:                         ;CCP'S STACK ABOVE HERE
SUBFLG: .BYTE 0                 ;FLAG FOR SUBMIT FILE BEING PROCESSED

FCBSUB: .BYTE 2                 ;FCB FOR SUBMIT
        .ASCII /$$$     SUB/
        .BLKB 21

CMDFCB: .BLKB 1                 ;FCB STARTS HERE W/ DRIVE NUMBER
CMDNAM: .BLKB 15                ;USER'S COMMAND VERB OR FILENAME GOES HERE
CMDF2:  .BLKB $11               ;ANOTHER ONE FOR RENAMES

FCBDM:  .BLKB 1
CMDUNI: .BLKB 1                 ;UNIT AT COMMAND LEVEL
PRSUNI: .BLKB 1                 ;UNIT RECOGNIZED BY PARSER
TYPCH:  .BLKB 1                 ;TEMP FOR TYPE COMMAND

;SPACE TO PAGE BOUNDARY
        .BLKB 52
;********** END OF CONSOLE COMMAND PROCESSOR **********

;*************** BDOS STARTS HERE ***********************

DOSSER: .BYTE 0,$0E,0,0,1,2     ;SERIAL NUMBER OF COPY OF DOS
SERSIZ=.-DOSSER                 ;SIZE OF SERIAL NUMBER

;ENTER HERE FOR ALL SERVICES. 5/ JMP BDOS

BDOS:   JMP     BDOSH           ;THIS IS A VECTOR. GO TO BODY.

VBADSC: .ADDR   BADSEC          ;HANDLER FOR BAD SECTOR
VSELER: .ADDR   SELERR          ;HANDLER FOR SELECT ERROR
VROERR: .ADDR   ROERR           ;HANDLER FOR R/O ERROR

;FOLLOWING ROUTINE AND CONSTANTS ARE MEDIUM-DEPENDANT

;ROUTINE TO PICK UP SKEWED SECTOR NUMBER

SKEWSC: LXI H,  SKEWTB          ;POINT TO TRANSLATION TABLE
        MVI B,  0               ;MAKE SECTOR INTO FULL WORD
        DAD     B               ;INDEX INTO TABLE
        MOV C,  M               ;PICK UP TRANSLATED SECTOR
        JMP     CBSTSC          ;BIOS VECTOR TO SET SECTOR
; (THIS IS THE ONLY SUCH CALL)
```

```
        .BYTE 0

;TABLE OF SECTORS, LENGTH 26. FOR SKEW

SKEWTB: .BYTE $01,$07,$0D,$13,$19
        .BYTE $05,$0B,$11,$17
        .BYTE $03,$09,$0F,$15
        .BYTE $02,$08,$0E,$14,$1A
        .BYTE $06,$0C,$12,$18
        .BYTE $04,$0A,$10,$16

        .BYTE 0,0,0,0,0,0

;CONSTANTS (READ ONLY) DEFINING DISK LAYOUT

SECPTK: .BYTE $1A       ;SECTORS PER TRACK
MAXFCB: .BYTE $3F       ;MAX FCB IN DISK DIRECTORY
LSECPC: .BYTE 3         ;LOG2 OF CLUSTER SIZE. 8 SECTORS PER ALLOC UNIT
ALLMSK: .BYTE 7         ;MASK FOR SECTOR WITHIN ALLOC UNIT
MXCLUS: .BYTE $F2       ;N CLUSTERS FOR ALLOCATOR
INIALO: .BYTE $C0       ;INITIAL ALLOC. ALWAYS PRESERVE FIRST 2 CLUSTERS,
                        ; WHICH ARE THE DIRECTORY
TKBIAS: .BYTE $02       ;BIAS ON ALL COMPUTED TRACKS. SKIPS BOOT IMAGE.
^L
;HERE TO HANDLE CALLS THRU 5.

BDOSH:  XCHG                    ;USER'S INFO ADDR TO HL
        SHLD    USERDE          ;SAVE IT HERE
        LXI M,  USRFCN          ;PUT USER'S FUNCTION HERE
        MOV M,  C               ; ..
        LXI H,  0               ;SAVE USER'S STACK POINTER
        DAD     SP              ;WHICH TAKES A BIT OF EFFORT
        SHLD    USERSP          ;STASH IT HERE
        LXI SP, BDOSSP          ;LOCAL STACK
        CALL    BDOS00          ;CALL GUTS AS A SUBR
        LHLD    USERSP          ;RESTORE USER STACK
        SPHL                    ; ..
        LHLD    ANSWER          ;GET ANSWER TO USER
        MOV A,  L               ;IN A AND B FOR PL/M CODE
        MOV B,  H
        RET

BADSEC: LXI H,  MSADSC          ;BAD SECTOR MESSAGE
        CALL    ERRPRT          ;ERROR MSG PRINTER
        CPI     CH.ATN          ;USER TYPED ^C?
        JZ      $0000           ;IF SO, REBOOT
        RET                     ;IF NOT, ACCEPT BAD DATA

SELERR: LXI H,  MSELEC          ;'SELECT' MESSAGE
        JMP     ERROR1          ;BIGGER ERROR MSG, AND REBOOT

ROERR:  LXI H,  MRONLY          ;'R/O' MESSAGE
ERROR1: CALL    ERRPRT          ;TYPE BIGGER ERR MSG
        JMP     $0000           ;RE-BOOT AS IF ^C

ERRPRT: PUSH    H               ;SAVE SPECIFIC MSG
        CALL    BDCRLF          ;TYPE A CRLF
        LDA     CURDSK          ;GET DRIVE NUMBER
        ADI     CH.A            ;MAKE IT A LETTER, A-D
```

```
        STA     MSG002          ;STUFF IT INTO TEXT OF MSG
        LXI B,  MSG001          ;GET START OF MSG
        CALL    DLSOUT          ;TYPE THE GENERAL MSG
        POP     B               ;GET SPECIFIC MSG
        CALL    DLSOUT          ;TYPE IT, TOO
        JMP     TYISVQ          ;GOBBLE A CHARACTER AND RETURN
^L
;TEXT FOR ABOVE MESSAGE PRINTERS

MSG001: .ASCII /BDOS ERR ON /
MSG002: .ASCII / : $/
MBADSC: .ASCII /BAD SECTOR$/
MSELEC: .ASCII /SELECT$/
MRONLY: .ASCII .R/O$.

;ROUTINE TO READ A CHAR FROM TYI. FIRST SEE IF ONE SAVED ALREADY.

TYISVQ: LXI H,  SAVCH           ;SEE IF THERE WAS A SAVED CH,
        MOV A,  M               ; FROM STOPPING TYPEOUT
        MVI M,  0               ;SEE IT ONLY ONCE
        ORA     A               ;WAS THERE ONE?
        RNZ                     ;RETURN IT IF SO
        JMP     CBCTYI          ;ELSE GO GET ONE FROM BIOS

;GET AND ECHO A CHARACTER

RDCONS: CALL    TYISVQ          ;GET SAVED, OR NEW, CHAR
        CALL    CKPRNT          ;CHECK IT FOR CONTROLS
        RC                      ;RETURN IF NON-TYPING CTL
        PUSH    PSW             ;ECHO IT.
        MOV C,  A               ;DIFFERENT AC FOR TYO
        CALL    CTYO2           ;TYPE IT OUT
        POP     PSW
        RET
^L
;CHECK FOR PRINTABLE CHAR. SIMPLE CONTROLS ARE PRINTABLE.
; RETURN WITH CY FLAG ON IF ITS A NON-PRINTABLE CTL CHAR, NEEDING "^"

CKPRNT: CPI     CH.CR
        RZ
CPI     CH.LF
        RZ
CPI     CH.TAB
        RZ
CPI     CH.CTL
        RET

CHKKBD: LDA     SAVCH           ;GET ANY SAVED CHAR
        ORA     A
        JNZ     CHKKB2          ;RETURN IT
        CALL    CBTTCK          ;CHECK KEYBOARD STAT, IN BIOS
        ANI     $01
        RZ                      ;RETURN IF NONE THERE
        CALL    CBCTYI          ;THERE'S SOMETHING READY. GET IT.
        CPI     CH.XOF
        JNZ     CHKKB1          ;JUMP UNLESS STOP REQUEST
        CALL    CBCTYI          ;IN WHICH CASE WAIT FOR ANOTHER CH  <-- HA!
        CPI     CH.AIN          ;IS IT A CONTROL C?
        JZ      $0000           ;IF SO, RE-BOOT
        XRA     A               <--  ;AFTER RESUMING, RETURN 0
```

```
        RET

CHKKB1: STA     SAVCH           ;SOMETHING BUT NOT XOFF. SAVE IT.
CHKKB2: MVI A,  $01             ;SAY HAVE SOMETHING TO READ
        RET
^L
;CONSOLE TYO ROUTINE

CTYO:   PUSH    B               ;SAVE CHAR IN C
        CALL    CHKKBD          ;CHECK FOR BEING FROZEN
        POP     B
        PUSH    B               ;RESTORE AND SAVE CHAR AGAIN
        CALL    CBCTYO          ;TTY OUTPUT IN BIOS
        POP     B               ;GET BACK CHAR AND SAVE AGAIN
        PUSH    B
        LDA     LPTFLG          ;COPYING ON PRINTER?
        ORA     A
        CNZ     CBLPTO          ;IF SO, PRINT IT.
        POP     B               ;GET THE CHAR AGAIN
        MOV A,  C               ;COPY IN A
        LXI H,  HPOS            ;POINT TO COLUMN POSITION
        CPI     CH.RUB          ;A RUBOUT DOESN'T SPACE
        RZ                      ;SO DON'T. DONE.
        INR     M               ;STEP HPOS
        CPI     CH.CTL          ;CONTROL CHAR?
        RNC                     ;IF NOT, ALL DONE.
        DCR     M               ;YES. DON'T SPACE AFTER ALL
        CPI     CH.LF           ;ON LF (SHOULD BE CR),
        RNZ
MVI M,  0               ;CLEAR HPOS
        RET
^L
;HERE FOR PRETTY TYO, WITH ^x FORMAT AND TAB SIMULATION

CTYO1:  MOV A,  C               ;COPY CHAR
        CALL    CKPRNT          ;CHECK FOR NEEDING UPARROW
        JNC     CTYO2           ;JUMP IF DON'T.
        PUSH    PSW             ;SAVE CHAR IN A
        MVI C,  CH.UPA          ;GET AN UPARROW
        CALL    CTYO            ;TYPE UPARROW
        POP     PSW             ;GET BACK CHAR
        ORI     $40             ;MAKE UN-CONTROL.
        MOV C,  A               ;BACK TO RIGHT AC
CTYO2:  MOV A,  C               ;COMMON TYO ROUTINE
        CPI     CH.TAB          ;TAB CHAR?
        JNZ     CTYO            ;IF NOT, TYPE IT AS IS.
CTYOTL: MVI C,  CH.SPC          ;SIMULATE TAB WITH SPACES
        CALL    CTYO            ;TYPE A SPACE
        LDA     HPOS            ;ENOUGH TO MAKE MULT OF 8?
        ANI     $07
        JNZ     CTYOTL          ;IF NOT, TYPE SOME MORE.
        RET

NMCRLF: MVI C,  CH.NUM          ;NUMBER SIGN
        CALL    CTYO            ;TYPE THE NUMBERSIGN
BDCRLF: MVI C,  CH.CR           ;TYPE CR
        CALL    CTYO
        MVI C,  CH.LF           ;AND LINEFEED
        JMP     CTYO            ;TYPE AND RETURN
```

