```
        TITLE   'AA1.ASM OMNIDISK FIRMWARE'
;       SOURCE IS AA1.ASM
;KK     NOTE: FOR CHANGES DONE BY DAVE KOSKI SEARCH FOR: KK
;
;*************************************************************************
;                                                                        *
;               DMA DISK CONTROLLER FIRMWARE                             *
;                                                                        *
;       COPYRIGHT (c) 1983 W/W COMPONENT SUPPLY                          *
;                                                                        *
;       MACHINE READABLE SOURCE CODE IS AVAIABLE TO QUALIFIED            *
;       CUSTOMERS.  CALL US FOR MORE INFO.                               *
;                                                                        *
;       WARNING: FEDERAL COPYRIGHT LAW PROHIBITS UNAUTHORIZED            *
;       DISTRIBUTION.                                                    *
;                                                                        *
;*************************************************************************
; STICKY LABEL: 97   TIME: 0800   DATE: 12 MAR 85
;
;-------FIX--HISTORY-----------------------------
;
;
; 24 NOV (Turkey day) 1983
;       ADD 16 MS DELAY AFTER FLOPPY DISK 8<==>5 SWAP
;       SHOULD NOW DO SINGLE/DOUBLE ON 8" W/O READY CHANGE
;         (MOVED NECINT2..NECINT5 TO RPI. NAMES: RPINTx)
;
; 26 NOV
;       CLEAN UP RPINTx CODE
;
; 03 DEC
;       RPI NOW CHECKS RES$C FOR CORRECT CYILNDER W/ RETRY IF BAD
;
; 23 DEC
;       DEBUG CODE TO TEST WHO-IS-TO-BE-THE-A:-DISK SWITCH SETTING
;
; 25 DEC (Merry Christmas)
;       REMOVE 23 DEC DEBUG CODE
;       DOCUMENT SWITCH SETTING
;       REMOVE DEBUG DELAYS FOR 5" FLOPPY
;       USED RRC/JNC INSTED OF ANI/JZ IN GETHOST
;       PULLED ALL DEBUG CALLS
;       SET RAM SIZE TO 12K
;
; 26 DEC
;       (25 DEC SYSTEM DID NOT WORK)
;       INITSPEC USED INSTED OF HEX CONSTANT FOR STEP RATE SETTING IN INIT
;       PUT RAM BACK TO 16K -- AND BACK TO 12K
;       RESTORE ANI/JZ TO PRE-CHRISTMAS USE -- AND BACK TO CHRISTMAS
;       RESTORE PBUFFER: AND DELETE
;       RESTORE EXTRA DELAYS
;
; 27 DEC
;       PULL DELAYS IN NECSERV AND WNB2
;
; 28 DEC
;       SET RAM SIZE TO 10K
;
; 03 JAN 84
;       HAPPY NEW YEAR
;       ONLY A: DEFINED AT POWER ON, SWITCH SETTING DEFINES DISK
;
; 05 JAN
;       FIX 03 JAN CODE.  FORCE UNDEFINE ALL BUT A: DISK.
;
; 06 JAN
;       REMOVE DROP-DEAD-ON-ERROR SUBROUTINE
;       CHANGE AND DOCUMENT SEVERAL UNIQUE ERROR CODES
;       REMOVE SEEK-TO-INNER FROM RWRETRY ETC
;
; 11 JAN 84
;       ADDED HARD DISK STUFF BCW
;
; 29 FEB (leap day)
;       SPLIT INTO 25 SECTIONS
;       MADE ALL THINGS NEW UNDER THE GETDPB COMMAND
;       NEW GETDPB: FIXED BAD JMP THAT STOPPED SD DISKETTS FROM WORKING
;
; 01 MAR
;       NEW GETDPB: FIXED WRONG REGISTER IN LOCDPB FDRELTAB SEARCH
;       NEW GETDPB: ADDED MISSING CODE TO LOCDPB
;
; 02 MAR
;       NEW GETDPB: YET ANOTHER FIX IN THE NEW FDREL SEARCH
;       RESTORED PART OF FMT MESSED UP IN DOING NEW GETDPB
;
; 05 MAR
;       STILL 5" SYSWRT PROBLEM
;       ADDED RE-ORGANIZED HARD DISK CODE WITH 3 DIVIDE ROUTINES
;       REMOVE ALL "PRIORITY BUFFER" STUFF
;
; 09 MAR
;       REMOVED "PRIORITY BUFFER" LENGTH FROM MAKECUR
;       ADDED ZAP OF "WDDEV" TO DEV CHANGE IN MAKECUR (BOOT PROBLEM)
;
; 11 MAR
;       REMOVE SOME DEBUG CODE
;       PULL GARBAGE (??) FROM HDINIT
;
; 16 MAR
;       FIX HD$DPB ALLOC COUNT
;       START TO MAKE VERIFY ONLY ON HARD DISK WORK
;       MAKE HD FORMAT WRITE E5'S TO ALL SECTORS
;
; 17 MAR (Have a nice St. Pat's day.)