```
;TYPE STRING UP TO TERMINATING DOLLARSIGN

DLSOUT: LDAX    B               ;GET CURRENT CHAR
        CPI     CH.DOL          ;IS IT A DOLLAR?
        RZ                      ;IF SO, RETURN WITHOUT PRINTING IT
        INX     B               ;ELSE, STEP FOR NEXT ONE,
        PUSH    B               ;SAVE POINTER,
        MOV C,  A               ;MOVE CURRENT CHAR TO C,
        CALL    CTYO2           ;AND TYPE CURRENT CHAR
        POP     B               ;GET BACK MEMORY POINTER
        JMP     DLSOUT          ;GO DO ANOTHER CHAR
;
;HERE FOR READ STRING BUFFERED (PSIN) COMMAND

PSIN:   LHLD    USERDE                                  ; 326B  2A 0A 33
        MOV C,  M                                       ; 326E  4E
        INX     H                                       ; 326F  23
        PUSH    H                                       ; 3270  E5
        MVI B,  0                                       ; 3271  06 00
PSIN1:  PUSH    B               ;SAVE SOME REGS
        PUSH    H
PSIN2:  CALL    TYISVQ          ;RAW TYI, OR SAVED ONE FROM TYO/STOP
        ANI     CH.MSK          ;JUST 7 BITS
        POP     H
        POP     B
        CPI     CH.CR           ;HANDLE CAR RET
        JZ      PSIN13          ;TERMINATE STRING
        CPI     CH.RUB          ;RUBOUT?
        JNZ     PSIN3           ;JUMP IF NOT
        MOV A,  B               ;HANDLE CHAR DELETION
        ORA     A                                       ; 3287  B7
        JZ      PSIN1                                   ; 3288  CA 73 32
        MOV A,  M                                       ; 328B  7E
        DCR     B                                       ; 328C  05
        DCX     H                                       ; 328D  2B
        JMP     PSIN11                                  ; 328E  C3 E6 32

PSIN3:  CPI     $05             ;CONTROL E? (PHYS CRLF)
        JNZ     PSIN4                                   ; 3293  C2 9E 32
        PUSH    B                                       ; 3296  C5
        PUSH    H                                       ; 3297  E5
        CALL    BDCRLF                                  ; 3298  CD 53 32
        JMP     PSIN2                                   ; 329B  C3 75 32

PSIN4:  CPI     $10             ;CONTROL P?
        JNZ     PSIN5           ;JUMP IF NOT
        PUSH    H               ;SAVE STRING POINTER
        LXI H,  LPTFLG          ;TOGGLE PRINTER-COPY FLAG
        MVI A,  $01             ; ..
        SUB     M               ; ..
        MOV M,  A               ; ..
        POP     H               ;RESTORE POINTER
        JMP     PSIN1           ;GET ANOTHER CHAR.

PSIN5:  CPI     $18             ;CONTROL X?
        JZ      PSIN6                                   ; 32B1  CA B9 32
        CPI     $15             ;OR CONTROL U?
        JNZ     PSIN7                                   ; 32B6  C2 C0 32
PSIN6:  CALL    NMCRLF          ;HERE TO DELETE WHOLE LINE, TYPE "#"
        POP     H               ;  AND A CRLF
```

```
        JMP     PSIN            ;TRY AGAIN

PSIN7:  CPI     $12             ;CONTROL R?
        JNZ     PSIN10          ;JUMP IF NOT
        PUSH    B               ;YES. RETYPE LINE
        CALL    NMCRLF          ;TYPE "#" CRLF FIRST
        POP     B               ;RESTORE COUNT
        POP     H               ;AND POINTER
        PUSH    H               ;THEN RE-SAVE
        PUSH    B               ; AND COUNT
PSIN8:  MOV A,  B               ;CHECK COUNT FOR DONE
        ORA     A               ; ..
        JZ      PSIN9           ;QUIT IF DONE
        INX     H               ;NOT DONE. STEP TO ANOTHER CHAR
        MOV C,  M               ;PICK IT UP
        DCR     B               ;COUNT THEM
        PUSH    B               ;SAVE COUNT AND POINTER OVER TYPEOUT
        PUSH    H
        CALL    CTYO1           ;TYPE CHAR, WITH ^ PROCESSING
        POP     H
        POP     B
        JMP     PSIN8           ;TYPE WHOLE LINE

PSIN9:  POP     B               ;DONE RE-TYPING. RESTORE REAL COUNT
        JMP     PSIN1           ;AND GO GET ANOTHER CHAR

;HERE ON ORDINARY CHAR.

PSIN10: INX     H               ;SAVE IN BUFFER
        MOV M,  A
        INR     B               ;COUNT THEM
PSIN11: PUSH    B                                       ; 32E6  C5
        PUSH    H                                       ; 32E7  E5
        MOV C,  A                                       ; 32E8  4F
        CALL    CTYO1           ;TYPE CHAR
        POP     H                                       ; 32EC  E1
        POP     B                                       ; 32ED  C1
        MOV A,  M                                       ; 32EE  7E
        CPI     CH.ATN          ;WAS IT A CONTROL C?
        MOV A,  B                                       ; 32F1  78
        JNZ     PSIN12                                  ; 32F2  C2 FA 32
        CPI     $01             ;IF SO, WAS IT THE FIRST CHAR?
        JZ      $0000           ;IF SO, RE-BOOT.
PSIN12: CMP     C               ;UP TO FULL BUFFER?
        JC      PSIN1           ;IF NOT, GET MORE.
PSIN13: POP     H               ;BACK TO START OF BUFFER
        MOV M,  B               ;AND PUT THE COUNT THERE
        MVI C,  CH.CR           ;ECHO THE C-R AT END
        JMP     CTYO            ;TYPE CR

HPOS:   .BYTE 0                 ;HORIZ POSITION OF TTY
LPTFLG: .BYTE 0                 ;NONZERO TO COPY TYO TO PRINTER
SAVCH:  .BYTE 0                 ;BKJFN'ED TYI CHAR
CURDSK: .BYTE 0                 ;CURRENT DISK # (0-3)
USRFCN: .BYTE 0                 ;CALLER'S AC C, = BDOS FUNCTION
USERDE: .BYTE 0,0               ;CALLER'S DE, = ARG ADDRESS
ANSWER: .BYTE 0,0               ;ANSWER TO RETURN TO USER
USERSP: .BYTE 0,0               ;USER'S STACK POINTER
        .BLKB 48                ;STACK AREA
BDOSSP:                         ;BDOS HAS STACK ABOVE HERE
```

```
FRECHR: .BYTE $E5                ;FILL CHAR ON BLANK DISK, FOR DELETED FILES
FL
;HOME THE DISK AND SET THE TRACK NUMBER IN CORE TO ZERO

HOMSET: CALL    CBHOME           ;HOME THE DISK (BIOS). THIS IS THE
                                 ; ONLY SUCH CALL
        LHLD    TKBIAS           ;GO TO LOGICAL TRACK ZERO
        MOV C,  L                ;TO C FOR BIOS ARGUMENT
        CALL    CBSTTK           ;SET TRACK NUMBER (BIOS) FROM C
        LHLD    TKSEC0           ;THAT'S SECTOR NUMBER ZERO, TOO.
        MVI A,  0                ; ..
        MOV M,  A                ; ..
        INX     H                ;ITS TWO BYTES EACH
        MVI M,  0
        LHLD    TKPNTR           ;POINT TO TRACK NUMBER
        MOV M,                   ;CLEAR TRACK NUMBER
        RET

COMPTK: LHLD    TKSEC0           ;SECTOR 0 ON CURRENT TRACK
        LXI D,  CURRMW           ;THIS CLUSTER
        CALL    DEMIHL           ;(DE)-(HL)=>HL
        JNC     CMPTK1           ;JUMP IF TEST SECTOR .GE. TRACK'S SECTOR
        LHLD    TKSEC0           ;BACK UP ONE TK. REDUCE (HL).
        XCHG                     ;PUT IT IN DE
        LDA     SECPTK           ;SECTORS PER TRACK
        CALL    DEMIDA           ; (DE) - 0A => HL
        XCHG
DCX     H                ;CORRECT FOR MOD BY DEMIDA. BACK TO TKSEC0
        MOV M,  E                ;PUT BACK REDUCED SECTOR NUMBER
        INX     H
        MOV M,  D                ;AND STORE BACK IN (DE)
        LHLD    TKPNTR           ;POINT TO TRACK NUMBER
        DCR     M                ;REDUCE TRACK NUMBER
        JMP     COMPTK           ;LOOP TIL ON RIGHT TRACK

CMPTK1: LHLD    TKSEC0           ;STEP UP ONE TRACK
        LDA     SECPTK           ;SECTORS PER TRACK
        CALL    APLSMM           ; 0A + (HL) => HL
        SHLD    CTKTMP           ;TEMP, NEW SECTOR
        LXI D,  CURRMW           ;SECTOR NUMBER
        CALL    IDEMHL           ; (DE) - HL => HL
        JC      CMPTK2           ;JUMP IF DESIRED > TEST
        LHLD    TKSEC0           ;CURRENT REAL SECTOR 0 OF THIS TK
        PUSH    H
        LHLD    CTKTMP           ;TEST SECTOR
        XCHG
POP     H                ;STORE TEST AS CURRENT
        MOV M,  E
        INX     H
        MOV M,  D
        LHLD    TKPNTR           ;POINT TO TRACK NUMBER
        INR     M                ;MAKE IT CORRESPOND TO SECTOR
        JMP     CMPTK1           ;SEE IF THAT'S HIGH ENOUGH

CMPTK2: LHLD    TKPNTR           ;POINT TO TRACK NUMBER
        LDA     TKBIAS           ;OFFSET ALL LOGICAL TRACKS FOR BOOT
        ADD     M
        MOV C,  A                ;ARG TO BIOS
        CALL    CBSTTK           ;SET TRACK (BIOS) FROM C
        LHLD    TKSEC0           ;SECTOR ZERO ON CHOSEN TRACK
```

```
        LXI D,  CURRMW          ;SECTOR NUMBER DESIRED
        CALL    DEMIHL          ;16 BIT SUBTRACT, TO HL
        MOV C,  L               ;LOGICAL SECTOR ON THE TRACK
        CALL    SKEWSC          ;GET CORR'ING SECTOR FROM SKEW TBL,
        RET                     ; CALL BIOS WITH IT (ONLY SUCH CALL)

;COMMON ENTRY FOR READ AND WRITE. IF READ, C/1. IF WRITE, C/0.

RWCOM:  LXI H,  RWARG           ;SAVE READ OR WRITE COMMAND HERE
        MOV M,  C               ;READ OR WRITE COMMAND
        LDA     RWARG           ;PICK IT BACK UP
        RAR                     ;SEE WHICH IT IS
        JNC     RWCOMW          ;JUMP IF 0, FOR WRITE.
        CALL    CBREAD          ;CALL BIOS TO READ A SECTOR.
; (THIS IS THE ONLY SUCH CALL)
        STA     RWCOMT          ;STORE SUCCESS/FAIL CODE
        JMP     RWCOMF          ;COMMON CLEANUP FOR RD AND WRITE.

RWCOMW: CALL    CBWRIT          ;CALL BIOS TO WRITE A SECTOR.
; (THIS IS THE ONLY SUCH CALL)
        STA     RWCOMT          ;STORE SUCCESS/FAIL CODE
RWCOMF: LDA     RWCOMT          ;SEE IF THERE WAS AN ERROR
        CPI     0
        JNZ     RWCOME          ;IF ERROR, GO HANDLE IT.
        RET                     ;IF NO ERROR, RETURN FROM RD/WRITE

RWCOME: LXI H,  RWCOMZ          ;PLACE TO RETURN TO
        PUSH    H               ;SAVE FOR LATER POPJ
        LHLD    WBADSC          ;GET BAD SECTOR HANDLER (MAYBE PATCHED)
        PCHL                    ;GO TO IT.
RWCOMZ: RET                     ;IF HANDLER COMES BACK, ACCEPT THE ERR

;ALL READS COME THROUGH HERE

RDSEC:  MVI C,  $01             ;FLAG FOR READ
        CALL    RWCOM           ;CALL COMMON I/O ROUTINE
        RET

;ALL WRITES COME THROUGH HERE

WRSEC:  MVI C,  0               ;FLAG FOR WRITE
        CALL    RWCOM           ;CALL COMMON I/O ROUTINE
        RET

;GET MAP BYTE FOR CURRENT NR

GTMAPB: LHLD    LSECPC          ;CONSTANT, LOG SEC PER CLUSTER
        MOV C,  L
        LXI H,  CURRNR                          ; 33F8  21 D3 3D
        CALL    RARNM           ;GET M, DO (C) RAR'S, GIVING CLUSTER
        ADI     $10             ;STEP TO MAP AREA OF FCB
        MOV C,  A               ;FOR THIS CLUSTER
        MVI B,  0               ;AS ADDRESS (16 BITS)
        LHLD    USERDE          ;IN FCB
        DAD     B               ;ADDR OF THIS MAP BYTE
        MOV L,  M               ;THIS IS THE CLUSTER NUMBER
        MVI H,  0               ;TREAT AS THOUGH COULD BE 16 BITS
        SHLD    CURRMW          ;SAVE MAP BYTE (CLUSTER NUMBER)
        RET
```

```
;COMPUTE SECTOR NUMBER FROM CURRNR AND CURRMW

COMPSN: LHLD    LSECPC          ;SHIFT FACTOR, SECTORS/CLUSTER
        MOV C,  L
        LXI H,  CURRMW          ;GET THE CLUSTER NUMBER
        CALL    SHLMM           ;MAKE A SECTOR NUMBER
        LDA     ALLMSK          ;MASK FOR SECTOR IN CLUSTER
        PUSH    H
        LXI H,  CURRNR
        ANA     M               ;MASK SECTOR WITHIN CLUSTER
        POP     H               ;GET BACK CLUSTER AS SECTOR
        CALL    AORHL           ;OR IN SECTOR WITHIN CLUSTER
        SHLD    CURRMW          ;NOW HAVE FULL SECTOR NUMBER
        RET

;GET NR AND RC FROM FCB

STNRRC: LXI B,  $0020           ;GET NR BYTE
        LHLD    USERDE          ;IN FCB
        DAD     B
        MOV A,  M               ;PICK IT UP
        STA     CURRNR          ;SAVE RECORD NUMBER
        LXI B,  $000F           ;ALSO GET RC (REC COUNT)
        LHLD    USERDE          ;FROM FCB
        DAD     B
        MOV A,  M               ;PICK IT UP
        STA     CURRRC          ;SAVE RECORD COUNT
        RET

;UPDATE NR AND RC IN FCB

UDNRRC: LDA     CURRNR          ;RECORD NUMBER
        INR     A               ;STEP, PREPARE FOR NEXT RECORD
        LXI B,  $0020           ;POINT TO NR FIELD
        LHLD    USERDE          ;IN USER'S FCB
        DAD     B               ;STEP TO NR BYTE
        MOV M,  A               ;STORE IT
        LXI B,  $000F           ;SAME FOR RC BYTE
        LHLD    USERDE          ;IN FCB
        DAD     B                                       ; 3451  09
        LDA     CURRRC                                  ; 3452  3A D2 3D
        MOV M,  A               ;UPDATE IT.
        RET

GDIRSC: LDA     DOSFCB                                  ; 3457  3A F3 3C
        ANI     $FE             ;DIVIDE BY FOUR, GETS SECTOR NUMBER
        RAR                     ; FROM THE FCB NUMBER
        RAR
STA     DIRSEC          ;THIS IS THE DIRECTORY SECTOR NUMBER
        MOV C,  A               ;MAKE A DOUBLE BYTE OF IT
        MVI B,  0
        MOV H,  B               ;COPY IT
        MOV L,  C
        SHLD    CURRMW          ;READY TO COMPUTE TK FOR THIS SECTOR
        CALL    COMPTK          ;FIND TRACK AND SECTOR IN TRACK
        RET

;CHECKSUM ROUTINE FOR DIRECTORY SECTORS. VERY INEFFICIENT. COMPILED?

GETCKS: LXI H,  CKSTM2          ;SUM GOES HERE
```

```
        MVI M, 0                ;START WITH ZERO
        DCX     H               ;HERE IS THE COUNT OF BYTES
        MVI M, 0                ;WHICH ALSO STARTS AT 0
GTCKS1: MVI A, NSECCH-1         ;DONE ALL BYTES?
        LXI H, CKSTM1           ;CHECK COUNT
        CMP     M
        JC      GTCKS2          ;QUIT WHEN DONE
        LHLD    CKSTM1          ;NO, GET ADDRESS
        MVI H, 0                ;WITHIN BLOCK
        XCHG
LHLD    SYSDMA          ;COMPUTE ADDRESS IN BUFFER
        DAD     D
        LDA     CKSTM2          ;PARTIAL SUM
        ADD     M               ;ADD ANOTHER BYTE
        STA     CKSTM2          ;SAVE PARTIAL SUM
        LXI H, CKSTM1           ;AND COUNT ANOTHER ONE DONE
        INR     M
        JNZ     GTCKS1          ;LOOP
GTCKS2: LDA     CKSTM2          ;GET ANSWER
        RET

;OR ARG WITH BIT FOR CURR DISK

ORCURR: LXI H, ORCURT           ;SAVE ARG
        MOV M, C                ; ..
        LDA     CURDSK          ;NUMBER OF CURR DISK
        INR     A
        MOV C, A
        MVI A, 0x01             ;SINGLE BIT PATTERN
        CALL    RLCN            ;DO (C) RLC'S
        RAL                     ;BACK ONE
        LXI H, ORCURT           ;GET CALLER'S ARG
        ORA     M               ;OR IT IN WITH NEW BIT
        RET

;GET R/O BIT FOR CURRENT DISK

CHKRO:  LDA     CURDSK
        INR     A
        MOV C, A
        LXI H, ROVEC            ;READ ONLY DISKS
        CALL    RALNM           ;SHIFT IN RIGHT BIT
        RLC                     ;PUT IT IN B0
        RET

MAKERO: LHLD    ROVEC           ;DISKS ALREADY LOCKED
        MOV C, L
        CALL    ORCURR          ;OR IN BIT FOR CURRENT DISK
        STA     ROVEC           ;NOW IT'S LOCKED, TOO.
        RET

;CHECK R/O BIT AND GIVE ERR IF SET.

ERRRO:  CALL CHKRO              ;CHECK IT
        RAR                     ;BIT INTO CY
        JNC     ERRROX          ;JUMP IF OK TO WRITE
        LXI H, ERRROX           ;ELSE SET THIS RETURN PC
        PUSH    H               ;ON STACK FOR RET
```

```
ERRROX: RET
;L
;VERDIR VERIFIES DIRECTORY SECTORS BY CHECKSUMS. CATCHES H'WARE ERRORS,
; AND CATCHES CHANGING DISKS OUT FROM UNDER BDOS.
; C/0 SAYS CHECK IT AGAINST LAST VALUE. C/1 SAYS SET NEW VALUE.

VERDIR: LXI H,  VERDIT          ;LOCAL TEMP, CHECK CKSM OR NOT
        MOV M,  C
        LDA     DIRSEC          ;SECTOR FOR DIRECTORY
        CPI     $10             ;LEGAL SECTOR FOR DIR?
        JC      VERDI1          ;JUMP IF SO.
        RET                     ;IF NOT, PROB FF (FILE NOT FOUND)

VERDI1: LDA     VERDIT          ;SEE IF CKSUM KNOWN
        RAR
JNC     VERDI2          ;JUMP IF KNOWN
        CALL    GETCKS          ;COMPUTE CHECKSUM OF THIS BUFFER
        LHLD    DIRSEC          ;SECTOR IT CAME FROM
        MVI H,  0               ;16 BIT SECTOR NUMBER
        XCHG
LHLD    VECCKS          ;POINTER TO CHKSUM TABLE
        DAD     D
        MOV M,  A               ;HERE'S THE CHECKSUM, NOW KNOWN
        JMP     VERDI3

VERDI2: LHLD    DIRSEC          ;KNOWN. GET SECTOR NUMBER
        MVI H,  0               ;AS AN ADDR
        XCHG
LHLD    VECCKS          ;POINTER TO CHECKSUM TABLE
        DAD     D               ;WHERE CKS OF THIS DIR SEC GOES
        PUSH    H
        CALL    GETCKS          ;COMPUTE CHECKSUM OF THIS BUFFER
        POP     H               ;WHERE WE LAST PUT IT
        CMP     M               ;IS IT THE SAME?
        JZ      VERDI3          ;JUMP IF OK
        CALL    MAKERO          ;MAKE DRIVE UNWRITABLE, DIRECTORY ERRORS
VERDI3: RET

;PDATE DIRECTORY SECTOR ON DISK

WDDIR:  MVI C,  $01             ;SET NEW CKSUM INTO TABLE
        CALL    VERDIR          ; ..
        CALL    WRSEC           ;WRITE A SECTOR
        RET

;ET NEXT DIR SLOT NUMBER. GET ITS SECTOR IN CORE.

MAPNXT: LXI H,  MAPNXV          ;SAVE CHECK OR SET CKSUM FLAG
        MOV M,  C
        LDA     DOSFCB          ;LAST FCB NUMBER
        INR     A               ;STEP TO NEXT ONE.
        STA     DOSFCB          ;PUT IT BACK IN CORE
        MOV C,  A               ;COPY IT
        LDA     MAXFCB          ;HIGHEST LEGAL NUMBER?
        CMP     C
        JNC     MAPNX1          ;JUMP IF OK
        LXI H,  DOSFCB          ;ELSE SAY NOT FOUND
        MVI M,  $FF
        RET
```

```
MAPNX1: LDA     DOSFCB          ;NEW FCB NUMBER
        ANI     $03             ;NUMBER WITHIN A SECTOR
        ADD     A               ;AS A BYTE NUMBER IN SECTOR
        ADD     A
        ADD     A
        ADD     A
        ADD     A
        STA     CHROFS          ;CHARACTER OFFSET OF FCB IN DIR BUFFER
        CPI     0               ;FIRST FCB IN A SECTOR?
        JNZ     MAPNX2          ;IF NOT, HAVE IT IN CORE ALREADY
        CALL    GDIRSC          ;COMPUTE DIRECTORY SECTOR NEEDED
        CALL    RDSEC           ;READ A SECTOR
        LHLD    MAPNXV                                  ; 354C  2A DE 3D
        MOV C,  L               ;CHECK OR SET CKSUM FLAG
        CALL    VERDIR          ;DO THE CHECKSUMMING
MAPNX2: RET

;GET ALLOC BIT FOR CLUSTER IN C

GETBTC: LXI H,  GETBTT          ;CLUSTER NUMBER
        MOV M,  C
        LDA     GETBTT          ;OVER 8 (BITS PER BYTE)
        ANI     $FC
        RAR
RAR
RAR
MOV C,  A               ;AS AN ADDRESS
        MVI B,  0
        LHLD    CURALV          ;CURRENT DISK'S ALLOC VECTOR
        DAD     B               ;ADDR OF BIT IN TABLE
        LDA     GETBTT          ;NOW THE BIT NUMBER IN BYTE
        ANI     $07             ;BITS IN A BYTE
        INR     A               ;OFFSET BY ONE
        MOV C,  A
        CALL    RLCNM           ;PICK UP (HL) INTO A, AND DO (C) RLC'S
        RET