;       HDINIT NOW USES (OPTIONAL) RECAL PLUS SEEK [NOT RECAL + RECAL]
;
; 18 MAR
;       FIX MORE OF HDINIT
;
; 19 MAR
;       AGAIN REMOVE UNCONDITIONAL HEAD LOAD AND RECAL AFTER RESET
;       ** DEBUG VERSION: SA712 HAS ONLY 3 HEADS
;
; 20 MAR
;       MADE WRITE AN INTERNAL SUBROUTINE TO HELP SYSWRT WHEN NO
;         VALID ID CAN BE READ FROM FLOPPY DISK TRACK
;
; 22 MAR
;       ADDED SEVERAL NEW FD FORMATS IN FDREL
;
; 25 MAR
;       PUT SA712 BACK TO 4-HEAD DEVICE
;
; 27 MAR
;       READID ON SIDE 2 OF 5.25" FLOPPIES FOR DOUBLE-SIDED CHECK
;       REMOVE "SETHEAD" USE FROM "GETDPB" FUNCTION
```

```
;
; 29 MAR
;       BOOT/SYSWRT TO USE (DPB) LOGICAL SECTORS/LOGICAL TRACK
;       REMOVED CURLDT AREA
;       ZAPPED BUFSIDE IN MAKECUR TO KEEP FWT FROM ACTING UP
;
; 30 MAR
;       5.25" SIDE 2 READID ERROR NO LONGER LEFT IN GENSTAT
;
; 01 APR (tax time approaches...no foolen)
;       CHANGED 5.25" STEP RATE TO 16 MS FOR DEBUG
;       FIXED 8/5 STEP RATE SETUP NEAR MAKEC1A
;
; 05 APR
;       ZEROED CPS$SIDE BEFORE LOCDPB CALL IN BOOT/SYSWRT
;       5.25" STEP RATE BACK TO 6MS
;
; 06 APR
;       FIXED SETSKEW FUNCTION (SKEW BYTE ADDRESS WAS MESSED UP)
;
; 10 APR (property tax day)
;       ADDED GET ROUTINES FOR EXTENDED DPB AND MS-DOS BPB
;       EXTENDED COMMAND TABLE FOR NEW OPCODES
;       CHANGED END-OF-TABLE EQUATE
;       MOVED FF OPCODE INTO COMAND TABLE
;
; 12 APR
;       ADDING EXTENDED DPB AND BPB STUFF...
;       AT THIS POINT WE ARE USING 128-BYTE LOGICAL SECTORS
;
; 14 APR (At last my federal tax refund came -- just in time to pay state tax)
;       ADDING CODE FOR PHYSICAL FORMAT I/O
;       MAX LENGTH FOR HARD DISK IS 2000 (NOT 2048) ALLOC UNITS
;
; 15 APR
;       PHYSICAL MODE I/O DONE...NOW TO DEBUG
;
; 16 APR
;       FIXED STUPID BUG IN INBUFREAD/WRITE -- NEED "0080H" NOT JUST "0080"
;
; 17 APR
;       FIXES IN MS-DOS BPB GENERATOR
;       FIXED WRONG SENSE OF RTN ON PFIOFLAG IN HARD DISK BUFFER ADDRESS
;       FIXED BAD LENGTH IN 200H & 400H SECTOR SIZE
;
; 18 APR
;       FIXED BAD LENGTH IN FLOPPY DISK GETDPB
;       FIX BAD LENGTH IN PHYS FORMAT WRITE
;
; 19 APR
;       MORE LITTLE CHANGES IN INBUF READ/WRITE LENGTH USAGE
;       MULTI-LOGICAL-SECTOR DMA XFERS JUST MIGHT NOW WORK
;
; 20 APR
;       ADDED 5" DD 512-BYTE 9 SECTOR FORMAT
;       PULLED DD 128 FORMATS
;       ADJUSTED 5" DD 1024-BYTE 5 SECTOR FORMAT
;
; 23 APR
;       FIXED HANG IN PHYS MODE WRITE
;
; 25 APR
;       CHANGED GET MEDIA CHANGE STATUS STUB TO REAL ROUTINE
;       MEDIA CHANGE STAT PUT INTO GETDPB, NECSERV AND UNLOAD
;
; 26 APR
;       MEDIA CHANGE STATUS PUT INTO 8<==>5 SWAP
;       HARD DISK DPB EXTENDED FOR MS-DOS ETC
;       MASK ST3 BEFORE USING TO MAKE DEVICE TABLE ADDRESS
;
; 27 APR
;       PUT A BUNCH OF CHANGES INTO 'MOST ALL MY DMA CODE
;
; 30 APR
;       FIX LOCAL DMA ADDRESS IN MEMORY DISK READ
;
; 02 MAY
;       CORRECTED DPB EXTENSION FOR CP/M+
;
; 03 MAY
;       ADDED # SIDES CHECK IN FORMAT
;       RECAST (FLOPPY DISK) FORMAT AND VERIFY INTO SEPARATE SUBROUTINES
;
; 04 MAY