;MARK THE BIT TABLE WITH THE BIT IN E FOR CLUSTER IN C

MRKBTB: LXI H,  MRKBTW                                  ; 3572  21 E1 3D
        MOV M,  E               ;SAVE CALLER'S C
        DCX     H                                       ; 3576  2B
        MOV M,  C               ;SAVE CLUSTER NUMBER
        LHLD    MRKBTV          ;CLUSTER NUMBER AGAIN
        MOV C,  L               ;AS AN ADDRESS
        CALL    GETBTC          ;GET ALLOCATION BIT
        ANI     $FE             ;MASK OUT OLD VALUE
        LXI H,  MRKBTW          ;POINT TO NEW VALUE
        ORA     M               ;DEPOSIT THE NEW BIT
        PUSH    PSW             ;SAVE NEW BIT
        MVI A,  $07             ;MASK FOR BIT NUMBER
        DCX     H               ;BACK TO THE CLUSTER NUMBER
        ANA     M               ;MASK FOR BIT IN BYTE
        INR     A               ;OFFSET BY ONE AGAIN
        MOV C,  A               ;COPY FOR SUBR CONVENTION
        POP     PSW             ;NEW BITS AGAIN
        CALL    RALN            ;DO (C) RAL'S
        PUSH    PSW             ;SAVE IT, SHIFTED BACK
        MOV A,  M               ;CLUSTER NUMBER AGAIN
        ANI     $FC             ;DIV BY 8 FOR A BYTE NUMBER
```

```
        RAR
RAR
RAR
MOV C,  A               ;BYTE NUMBER MAKES ADDRESS AGAIN
        MVI B,  0
        LHLD    CURALV          ;CURRENT DISK'S ALLOC VECTOR
        DAD     B               ;WHERE THE CURRENT BYTE LIVES
        POP     B
        MOV C,  B               ;UPDATED BYTE
        MOV M,  C               ;TO VECTOR
        RET
^L
;SET OR CLEAR THIS FCB'S ALLOCATION IN BITTABLE

ALOFCB: LXI H,  ALOFCV          ;WILL SET BITS TO THIS (0 OR 1)
        MOV M,  C
        LDA     CHROFS          ;CHARACTER OFFSET OF FCB IN DIR BUFFER
        ADI     $10             ;MAP AREA
        STA     ALOFCW          ;SAVE MAP ADDR
ALOFC1: LDA     CHROFS          ;CHARACTER OFFSET OF FCB IN DIR BUFFER
        ADI     $1F             ;SEE IF DONE
        LXI H,  ALOFCW          ;WHERE WE ARE BY NOW
        CMP     M               ;DONE?
        JC      ALOFC3          ;JUMP IF ALL DONE
        LHLD    ALOFCW          ;NOT YET. POINT TO MAP BYTE
        MVI H,  0               ;AS AN ADDRESS
        XCHG
LHLD    SYSDMA          ;IN BUFFER
        DAD     D               ; ..
        MOV A,  M               ;GET A MAP BYTE
        STA     ALOFCX          ;KEEP A COPY OF IT
        CPI     0               ;IS THIS CLUSTER FREE?
        JZ      ALOFC2          ;IF SO, DON'T (DE)ALLOCATE IT
        LHLD    ALOFCX          ;NO, MUST (DE)ALLOCATE IT
        MOV C,  L               ;CLUSTER NUMBER
        LHLD    ALOFCV          ;0 OR 1 FOR THAT CLUSTER
        XCHG
CALL    MRKBTB          ;PUT BIT IN BIT TABLE
ALOFC2: LXI H,  ALOFCW          ;STEP TO NEXT BYTE OF MAP AREA
        INR     M
        JNZ     ALOFC1          ;LOOP UNTIL DONE
ALOFC3: RET
^L
;READ THE DIRECTORY AND ALLOCATE ALL FILES IN BIT TABLE

RDALOC: LXI H,  ANSARL                                  ; 35E0  21 F0 3C
        MVI M,  0                                       ; 35E3  36 00
        LXI H,  ROVEC                                   ; 35E5  21 7C 3D
        MVI M,  0               ;SAY NO DISKS ARE R/O
        LHLD    CURALV          ;CURRENT DISK'S ALLOC VECTOR
        LDA     INIALO
        MOV M,  A               ;ALLOCATE THE DIRECTORY CLUSTERS
        LXI H,  RDALOV          ;ADDR WITHIN ALLOC VECTOR
        MVI M,  $01             ;START WITH BYTE AFTER ABOVE INIT
RDALO1: MVI A,  NALLOV-1        ;DONE THEM ALL?
        LXI H,  RDALOV          ; ..
        CMP     M               ; ..
        JC      RDALO2          ;JUMP IF SO
        LHLD    RDALOV          ;NO, DO ANOTHER
        MVI M,  0
```

```
        XCHG
LHLD    CURALV          ;CURRENT DISK'S ALLOC VECTOR
        DAD     D               ;BYTE IN ALLOC VECTOR
        MVI M,  0               ;CLEAR ALLOCATION
        LXI H,  RDALOV          ;STEP THRU VECTOR
        INR     M               ; ..
        JNZ     RDALO1          ;LOOP TILL DONE WHOLE VECTOR THIS DSK
RDALO2: CALL    HOMSET          ;HOME THE DISK
        LXI H,  DOSFCB          ;SAY NO FCB NOW OPEN
        MVI M,  $FF
RDALO3: MVI C,  $01             ;MAP AN FCB, SETTING UP ALLOC VEC
        CALL    MAPNXT          ;GET NEXT FCB NUM, MAP ITS SECTOR INTO CORE
        LDA     DOSFCB          ;DONE ALL FCB'S?
        CPI     $FF             ;WHEN ALL DONE, RETURNS FF
        JNZ     RDALO4          ;KEEP ON TILL HIGHEST LEGAL NUMBER
        RET

RDALO4: LHLD    CHROFS          ;CHARACTER OFFSET OF FCB IN DIR BUFFER
        MVI H,  0               ;COMPUTE ADDR IN BUFFER
        XCHG
LHLD    SYSDMA
        DAD     D
        MOV A,  M               ;GET FIRST CHAR OF FCB
        CPI     $E5             ;DISK FILL CHAR, = BLANK ENTRY
        JZ      RDALO5          ;IF SO, THIS FCB IS A FREE ONE.
        LHLD    CHROFS          ;CHARACTER OFFSET OF FCB IN DIR BUFFER
        MVI H,  0               ;NOT FREE. COMPUTE UP THE NAME ADDR
        LXI B,  $0001           ;WHERE NAMES START
        DAD     B
        XCHG
LHLD    SYSDMA
        DAD     D
        MOV A,  M               ;GET FIRST CHAR OF NAME
        SUI     CH.DOL
        SUI     $01             ;MAKE 8-BIT MATCH FLAG
        SBB     A               ;A/FF IF CHAR WAS DOLLAR, ELSE A/0
        LXI H,  ANSARL
        ORA     M               ;SET ANSWER TO ONES IF ANY $'S FOUND
        MOV M,  A               ; ..
        MVI C,  $01             ;IN ANY CASE, ALLOCATE THE SPACE
        CALL    ALOFCB          ;SET ALLOCATION FOR THIS FCB IN BITTABLE
RDALO5: JMP     RDALO3          ;LOOP FOR ALL FCB'S

        RET                     ;EXTRA RET FROM COMPILER

;GET NEXT MATCHING FILE

STPFIL: LHLD    STPFIW          ;USER'S FCB FOR PREV LOOKUP
        SHLD    USERDE          ;RE-USE IT
STPFI5: MVI C,  0               ;REMEMBER TO CHECK CHECKSUMS
        CALL    MAPNXT          ;GET NEXT FCB NUM, MAP ITS SECTOR INTO CORE.
        LDA     DOSFCB          ;WHICH ONE DID IT FIND?
        STA     ANSARL          ;SAVE THE ANSWER
        CPI     $FF             ;WERE ANY FOUND?
        JNZ     STPFI1          ;JUMP IF SO
        RET                     ;RETURN IF NOT

STPFI1: LXI H,  STPFIX          ;NUMBER CHARS CHECKED SO FAR
        MVI M,  0               ;INITIALLY, NONE CHECKED
STPFI2: LXI H,  STPFIV          ;# CHARS TO CHECK
```

```
        LDA     STPFIX          ;NUMBER DONE
        SUB     M               ;ALL DONE?
        SBB     A               ;SET TO FF UNLESS ALL CHECKED, THEN 0
        LHLD    STPFIX          ;CHAR POSITION IN FCB
        MVI H,  0               ;AS AN ADDR
        XCHG
LHLD    USERDE
        DAD     D
        PUSH    PSW             ;SAVE ALL-CHECKED FLAG
        MOV A,  M               ;GET THIS CHAR FROM USER FCB
        STA     STPFIY          ;SAVE USER'S CHAR
        LDA     STPFIX          ;NUMBER CHECKED SO FAR
        PUSH    H
        LXI H,  CHROFS          ;CHARACTER OFFSET OF FCB IN DIR BUFFER
        ADD     M               ;CURRENT CHAR NUMBER IN BUFFER
        MOV C,  A               ;AS ADDRESS
        MVI B,  0
        LHLD    SYSDMA          ;AS ADDR IN BUFFER
        DAD     B
        POP     B               ;BACK TO USER FCB
        LDAX    B               ;CHAR FROM USER'S FCB
        SUB     M               ;COMPARE THEM
        SUI     $01             ;IF MATCH, SET CY
        SBB     A               ;AND MAKE FF IF MATCH, ELSE 0
        PUSH    PSW             ;SAVE MATCHED CHAR FLAG
        LDA     STPFIY          ;USER'S CHAR
        SUI     $3F             ;QUESTIONMARK? (WILDCARD)
        SUI     $01             ;MAKE FLAG FOR THAT
        SBB     A
        POP     B               ;MATCHED CHAR THIS TIME FLAG
        MOV C,  B
        ORA     C               ;ONES IF MATCHED OR WILDCARD
        POP     B
        MOV C,  B               ;MORE CHARS LEFT FLAG
        ANA     C
        RAR
JNC     STPFI3          ;JUMP IF ALL DONE OR DIDNT MATCH
        LXI H,  STPFIX          ;HAVE TO CHECK SOME MORE
        INR     M               ;STEP TO NEXT CHAR NUMBER
        JMP     STPFI2          ;GO DO ANOTHER CHAR

;HERE IF NON-MATCH OR ALL DONE.

STPFI3: LXI H,  STPFIV          ;NUMBER CHARS TO CHECK
        LDA     STPFIX          ;NUMBER CHECKED
        CMP     M               ;ALL DONE?
        JNZ     STPFI4          ;IF NOT, NON-MATCH. QUIT.
        RET                     ;ELSE ALL OK, SO RETURN

STPFI4: JMP     STPFI5          ;GO TRY ANOTHER FCB

        RET                     ;EXTRA RET FROM COMPILER
L
;ROUTINE TO LOOK FOR A FILENAME, OR SPECIFIC EXTENT.

LOOKUP: LXI H,  LOOKUV          ;SAVE NUMBER CHAR TO VERIFY
        MOV M,  C               ;HERE
        LDA     LOOKUV
        STA     STPFIV          ;SAVE FOR LATER SCAN
        LHLD    USERDE          ;COPY USER'S FCB ADDR
```

```
        SHLD    STPFIW          ;FOR LATER
        LXI H,  DOSFCB          ;ASSUME FAILURE
        MVI M,  $FF             ;FLAGGED BY FF
        CALL    HOMSET          ;HOME AND SET TRACK
        CALL    STPFIL          ;GET NEXT MATCHING FILE
        RET

;FUNCTION ROUTINE FOR DELETE FILE CALLS THIS TO DO THE WORK.

DODEL:  CALL    ERRRO           ;CHECK R/O AND ERROR OUT IF SET
        MVI C,  $0C             ;CHECK THIS MANY CHARS
        CALL    LOOKUP          ;FIND USER'S FILE
DODEL1: LDA     DOSFCB          ;WAS THERE ONE?
        CPI     $FF
        JNZ     DODEL2          ;JUMP IF SO
        RET                     ;ELSE JUST RETURN

;DOING A DELETE, AND THERE IS A FILE TO DO.

DODEL2: MVI C,  0               ;DEALLOCATE FLAG FOR BITIABLE
        CALL    ALOFCB          ;CLEAR ALLOC FOR THIS FCB IN BITIABLE
        LHLD    CHROFS          ;CHARACTER OFFSET OF FCB IN DIR BUFFER
        MVI H,  0               ;AS AN ADDR
        XCHG
LHLD    SYSDMA
        DAD     D
        MVI M,  $E5             ;DATA FILL CHAR, MAKING FREE FCB
        CALL    UPDDIR          ;UPDATE DIR SEC ON DISK
        CALL    STPFIL          ;GET NEXT MATCHING FILE OR EXTENT
        JMP     DODEL1          ;LOOP FOR ALL

        RET                     ;EXTRA RET FROM COMPILER
^L
;FROM WRITE NEXT RECORD ROUTINE
; FIND A FREE CLUSTER NEAR CLUSTER IN C.

ALLOC:  LXI H,  ALO.LO          ;SAVE DESIRED CLUSTER
        MOV M,  C
        LDA     ALO.LO          ;AS BOTH LOW AND HIGH TESTS
        STA     ALO.HI          ; ..
ALLOC1: LXI H,  MXCLUS          ;NUMBER OF CLUSTERS ON DISK
        LDA     ALO.HI          ;TEST HIGH FOR BEING OFF TOP
        SUB     M
        SBB     A
        PUSH    PSW             ;SAVE HI LEGAL FLAG
        MVI A,  0               ;DOWN TO 0 ON LOW SIDE?
        LXI H,  ALO.LO
        SUB     M               ;I.E., IS LO LEGAL?
        SBB     A
        POP     B               ;HIGH LEGAL FLAG
        MOV C,  B               ;INEFFICIENCY OF COMPILER
        ORA     C               ;SEE IF EITHER ONE LEGAL
        RAR
JNC     ALLOC4          ;JUMP IF NEITHER ONE GOOD
        MVI A,  0               ;NOW SEE WHICH IS OK.
        LXI H,  ALO.LO          ;BY WASTEFULLY RECOMPUTING
        SUB     M               ;WHETHER .LO IS LEGAL
        SBB     A
        ANI     $01             ;1 IF NOT ZERO
        PUSH    PSW
```

```
        MOV A,  M               ;.LO
        POP     B
        MOV C,  B               ;1 IF NOT FREE
        SUB     C               ;DESIRED -(1 OR 0)
        MOV M,  A               ;STORE NEW TARGET
        INX     H               ;ALO.HI
        MOV A,  M
        LXI H,  MXCLUS          ;NUMBER OF CLUSTERS ON DISK
        SUB     M               ;IS HIGH AT MAX?
        SBB     A               ;IF LEGAL, MAKE FF
        ANI     $01             ;1 IF LEGAL, 0 IF NOT
        LXI H,  ALO.HI          ;STEP HI UP IF LEGAL
        ADD     M               ; ..
        MOV M,  A               ; ..
        LHLD    ALO.HI          ;SEE IF .LO IS FREE
        MOV C,  L
        CALL    GETBTC          ;GET ITS ALLOCATION BIT
        RAR                     ;TEST IT
        JC      ALLOC2          ;JUMP IF NOT FREE
        LDA     ALO.HI          ;IT'S FREE, USE IT.
        RET
^L
; MORE OF SEARCH FOR FREE CLUSTER

ALLOC2: LHLD    ALO.LO          ;SEE IF THE LOW SIDE GUY IS FREE
        MOV C,  L
        CALL    GETBTC          ;GET ITS ALLOC BIT
        RAR
JC      ALLOC3          ;JUMP IF NOT FREE
        LDA     ALO.LO          ;FREE. USE IT.
        RET

ALLOC3: JMP     ALLOC1          ;GO BACK AND TRY AGAIN

ALLOC4: MVI A,  0               ;THERE ARE NO FREE CLUSTERS
        RET

;COPY NAME STRING INTO SYSTEM DIRECTORY. FOR MAKE OR RENAME.

CPYNAM: LXI H,  CPYNAM          ;TWO ARGUMENTS
        MOV M,  E               ;COUNT OF CHARS TO MOVE
        DCX     H
        MOV M,  C               ;WHERE IN USER FCB TO GET CHARS
CPYNM1: LDA     CPYNAM          ;GET THE COUNT
        DCR     A               ;COUNT IT DOWN
        STA     CPYNAM          ;IN CORE, TOO
        CPI     $FF             ;ALL DONE?
        JZ      CPYNM2          ;JUMP IF SO
        LDA     CPYNAM          ;IF NOT, GO GET ANOTHER CHAR FROM FCB
        LXI H,  CPYNAV          ;AT THIS OFFSET IN USER FCB
        ADD     M
        MOV C,  A               ;MAKE 16 BIT ADDR
        MVI B,  0
        LHLD    USERDE          ;IN FCB
        DAD     B
        LDA     CHROFS          ;CHARACTER OFFSET OF FCB IN DIR BUFFER
        PUSH    H               ;SAVE WHERE USER'S CHAR IS
        LXI H,  CPYNAM          ;COMPUTE WHERE IT IS IN DIR BUFFER
        ADD     M
        MOV C,  A               ;AGAIN, 16 BIT ARITHMETIC
```

```
        MVI B, 0
        LHLD    SYSDMA          ;IN SYSTEM BUFFER
        DAD     B
        POP     B               ;WHERE USER CHAR IS
        LDAX    B               ;GET USER CHAR
        MOV M, A                ;PUT IN SYSTEM DIR
        JMP     CPYNM1          ;SEE IF ALL DONE

CPYNM2: CALL    GDIRSC          ;COMPUTE DIR SECTOR AND TK
        CALL    UPDDIR          ;GO WRITE IT
        RET
^L
;COPY A WHOLE FCB

CPYFCB: MVI E, $20              ;COPY THIS MANY CHAR'S
        MVI C, 0                ;AT THIS OFFSET
        CALL    CPYNAM          ;COPY USER NAME INTO DIR
        RET

;RENAME FILE ROUTINE

DOREN:  CALL    ERRRO           ;CHECK R/O AND ERROR OUT IF SET
        MVI C, $0C              ;CHECK THIS MANY CHARS
        CALL    LOOKUP          ;TRY TO FIND A FILE
        LHLD    USERDE          ;POINT TO NEW NAME
        LXI B, $0010            ;IT STARTS HERE
        PUSH    H               ;SAVE START OF FCB
        LHLD    USERDE
        DAD     B               ;FIND NEW NAME
        POP     D               ;COPY ENTRY TYPE BYTE
        LDAX    D
        MOV M, A                ;GET IT
DOREN1: LDA     DOSFCB          ;THERE WAS A FILE, WASN'T THERE?
        CPI     $FF             ; ..
        JZ      DOREN2          ;JUMP IF NOT
        MVI E, $0C              ;COPY THIS MANY CHARS
        MVI C, $10              ;FROM HERE IN USER'S FCB
        CALL    CPYNAM          ;COPY NEW NAME INTO DISK DIR
        CALL    STPFIL          ;GET NEXT MATCHING FILE OR EXTENT
        JMP     DOREN1          ;LOOP TILL ALL FCB'S FOUND

DOREN2: RET                     ;SPARE RET FROM COMPILER
^L
;OPEN FILE ROUTINE. DO LOOKUP, COPY MAP AREA FROM DISK DIR TO FCB.

OPENFI: MVI C, $0D              ;CHECK THIS MANY CHARS
        CALL    LOOKUP          ;GO FIND USER'S NAME
        LDA     DOSFCB          ;GET ANSWER
        CPI     $FF             ;WAS IT THERE?
        JZ      OPENNF          ;IF NOT, GO GIVE ERROR
        LXI H, OPNFIV           ;COPY MAP AREA INTO FCB IN USER SPACE
        MVI M, $0D              ;INITIAL INDEX
OPNFI1: MVI A, $1F              ;END TEST. DONE THEM ALL?
        LXI H, OPNFIV           ; ..
        CMP     M               ; ..
        JC      OPENNF          ;JUMP IF DONE
        LDA     OPNFIV          ;MORE TO MOVE. BUILD ADDRESS
        LXI H, CHROFS           ;CHARACTER OFFSET OF FCB IN DIR BUFFER
        ADD     M
        MOV C, A                ;AS WORD ADDR
```

```
        MVI B, 0
        LHLD    SYSDMA          ;IN SYSTEM'S DIR COPY
        DAD     B
        PUSH    H
        LHLD    OPNFIV          ;MOVE IT TO SAME PLACE IN USER FCB
        MVI H, 0
        XCHG
LHLD    USERDE          ;WHICH IS HERE
        DAD     D
        POP     B               ;GET POINTER TO SYS DIR CHAR
        LDAX    B               ;GET THE CHAR
        MOV M, A                ;PUT IT IN USER FCB
        LXI H, OPNFIV           ;STEP THE INDEX
        INR     M               ; ..
        JNZ     OPNFI1          ;LOOP TILL ALL COPIED
OPENNF: RET

;CLOSE FILE LOGIC

CLOSFI: LXI H, ANSARL           ;WHERE USER GETS ANSWER
        MVI M, 0                ;SUCCESS FLAG
        CALL    CHKRO           ;IS DISK WRITABLE?
        RAR                     ; ..
        JNC     CLSFI1          ;JUMP IF SO
        RET                     ;RETURN IF NOT. IT'S CLOSED.

CLSFI1: MVI C, $0D              ;THIS MANY CHARS
        CALL    LOOKUP          ;GET THE FCB
        LDA     DOSFCB          ;THERE WAS ONE, YES?
        CPI     $FF
        JZ      CLSFI2          ;JUMP IF NO
        CALL    CPYFCB          ;COPY WHOLE FCB, UPDATING SIZE, ALLOC...
CLSFI2: RET

;MAKE FILE ROUTINE

MAKEFI: CALL    ERRO            ;CHECK R/O AND ERROR OUT IF SET
        LHLD    USERDE                                  ; 3845  2A 0A 33
        SHLD    MAKFIW          ;SAVE NAME TO MAKE
        LXI H, FRECHR           ;LOOK FOR A FREE FILE
        SHLD    USERDE          ;THAT'S A FILENAME, SORT OF.
        MVI C, $01              ;ITS ONE CHAR LONG
        CALL    LOOKUP          ;LOOK FOR A FILE BY THAT NAME
        LDA     DOSFCB          ;ANY FREE?
        CPI     $FF
        JZ      MAKFI3          ;JUMP IF NO ROOM
        LHLD    MAKFIW          ;GET BACK NAME TO MAKE
        SHLD    USERDE          ;USER'S FCB
        LXI H, MAKFIV           ;HERE WE COUNT THE CHARS
        MVI M, $0D              ;INITIAL CHAR TO CHECK
MAKFI1: MVI A, $1F              ;DONE?
        LXI H, MAKFIV
        CMP     M
        JC      MAKFI2          ;JUMP WHEN DONE
        LHLD    MAKFIV          ;COMPUTE ADDR IN USER'S FCB
        MVI H, 0
        XCHG
LHLD    USERDE          ;USER'S FCB ADDR
        DAD     D
        MVI M, 0                ;CLEAR THESE CHARS
```

```
        LXI H,  MAKFIV          ;DO THEM ALL
        INR     M               ;STEP THRU THEM
        JNZ     MAKFI1          ;LOOP TILL ALL DONE
MAKFI2: CALL    CPYFCB          ;COPY WHOLE FCB TO DISK BUFFER
MAKFI3: RET

;STEP TO THE NEXT EXTENT

STPXTN: LXI H,  STPXTV          ;SAVE RD/WR FLAG
        MOV M,  C
        CALL    CLOSFI          ;CLOSE THE OLD EXTENT
        LDA     DOSFCB          ;DID THAT WORK?
        CPI     $FF             ; ..
        JNZ     STPXT1          ;JUMP IF SO
        RET                     ;FAILED TO CLOSE.