;       BUG FIXES IN CHANGED FORMAT STUFF
;
; 09 MAY
;       MOVED CODE FROM INITALL TO INIT TO STOP FWT AFTER RESET
;
; 11 MAY
;       DEBUG TURNED ON TO FIND STRANGE HANG
;
; 12 MAY
;       MS-DOS SS 5" FD TO ONE PHYSICAL SECTOR PER ALLOCATION UNIT
;
; 13 MAY
;       DEBUG TURNED OFF (FMT BUG KILLED ONLY DISKETTE GIVING STRANGE HANG)
;       FORMAT CODE FOR NUMBER-OF-SIDES CHECK NOW IN PLACE
;       FORMAT CODE FOR KILL BACK SIDE OF DISKETTE NOW IN PLACE
;
; 14 MAY
;       HARD DISK BPB FIXED ('TWAZ OFF BY 4 BYTES)
;       SEEK TO GIVEN TRACK BEFORE DOING NUMBER-OF-SIDES CHECK USING WRITE
;
; 21 MAY
;       BPB FIX IN 8" 8 1024-BYTE SINGLE DENSITY
;       DEFINE "SET FLOPPY DISK PARAMETERS" COMMAND
;       VARIOUS CHANGES TO USE VALUES SET IN ABLVE NEW COMMAND
;       INIT PART OF RAM BY A LIST COPY
;
; 23 MAY
;       DEFINE MS-DOS FAT SIZE FOR FLOPPY USING MATH INSIDE MACRO
;       IMPLEMEMT HEAD SETTLE DELAY IN WAITNBUSY
;       PUT REPLY TO SETSKEW COMMAND IN GENERAL STATUS
;
; 29 MAY
;       POWER-ON BUG FIXED (READ AND TRASH DONE TWICE)
;
; 04 JUN
;       FIX FOR 8" GET-MEDIA-CHANGE STATUS
;
; 05 JUN
;       FIX FOR ALL FLOPPY GET-MEDIA-CHANGE STATUS
;
; 24 JUN
;       FOR MS-DOS 8"DS 1024-BYTE SECTORS: 1 SECTOR = 1 ALLOCATION UNIT
;
; 25 JUN
```

```
;       BETTER CHECK FOR READY CHANGE IN NECSERV
;
; 26 JUN
;       FIX STUPID BUG IN PHYSICAL (OPS 1E/1F) HARD DISK READ/WRITE
;
; 27 JUN
;       NEW SUBROUTINE OF NECSERV TO TAKE ALL INTERRUPT STAT BEFORE
;         DOING ANYTHING ELSE (RECURSIVE SUBROUTINE)
;
; 19 JUL
;       PART WAY TO COMMON HARD DISK STUFF:
;         NUMBER OF HEADS IS IN MEMORY
;         (HARD DISK) CONTROLLER **ECC/NO-ECC** TYPE IS IN MEMORY
;
; 20 JUL
;       BUG FIXES FROM 19 JUL:
;         BAD VALUE FOR CTLR TYPE
;         NO "RET" IN DIVIDE SUBROUTINE
;
; 26 JUL
;       MORE COMMON HARD DISK STUFF:
;         STEP RATE NOW IN MEMORY BYTE
;         HAVE HARD DISK DEVICE TABLE ENTRY FOR OS TABLE (DPB/BPB/WHATEVER)
;
; 27 JUL
;       MOVED STEP RATE SETTER TO ABOVE HDINIT TO ALSO SET STEP RATE
;
; 28 JUL
;       ADDED PRECOMP CYL TO MEMORY PARAMETER
;       GETDPB ETC NOW USE OS BLOCK IN HARD DISK DEVICE TABLE
;       FIXED BAD REGISTER IN GETDPB ETC FROM HARD DISK
;
; 29 JUL
;       HD BOOT SHOULD NOW WORK--SECTORS/TRK NO LONGER FROM DPB
;       HD I/O CODE NOW USES DEV SEL AND HARDWARE OFFSET
;
; 31 JUL
;       CHANGED GET OF MEMORY DISK PARAMETER STUFF
;         STRUCTURE NOW: INIT FLAG, FOUR LEN/LOC PAIRS, LEN OS BLOCK, OS BLOCK
;         (PRIOR STRUCTURE HAD INIT FLAG LAST AND ONLY DPB ALLOWED)
;
; 01 AUG
;       CHANGED TO RETURN MEM DISK OS BLOCK DEFINED YESTERDAY
;
; 02 AUG
;       BOOT/SYSWRT CODE TO SWAP CTLR TYPE ON CERTAIN HARD DISK ERRORS
;       "JMP BOOTSUB" NOW SETUP IN INITALL LIST
;       INSTALL DEBUG CODE SO I CAN SEE HARD DISK ERROR CODE IN BOOT
;         (Brent is so very busy this year)
;
; 03 AUG
;       INSTALL MORE DEBUG CODE (SEE 02 AUG)
;       TRY EC=40 (BUT CHECK IF FLOPPY ERROR CODE)
;       ITS OK W/ ECC CONTROLLER.  FLOPPY DOES NOT MAKE EC=40.