STPXT1: LXI B,  $000C           ;EXTENT BYTE
        LHLD    USERDE          ;IN USER'S FCB
        DAD     B               ; FOR FILE JUST CLOSED
        INR     M               ;STEP TO NEXT EXTENT
        MVI C,  $0D             ;LOOK FOR THAT EXTENT
        CALL    LOOKUP          ; ..
        LDA     DOSFCB          ;FIND IT?
        CPI     $FF
        JNZ     STPXT3          ;JUMP IF SO
        LDA     STPXTV          ;NOT FOUND. WHAT NEXT?
        RAR                     ;WERE WE READING OR WRITING?
        JNC     STPXT2          ;JUMP IF WRITING
        RET                     ;READING. JUST EOF, THEN.

STPXT2: CALL    MAKEFI          ;WRITING. MAKE ANOTHER EXTENT.
        JMP     STPXT4          ;IT'S OPEN.

STPXT3: CALL    OPENFI          ;OPEN THE NEXT EXTENT
STPXT4: LDA     DOSFCB          ;OPENED OK?
        CPI     $FF
        JNZ     STPXT5          ;JUMP IF SO
        LXI H,  ANSARL          ;NOT ABLE TO OPEN NEXT EXTENT.
        MVI M,  $01             ;GIVE THIS ERROR RETVALUE
        RET                     ;RETURN TO USER

STPXT5: CALL    STNRRC          ;GET NR AND RC FROM FCB
        LXI H,  ANSARL          ;USER ANSWER
        MVI M,  0               ;IS SUCCESS
        RET

SETDMA: LDA     FRZDMA          ;DMA FROZEN?
        RAR
JNC     SETDM1          ;JUMP IF SO
        LHLD    USRDMA          ;ALLOWED TO MOVE.
        MOV B,  H               ;SET TO WHAT USER LAST ASKED FOR
        MOV C,  L
        CALL    CBSTMA          ;BIOS SET DMA ADDRESS
SETDM1: RET

;RESET DMA TO STANDARD

RSTDMA: LDA     FRZDMA          ;ALLOWED TO MOVE DMA?
        RAR
JNC     RSTDMY          ;IF FROZEN, SKIP THIS.
```

```
        LHLD    SYSDMA          ;OK TO MOVE IT.
        MOV B,  H                                       ; 38F0  44
        MOV C,  L                                       ; 38F1  4D
        CALL    CBSTMA          ;BIOS SET DMA ADDRESS
RSTDMX: RET                                             ; 38F5  C9

;HERE TO READ NEXT RECORD

RDNXTR: CALL    STNRRC          ;GET NR AND RC FROM FCB
        LDA     CURRNR          ;WHERE WERE WE?
        LXI H,  CURRRC          ;HOW FAR CAN WE GO?
        CMP     M               ;OVERDONE IT?
        JC      RDNXT2          ;JUMP IF OK
        LXI H,  ANSARL          ;IF TOO FAR, GIVE EOF RETURN
        MVI M,  $01             ;TO USER'S ANSWER
        LDA     CURRNR          ;WERE WE TO END OF EXTENT?
        CPI     $80             ; ..
        JNZ     RDNXT1          ;JUMP IF NOT
        MVI C,  $01             ;YES. SAY READING
        CALL    STPXIN          ;AND TRY FOR ANOTHER EXTENT
RDNXT1: LXI H,  CURRNR          ;SAY FIRST RECORD IN EXTENT
        MVI M,  0
        LDA     ANSARL          ;WHAT WAS ANSWER?
        CPI     0               ;SUCCESS?
        JZ      RDNXT2          ;IF SO, MORE TO DO.
        RET                     ;ELSE GIVE UP HERE.

RDNXT2: CALL    GTMAPB          ;GET MAP BYTE FOR CURRENT NR
        LHLD    CURRMW          ;THAT'S CLUSTER NUMBER
        MOV A,  L               ;AS 8 BIT NUMBER
        CPI     0               ;IS IT ALLOCATED?
        JNZ     RDNXT3          ;JUMP IF SO
        LXI H,  ANSARL          ;NOT ALLOCATED. SAY EOF.
        MVI M,  $01             ;ALLOCATION CAN'T BE SPARSE.
        JMP     RDNXT4          ;SO QUIT ON FIRST ZERO

RDNXT3: CALL    COMPSN          ;GET SECTOR NUMBER, FROM CURRMW, CURRNR
        CALL    COMPTK          ;AND COMPUTE THE TRACK FROM THAT
        CALL    SETDMA          ;SET UP TO USER'S (OR STD) DMA ADDR
        CALL    RDSEC           ;READ A SECTOR
        CALL    RSTDMA          ;BACK TO WHAT USER SAID
        CALL    UDNRRC          ;UPDATE NR AND RC IN FCB
RDNXT4: RET                     ;END OF READ-NEXT-RECORD

;WRITE NEXT RECORD ROUTINE

WTNXTR: CALL    ERRRO           ;CHECK R/O AND ERROR OUT IF SET
        CALL    STNRRC          ;GET NR AND RC FROM FCB
        MVI A,  $7F             ;ABOUT TO GO OFF END OF EXTENT?
        LXI H,  CURRNR          ;WEREN'T ABLE TO MAKE A NEW EXTENT, THEN.
        CMP     M
        JNC     WTNXT1          ;JUMP IF WITHIN AN EXT
        LXI H,  ANSARL          ;REPLY FULL.
        MVI M,  $01             ;COULDN'T EXTEND FILE.
        JMP     WTNXT8          ;RETURN TO USER

WTNXT1: CALL    GTMAPB          ;GET CLUSTER NUMBER FROM MAP
        LHLD    CURRMW          ;PICK IT UP
        MOV A,  L
```

```
        CPI     Ø               ;IS IT ALLOCATED?
        JNZ     WTNXT4          ;JUMP IF SO
        LXI H,  WTNXTV          ;NO, MUST ALLOCATE IT
        MVI M,  Ø               ;PREV ALLOC = Ø
        LDA     CURRNR          ;CONVERT TO CLUSTER NUMBER
        ANI     $FC             ;BY DIVIDING BY 8
        RAR
RAR
RAR
ADI     $1Ø             ;STEP INTO FCB TO MAP AREA
        INX     H               ;TO WTNXTW
        MOV M,  A               ;SAVE FCB OFFSET FOR THIS CLUSTER
        MOV C,  A               ;IS IT THE INITIAL ONE OF FILE?
        MVI A,  $1Ø
        CMP     C               ; ..
        JNC     WTNXT2          ;JUMP IF SO
        LDA     WTNXTW          ;AFTER FIRST CLUSTER
        DCR     A               ;BACK UP ONE, SEE WHERE IT WAS
        MOV C,  A               ;GET THAT MAP BYTE
        MVI B,  Ø
        LHLD    USERDE
        DAD     B               ;OFFSET INTO USER'S FCB
        MOV A,  M               ;FINALLY, GET CLUSTER NUMBER
        STA     WTNXTV          ;STORE IT
WTNXT2: LHLD    WTNXTV          ;NOW GET DESIRED NEIGHBOR
        MOV C,  L               ;IN RIGHT AC FOR SUBR
        CALL    ALLOC           ;ALLOCATE A CLUSTER NEAR IT
        STA     WTNXTV          ;HERE'S THE ONE WE CAN HAVE
        CPI     Ø               ;OR WAS IT FULL?
        JNZ     WTNXT3          ;JUMP IF OK
        LXI H,  ANSARL          ;FULL. USER'S ANSWER CELL
        MVI M,  $Ø2             ;PUT FULL FLAG IN IT
        JMP     WTNXT4          ;GO FINISH UP

WTNXT3: LHLD    WTNXTV          ;HERE IS THE NEW CLUSTER
        MOV C,  L
        MVI E,  $Ø1             ;PUT A ONE IN BIT TABLE
        CALL    MRKBTB          ; ..
        LHLD    WTNXTV          ;SAVE AS CURRENT CLUSTER
        MVI H,  Ø               ; FULL WORD QUANTITY
        SHLD    CURRMW          ; ..
        LHLD    WTNXTW          ;HERE'S THE FCB SLOT WHICH GETS IT
        MVI H,  Ø               ;FIGURE ADDRESS FOR IT
        XCHG
LHLD    USERDE          ; IN USER'S FCB
        DAD     D
        LDA     WTNXTV          ;THE CLUSTER NUMBER AGAIN
        MOV M,  A               ;INTO THE FCB
WTNXT4: LDA     ANSARL          ;WAS THERE AN ERROR YET?
        CPI     Ø               ; ..
        JNZ     WTNXT8          ;YES, SO DON'T WRITE.
        CALL    COMPSN          ;GET SECTOR NUMBER, FROM CURRMW, CURRNR
        CALL    COMPTK          ;AND THE TRACK FROM THAT
        CALL    SETDMA          ;SET UP TO USER'S (OR STD) DMA ADDR
        CALL    WRSEC           ;WRITE A SECTOR
        CALL    RSTDMA          ;BACK TO WHAT USER SAID
        LDA     CURRNR          ;THIS RECORD
        LXI H,  CURRRC          ;UP TO COUNT IN FCB?
        CMP     M               ; ..
```

```
        LDA     CURRNR          ;YES. CURRENT PLUS 1 TO MAX
        INR     A
        STA     CURRRC          ;FOR LATER WRITING ON DISK
WINXT5: LDA     CURRNR          ;DID WE JUST FILL EXTENT?
        CPI     $7F
        JNZ     WINXT7          ;NORMALLY, NO.
        CALL    UDNRRC          ;UPDATE NR AND RC IN FCB
        MVI C,  0               ;SAY WRITING
        CALL    STPXTN          ;AND TRY FOR ANOTHER EXTENT
        LDA     ANSARL          ;WAS IT SUCCESSFUL?
        CPI     0
        JNZ     WINXT6          ;JUMP IF SO
        LXI H,  CURRNR          ;PUT A BAD NR, FOR NEXT WRITE.
        MVI M,  $FF             ;WILL GET EOF IF TRIES AGAIN.
 NXT6:  LXI H,  ANSARL          ;SAY THIS WRITE WAS OK, THOUGH.
        MVI M,  0
WINXT7: CALL    UDNRRC          ;UPDATE NR AND RC IN FCB
WINXT8: RET

LOG IN THE DISK IN CURDSK

LOGCUR: MVI A,  $03             ;RANGE CHECK THE DRIVE NUMBER
        LXI H,  CURDSK          ;CHECK USER'S ARG
        CMP     M               ;LEGAL?
        JNC     LOGCU1          ;JUMP IF OK
        LXI H,  LOGCU1                                  ; 3A21  21 29 3A
        PUSH    H                                       ; 3A24  E5
        LHLD    VSELER          ;HANDLE BAD UNIT
        PCHL                                            ; 3A28  E9

LOGCU1: LDA     CURDSK          ;DISK NUMBER
        ADD     A               ;TIMES 32 BYTES PER DISK
        ADD     A
        ADD     A
        ADD     A
        ADD     A
        MOV C,  A               ;MAKE ADDRESS
        MVI B,  0
        LXI H,  ALLOCS          ;BASE OF ALLOC AREA
        DAD     B               ;FOR THIS DISK
        SHLD    CURALV          ;CURRENT DISK'S ALLOC VECTOR
        LDA     CURDSK          ;DISK NUMBER * 16
        ADD     A
        ADD     A
        ADD     A
        ADD     A
        MOV C,  A               ;AS AN ADDRESS
        MVI B,  0
        LXI H,  CKSTAB          ;SPACE FOR CHECKSUMS
        DAD     B               ;BASE FOR CKSM'S OF THIS DISK'S DIR
        SHLD    VECCKS          ;POINTER TO CKSUM TABLE
        LHLD    CURDSK          ;COMPUTE POSITION IN TRK POINTERS TABLE
        MVI H,  0               ;FOR THIS DISK
        LXI B,  TKPTRS          ; ..
        DAD     B               ; ..
        SHLD    TKPNTR          ;DEFINE WHERE TRACK NUMBER LIVES
        LHLD    CURDSK          ;FOR CURRENT DISK
        MVI H,  0               ;SIMILARLY FOR SECTOR 0 ON THAT TK
        LXI B,  TSEC0V          ;VECTOR OF SECTOR 0 ON TK
                                ;DOUBLE THE DISK NUMBER, TO MAKE WORD
```

```
        DAD     B               ;IN THIS TABLE
        SHLD    TKSEC0
        LHLD    CURDSK          ;NOW LET I/O PACKAGE KNOW IT
        MOV C,  L
        CALL    CBSELE          ;BIOS - SELECT A DISK. THIS IS
; THE ONLY SUCH CALL
        LDA     LOGVEC          ;CURRENTLY LOGGED DISKS
        RLC                     ;SEE IF IT WAS KNOWN ALREADY
        PUSH    PSW
        LDA     CURDSK          ; ..
        INR     A               ;COMPUTE UP BIT NUMBER FOR DISK
        MOV C,  A               ;SAVE NUMBER
        POP     PSW             ;GET BACK BITS
        CALL    RALN            ;DO (C) RAL'S
        RAR                     ;CHECK THE BIT
        JC      LOGCU2          ;JUMP IF ALREADY KNOWN
        LHLD    LOGVEC          ;CURRENTLY LOGGED DISKS
        MOV C,  L               ;TO RIGHT AC
        CALL    ORCURR          ;OR IN BIT FOR CURRENT DISK
        STA     LOGVEC          ;ADD TO KNOWN DISKS
        CALL    ROALOC          ;SET UP BIT TABLE FROM ALL DIR FCB'S
LOGCU2: RET

;LOG IN AND SELECT A DISK

LGNDSK: LXI H,  CURDSK
        LDA     ANSARG          ;NEWLY REQUESTED DISK
        CMP     M               ;SAME DISK AS BEFORE?
        JZ      LGNDS1          ;YES.
        LDA     ANSARG          ;NO, SO CHANGE TO IT
        STA     CURDSK          ;UPDATE
        CALL    LOGCUR          ;AND LOG IT IN
LGNDS1: RET

SETDSK: LHLD    USERDE          ;GET FCB ADDR
        MVI A,  $1F             ;LOW 5 BITS FOR DRIVE
        ANA     M               ;"ENTRY TYPE" FIELD. USUALLY 0.
        DCR     A               ;CONVERT TO 0-3
        STA     ANSARG          ;DISK USER HAS IN FCB
        CPI     $1E             ;DID USER SPECIFY ONE?
        JNC     SETDS1          ;NORMALLY ZERO, THAT'S ALL.
        LDA     CURDSK          ;SAVE CURRENT LOGGED DISK
        STA     RELOGN          ;MRETN WILL RE-LOG FROM HERE
        LHLD    USERDE
        MOV A,  M               ;PICK UP FCB'S ENTRY TYPE
        STA     RELOGF          ;FLAG NEED TO RE-LOG
        LHLD    USERDE          ;NOW CLEAR WHAT USER SAID
        MVI A,  $E0
        ANA     M
        MOV M,  A
        CALL    LGNDSK          ;LOG IN AND SELECT REQUESTED DISK
SETDS1: RET

;SET DMA FROM HERE. ONE CALL, SETS TO DEFBFR

DEFDMA: LHLD    USERDE          ;GET SYSTEM'S DMA ADDR ARG
        SHLD    SYSDMA          ;SAVE IN THIS SEMI-CONSTANT
        MOV B,  H               ;SWITCH REGISTERS FOR BIOS
        MOV C,  L
        CALL    CBSTMA          ;VECTOR SET DMA (BIOS)
```

```
        RET

;WORK ROUTINE FOR USER'S CALLS, FROM BDOSH, FROM 5

BDOS00: LHLD    USERDE
        MOV A,  L               ;USER'S "E"
        STA     ANSARG          ;SAVE IT
        LXI H,  0
        SHLD    ANSWER          ;DEFAULT ANSWER IS "0,,0"
        MOV A,  L                                       ; 3AE0  7D
        STA     ANSARL          ;0 TO "A" ANSWER (SEE T353)
        STA     RELOGF          ;ASSUME NO NEED TO RE-LOG
        LHLD    USRFCN          ;GET USER'S FUNCTION CODE
        MOV C,  L               ; ..
        MVI B,  0               ;AS A FULL ADDRESS
        LXI H,  FCNDTB          ;COMPUTE DISP TABLE ADDRESS
        DAD     B               ; ..
        DAD     B               ; ..
        MOV E,  M               ;PICK UP THE ROUTINE ADDR
        INX     H
        MOV D,  M               ;TWO BYTES
        XCHG                    ;AND GO TO IT.
        PCHL                    ; ..

;HERE ARE THE FUNCTION HANDLERS, FCN0 THRU FCN30


FCN0:   CALL    CBWBUT          ;DO A WARM RE-BOOT, IN BIOS
        JMP     MRETN

FCN1:   CALL    RDCONS          ;READ CONSOLE CHAR
        STA     ANSARL          ;SAVE ANSWER
        JMP     MRETN

FCN2:   LHLD    ANSARG          ;TYO ROUTINE
        MOV C,  L               ;USER'S D
        CALL    CTYO2           ;TYPE, WITH TAB SIMULATION
        JMP     MRETN

FCN3:   CALL    CBPTRI          ;READ FROM "READER", BIOS
        STA     ANSARL          ;SAVE ANSWER
        JMP     MRETN

FCN4:   LHLD    ANSARG          ;WRITE ON "PUNCH"
        MOV C,  L               ;CHAR TO WRITE
        CALL    CBPTPO          ;IN BIOS
        JMP     MRETN

FCN5:   LHLD    ANSARG          ;WRITE ON "LISTING"
        MOV C,  L               ;CHAR TO LIST
        CALL    CBLPTO          ;IN BIOS
        JMP     MRETN

FCN6:   LHLD    .BDOS.+1        ;GET ADDRESS OF BDOS
        SHLD    ANSWER          ;FOR USER
        JMP     MRETN

FCN7:   LHLD    $0003           ;GET I/O STATUS BYTE
        MVI H,  0               ;ONLY THE "L" PART
        SHLD    ANSWER          ;SAVE AS ANSWER
```

```
        JMP     MRETN

FCN8:   LDA     ANSARG          ;SET I/O STATUS BYTE
        STA     $0003           ;EASY?
        JMP     MRETN

FCN9:   LHLD    USERDE          ;PRINT STRING, TERM BY "$"
        MOV B,  H               ;SWAP AC'S
        MOV C,  L
        CALL    DLSOUT          ;WE HAVE A SUBROUTINE FOR THAT
        JMP     MRETN

FCN10:  CALL    PSIN            ;READ STRING, BUFFERED (PSIN)
        JMP     MRETN

FCN11:  CALL    CHKKBD          ;CHECK KEYBOARD FLAG
        STA     ANSARL          ;SAVE ANSWER
        JMP     MRETN

FCN12:  JMP     MRETN           ;LIFT HEAD (NOP)

FCN13:  LXI H,  CURDSK          ;INITIALIZE BDOS
        MVI M,  0                                       ; 3B6A  36 00
        LXI H,  LOGVEC                                  ; 3B6C  21 C1 3D
        MVI M,  0                                       ; 3B6F  36 00
        LXI H,  FRZDMA          ;SAY ALLOWED TO MOVE DMA AROUND
        MVI M,  0               ; ..
        LXI H,  $0080           ;DEFAULT DMA ADDR
        SHLD    USERDE                                  ; 3B79  22 0A 33
        CALL    DEFDMA                                  ; 3B7C  CD C7 3A
        CALL    LOGCUR                                  ; 3B7F  CD 18 3A
        JMP     MRETN

FCN14:  CALL    LGNDSK          ;SELECT AND LOG IN DISK
        JMP     MRETN

FCN15:  CALL    SETDSK          ;OPEN FILE
        CALL    OPENFI
        JMP     MRETN

FCN16:  CALL    SETDSK          ;CLOSE FILE
        CALL    CLOSFI                                  ; 3B97  CD 24 38
        JMP     MRETN

FCN17:  CALL    SETDSK          ;SEARCH FOR FILENAME
        MVI C,  $0D             ;CHECK 13 CHARS
        CALL    LOOKUP
        JMP     MRETN

FCN18:  LHLD    STPFIW          ;SEARCH FOR NEXT OCCURENCE OF NAME
        SHLD    USERDE          ;USERS FCB FROM PREV LOOKUP
        CALL    SETDSK          ;PROCESS ENTRY TYPE FOR DRIVE
        CALL    STPFIL          ;GET NEXT MATCHING FILE
        JMP     MRETN           ;ANSWER IS IN ANSARL

FCN19:  CALL    SETDSK          ;DELETE FILE
        CALL    DODEL                                   ; 3BBA  CD E7 36
        JMP     MRETN

FCN20:  CALL    SETDSK          ;READ NEXT RECORD
```

```
        CALL    RDNXTR                                  ; 3BC3  CD F6 38
        JMP     MRETN

FCN21:  CALL    SETDSK          ;WRITE NEXT RECORD
        CALL    WTNXTR                                  ; 3BCC  CD 4A 39
        JMP     MRETN

FCN22:  CALL    SETDSK          ;MAKE FILE
        CALL    MAKEFI                                  ; 3BD5  CD 42 38
        JMP     MRETN

FCN23:  CALL    SETDSK          ;RENAME FILE
        CALL    DOREN                                   ; 3BDE  CD B9 37
        JMP     MRETN

FCN24:  LDA     LOGVEC          ;INTERROGATE LOGIN VECTOR
        STA     ANSARL          ;WHICH IS JUST THESE BITS
        JMP     MRETN

FCN25:  LDA     CURDSK          ;GET CURRENT DRIVE NUMBER
        STA     ANSARL          ;SAVE AS ANSWER
        JMP     MRETN

FCN26:  LDA     FRZDMA          ;SET DMA ADDR. ALLOWED TO?
        RAR
        JNC     FCN26A          ;IF NOT, SKIP THIS.
        LHLD    USERDE
        SHLD    USRDMA          ;REMEMBER WHAT USER WANTED
        JMP     FCN26B                                  ; 3C03  C3 09 3C

FCN26A: CALL    DEFDMA          ;MAKE IT SYSTEM STANDARD
FCN26B: JMP     MRETN

FCN27:  LHLD    CURALV          ;GET CURRENT DRIVE'S ALLOCATION VECTOR
        SHLD    ANSWER                                  ; 3C0F  22 0C 33
        JMP     MRETN

FCN28:  CALL    MAKERO          ;UNDOCUMENTED. SET R/O BIT FOR LGD DISK.
        JMP     MRETN

FCN29:  LDA     ROVEC           ;UNDOCUMENTED. GET R/O VECTOR
        STA     ANSARL
        JMP     MRETN

;FREEZE DMA ADDR. DON'T LET USER CHANGE IT.