;
; 04 AUG
;       PULL DEBUG CODE IN BOOT/SYSWRT
;       SKEW FACTOR FROM SEEKSEC USED IN HARD DISK FORMAT
;
; 05 AUG
;       MOVE ZERO OF SEEKSEC OUT OF EARLY PART OF FMTTRK
;
; 10 SEP
;       FIX PHYSICAL SECTOR READ/WRITES FOR HARD DISK, MOSLTLY WRITES
;       ADD FAST READ/WRITES FOR PHYSICAL NON DMA HARD DISK
;
; 06 NOV (election day)
;       CORRECT MEDIA BYTES FOR IBM-PC COMPATABILITY; MAKE ALL UNIQUE
;       PULL SOME OLD COMMENTED-OUT HARD DISK STUFF
;
; 20 NOV
;       TRIM .75 USEC/BYTE OFF NON-DMA FULL SECTOR HARD DISK READ
;       RESTORE HARD DISK FWT WITH HD FULL SECOTR NON-DMA READ/WRITE
;
; 26 NOV
;       PUT BRENTS HEX PATCHES OF FDRELTAB INTO SOURCE
;       ADD IGNORE FD ERROR OPCODE
;
;       *** 1985 ***
;
; 16 JAN
;       ADD OPCODES TO SET AND GET (SEPARATE) HARD DISK PARAMS
;
; 11 MAR
;       MORE OF THE SAME...
;
; 12 MAR
;       GETTING THE FINAL BUGS OUT OF THE ABOVE
;       MOVED HDINIT CALL INTO INITALL
;
        PAGE
;
;       ORGANIZATION OF THIS CODE....
;
;        1.  EQUATES
;        2.  "RST" AREA
;            RST 0: POWER ON/RESET JMP
;            RST 4..7 JMP TO THE ONE-BYTE NEC/HOST READ/WRITE ROUTINES
;        3.  FINISH RESET PROCESSING
;            IDLE LOOP
;            COMMAND DECODE
;            COMMAND TABLE
;        4.  POWER-ON INIT
;        5.  VARIOUS SHORT COMMAND PROCESSORS
;        6.  THE ONE-BYTE NEC/HOST READ/WRITE ROUTINES
;        7.  BOOT/SYSWRT ORGANIZER
;        8.  FLOPPY DISK SEEK STUFF
;        9.  FLOPPY DISK HEAD LOAD
;       10.  FORMAT TRACK MAINLINE AND FLOPPY DISK FORMAT
;       11.  CALCLUATE PHYSICAL SECTOR FOR FLOPPY DISK
;       12.  FLOPPY DISK READ/WRITE
;KK     13.  FLOPPY DISK READ-PHYSICAL-ID (FOR DENSITY INFO)
;KK     14.  FLOPPY DISK FORCE WRITE (OF MODIFIED SECTORS), DUMP BUFTAGS
;KK     15.  NEC INTERRUPT SERVICE WITH ERROR CODE GENERATOR AND RESET$NEC
;       16.  ALL HARD DISK ACCESS CODE
;       17.  ALL DMA STUFF (FOR NORMAL READ/WRITE, MEM DISK AND MOVE MEMORY)
;       18.  BLOCK TRANSFER TO/FROM HOST AND RELATED STUFF
;       19.  SETUPS FOR PHYSICAL DISK CHANGE
;       20.  LOGICAL/PHYSICAL DEVICE SUBROUTINES
;       21.  SET AND USE RETRY COUNTERS
;KK     22.  SET SKEW (HOST CALL); BUILD FPY SKEW TABLE, FPY SECTOR OFFSET STUF
;               RETURN VERSION NUMBER (1.1 AND LATER)
;       23.  MACROS TO HELP BUILD FDREL; FDREL; HARD DISK DPB
;       24.  DEBUG SUBROUTINE
;       25.  RAM ORGANIZATION
;
;KK 22 MAR 86
;       DAVE COMPLETES MODIFICATIONS