FCN30:  LXI H,  FRZDMA          ;UNDOCUMENTED
        MVI M,  $01                                     ; 3C27  3e 01
        CALL    DEFDMA          ;AND FREEZE IT AT SYS STANDARD
        JMP     MRETN

;DISPATCH TABLE FOR USER FUNCTIONS.

FCNDTB: .ADDR   FCN0,FCN1,FCN2,FCN3,FCN4,FCN5
        .ADDR   FCN6,FCN7,FCN8,FCN9,FCN10,FCN11
        .ADDR   FCN12,FCN13,FCN14,FCN15,FCN16,FCN17
        .ADDR   FCN18,FCN19,FCN20,FCN21,FCN22,FCN23
        .ADDR   FCN24,FCN25,FCN26,FCN27,FCN28,FCN29
        .ADDR   FCN30
```

```
;COMMON RETURN TO USER

MRETN:  LDA     RELOGF          ;NEED TO RE-LOG A DISK?
        CPI     0
        JZ      MRETN1          ;JUMP IF NOT
        LHLD    USERDE          ;YES. POINT TO FCB
        LDA     RELOGF          ;PUT ENTRY TYPE BYTE BACK IN USER FCB
        MOV M,  A
        LDA     RELOGN          ;HAVE TO RE-LOG THIS DISK
        STA     ANSARG          ;LOG FROM HERE
        CALL    LGNDSK          ;GO DO IT
MRETN1: LDA     ANSARL          ;ANSWER TO USER
        LXI D,  ANSWER                          ; 3C88  11 0C 33
        CALL    AORIDE          ; 0A IOR (DE) => HL
        XCHG                                    ; 3C8E  EB
        DCX     H                               ; 3C8F  2B
        MOV M,  E                               ; 3C90  73
        INX     H                               ; 3C91  23
        MOV M,  D                               ; 3C92  72
        RET                                     ; 3C93  C9

        XCHG                                    ; 3C94  EB
APLSMM: MOV E,  A               ; 0A + (HL) => HL
        MVI D,  0                               ; 3C96  16 00
        XCHG                                    ; 3C98  EB
        LDAX    D                               ; 3C99  1A
        ADD     L                               ; 3C9A  85
        MOV L,  A                               ; 3C9B  6F
        INX     D                               ; 3C9C  13
        LDAX    D                               ; 3C9D  1A
        ADC     H                               ; 3C9E  8C
        MOV H,  A                               ; 3C9F  67
        RET                                     ; 3CA0  C9

AORHL:  MOV E,  A               ; 0A IOR HL => HL
        MVI D,  0                               ; 3CA2  16 00
        MOV A,  E                               ; 3CA4  7B
        ORA     L                               ; 3CA5  B5
        MOV L,  A                               ; 3CA6  6F
        MOV A,  D                               ; 3CA7  7A
        ORA     H                               ; 3CA8  B4
        MOV H,  A                               ; 3CA9  67
        RET                                     ; 3CAA  C9

AORIDE: XCHG                    ; 0A IOR (DE) => HL
        MOV E,  A                               ; 3CAC  5F
        MVI D,  0                               ; 3CAD  16 00
        XCHG                                    ; 3CAF  EB
        LDAX    D                               ; 3CB0  1A
        ORA     L                               ; 3CB1  B5
        MOV L,  A                               ; 3CB2  6F
        INX     D                               ; 3CB3  13
        LDAX    D                               ; 3CB4  1A
        ORA     H                               ; 3CB5  B4
        MOV H,  A                               ; 3CB6  67
        RET                                     ; 3CB7  C9

;MULTIPLE SHIFT ROUTINE, DOING RLC'S

RLCNM:  MOV A,  M                               ; 3CB8  7E
```

```
RLCN:   RLC                                             ; 3CB9  07
        DCR     C                                       ; 3CBA  0D
        JNZ     RLCN                                    ; 3CBB  C2 B9 3C
        RET                                             ; 3CBE  C9

RALNM:  MOV A,  M                                       ; 3CBF  7E
RALN:   RAL                                             ; 3CC0  0F
        DCR     C                                       ; 3CC1  0D
        JNZ     RALN                                    ; 3CC2  C2 C0 3C
        RET                                             ; 3CC5  C9

;PICK UP 16 BITS AT (HL), THEN DO (C) DAD H'S ON THEM

SHLMM:  MOV E,  M               ;PICK UP A DOUBLE BYTE
        INX     H
        MOV D,  M
        XCHG                    ;PUT THEM IN HL
SHLMML: DAD     H               ;SHIFT THEM LEFT
        DCR     C
        JNZ     SHLMML          ;(C) TIMES, AT LEAST ONCE
        RET

RARNM:  MOV A,  M               ;PICK UP MEM ARG
RARNL:  ORA     A               ;CLEAR CARRY
        RAR                     ;ROTATE IT RIGHT
        DCR     C               ;THIS MANY TIMES
        JNZ     RARNL
        RET

MOV L,  C                                               ; 3CD8  69
        MOV H,  B                                       ; 3CD9  60
DEMIHL: MOV C,  M               ;16 BIT SUBTRACT 2 MEMS TO HL
        INX     H                                       ; 3CDB  23
        MOV B,  M                                       ; 3CDC  46
        LDAX    D               ;(DE) - (HL) => HL
        SUB     C                                       ; 3CDE  91
        MOV L,  A                                       ; 3CDF  6F
        INX     D                                       ; 3CE0  13
        LDAX    D                                       ; 3CE1  1A
        SBB     B                                       ; 3CE2  98
        MOV H,  A                                       ; 3CE3  67
        RET                                             ; 3CE4  C9

DEMIZA: MOV L,  A               ; 0A => HL
        MVI H,  0                                       ; 3CE6  26 00
JDEMHL: LDAX    D               ; (DE) - HL => HL
        SUB     L                                       ; 3CE9  95
        MOV L,  A                                       ; 3CEA  6F
        INX     D                                       ; 3CEB  13
        LDAX    D                                       ; 3CEC  1A
        SBB     H                                       ; 3CED  9C
        MOV H,  A                                       ; 3CEE  67
        RET                                             ; 3CEF  C9
^L
;VARIABLES FOR BDOS PORTION

ANSARL: .BYTE 0         ;RETURN USER'S VALUE HERE, FOR A REG
ANSARG: .BYTE 0                                         ; 3CF1  00
CHROFS: .BYTE 0         ;CHARACTER OFFSET OF FCB IN DIR BUFFER
DOSFCB: .BYTE 0                                         ; 3CF3  00
```

```
DIRSEC: .BYTE 0                                          ; 3CF4  00
SYSDMA: .ADDR DEFBFR            ;OUR COPY OF CURRENT DMA ADDR
USRDMA: .ADDR DEFBFR            ;USER'S REQUESTED DMA ADDR
FRZDMA: .BYTE 0
ALLOCS: .BLKB NDISKS*NALLOV     ;ALL THE ALLOCATION VECTORS
CURALV: .BLKW 1                 ;CURRENT DISK'S ALLOC VECTOR
ROVEC:  .BYTE 0                 ;BITS FOR DRIVES THAT ARE READ-ONLY
CKSTAB: .BLKB $40               ;NDISKS * NDIRECTORY-SEGMENTS
VECCKS: .BLKW 1                 ;POINTER TO CHECKSUM TABLE
RELOGN: .BYTE 0                 ;MRETN MUST RE-LOG THIS DISK
RELOGF: .BYTE 0                 ;FLAG NEED TO RE-LOG AT MRETN
LOGVEC: .BYTE 0         ;BITS FOR CURRENTLY LOGGED IN DISKS
                        ; B0=A, B1=B, B2=C, B3=D
TKPTRS: .BLKB NDISKS    ;CURRENT TRACK, FOR EACH DISK
TSECOV: .BLKW NDISKS    ;FOR ALLOCATING TRACKS
TKPNTR: .BLKW 1         ;POINTER TO TRACK NUMBER FOR CURRENT OP
TKSECO: .BLKW 1         ;POINTER INTO TRACKV, FOR ALLOC TRACK
CURRRC: .BYTE 0         ;RECORD COUNT
CURRNR: .BYTE 0         ;RECORD NUMBER
CURRMP: .BYTE 0,0       ;CURRENT MAP INFO, W/ 8 HIGH ORDER 0'S

;BELOW HERE ARE LOCAL SUBR ARG STORAGE

CTKTMP: .BLKW 1         ;TEMP IN ALLOCATION COMPUTATION
RWARG:  .BYTE 0
RWCOMT: .BYTE 0
CKSTM1: .BYTE 0
CKSTM2: .BYTE 0
CRCURT: .BYTE 0
VERDIT: .BYTE 0
MAPNXV: .BYTE 0
GETBTT: .BYTE 0
MRKBTV: .BYTE 0
MRKBTW: .BYTE 0
ALOFCV: .BYTE 0
ALOFCW: .BYTE 0
ALOFCX: .BYTE 0
RDALCV: .BYTE 0
STPFIV: .BYTE 0
STPFIW: .BYTE 0,0
STPFIX: .BYTE 0
STPFIY: .BYTE 0
LOOKUV: .BYTE 0,0,0,0
ALO.LO: .BYTE 0
ALO.HI: .BYTE 0
CPYNAV: .BYTE 0
CPYNAW: .BYTE 0
CPNFIV: .BYTE 0
MAKFIV: .BYTE 0
MAKFIW: .BYTE 0,0
STPXTV: .BYTE 0
WINXTV: .BYTE 0
WINXTW: .BYTE 0

;A LITTLE SPACE TO END OF PAGE
        .BYTE 0,0,0,0,0,0
.L
;
;HERE STARTS THE CUSTOMIZED BIOS
;
```

```
;FIRST THE VECTOR FOR DEFINED ENTRIES

BIOS:   JMP     0               ;COLD BOOT
CBWBUT: JMP     0               ;WARM BOOT
CBTTCK: JMP     0               ;CONSOLE STATUS CHECK
CBCTYI: JMP     0               ;CONSOLE INPUT
CBCTYO: JMP     0               ;CONSOLE OUTPUT
CBLPTO: JMP     0               ;LIST OUTPUT
CBPTPO: JMP     0               ;PUNCH OUTPUT
CBPTRI: JMP     0               ;READER INPUT
CBHOME: JMP     0               ;HOME THE SELECTED DISK
CBSTLE: JMP     0               ;SELECT A DISK
CBSTTK: JMP     0               ;SET TRACK
CBSTSC: JMP     0               ;SET SECTOR
CBSTMA: JMP     0               ;SET DMA ADDRESS
CBREAD: JMP     0               ;READ A SECTOR
CBWRIT: JMP     0               ;WRITE A SECTOR

;REST NOT IN THIS SOURCE FILE

        .END
^L
*
```