```

```
;       FUNCTIONS ADDED:
;       SETRAMDPB       36  24  SET FLOPPY DPB AND PARAMETERS
;       GETRAMDPB       37  25  GET FLOPPY DPB AND PARAMETERS
;       DORID           38  26  READ I.D. INFO TO EXTENDED STATUS
;       INDIVSTEP       39  27  SET INDIVIDUAL STEP RATES
;       SETTPI          40  28  SET TRACK DENSITY STATUS (48-96 TPI)
;       GETTPI          41  29  INVERSE OF ABOVE
;       GETVERS         42  2A  RETURN VERSION TO HOST
;       DUMPTAGS        43  2B  RETURN BUFTAGS TO HOST
;
;       ADDITIONAL NOTES:
;       - 48 TPI DISKS MAY BE USED IN 96 TPI UNITS. USE FUNCTION 40 28H TO
;       SET/RESET THIS ATTRIBUTE.  IF SET, CPS$TRACK:= CPS$TRACK * 2.
;       - DPB PARAMETER "OFF" IS NOW TRACK OFFSET AND NOT CYLINDER OFFSET.
;       THIS MAKES IT POSSIBLE TO R/W KAYPRO IV DISKS AND OTHERS WHICH HAVE
;       A ONE TRACK (NOT ONE CYLINDER) OFFSET.  FOR SINGLE SIDED DISKS, TRACK
;       OF COURSE IS SAME AS CYLINDER.  CPS AND FDREL HAD TO BE MODIFIED
;       TO SUPPORT THE ABOVE.  SEPERATE ENTRY IS NOW AVAILABLE FOR SEEK$CPS.
;       - WHEN A SEEK TRACK (CPS$TRACK) IS GREATER THAN THE NUMBER OF
;       CYLINDERS OF A FLOPPY UNIT, CPS$SIDE IS INCRIMENTED AND THE NUMBER
;       OF PHYSICAL CYLINDERS IS SUBTRACTED FROM CPS$TRACK.  THIS MAKES IT
;       POSSIBLE TO R/W DISKS THAT ALLOCATE SIDE ZERO COMPLETELY BEFORE
;       ALLOCATING ANYTHING TO SIDE ONE (LIKE VECTOR GRAPHICS).
;       - EVEN WITH THE ABILITY TO ADD A BIAS TO SECTOR AND SIDE IDs
;       (FUNCTION 36 24H), IT IS ALAS, STILL NOT POSSIBLE TO R/W SOME TYPES.
;       OF THOSE ARE DISKS THAT HAVE EVEN PHYSICAL SECTOR ID NUMBERS ON ONE
;       SIDE AND ODD ON THE OTHER, DISKS THAT HAVE FALSE "R" ID FIELDS
;       (COMPUGRAPHIC USES FE, PRESUMABLY THEY USE A WESTERN DIGITAL 179X
;       CONTROLLER), DISKS THAT ARE BYTE COMPLEMENTED (THOUGH THE BIOS
;       COUD FIX THAT), ETC.
;
;KK 27 APR 86
;       BUG FOUND: RECAL NOT WORKING ON 96 TPI SOMETIMES BECAUSE
;       NEC STEPS IN 77 TIMES AND GIVES UP
;       SOLUTION: ON RECAL, DO IT TWICE UNCONDITIONALLY.  IF IT WORKED
;       THE FIRST TIME, SO WHAT?  LITTLE TIME IS WASTED TRYING SECOND TIME.
;
;KK 07 DEC 86
;       ADD FUNCTIONS TO ENABLE HOST DIAGNOSTIC SOFTWARE TO USE DYSAY DDD
;       (DIGITAL DIAGNOSTIC DISKETTE) FOR ALIGNMENT, ETC. FUNCTIONS INCLUDE
;       ADDING BIT TO PARAMETER OF HOST FORMAT COMMAND SO THAT HOST MAY CALL
;       OMNI TO "VERIFY" A TRACK STARTING AT A GIVEN PHYSICAL SECTOR.  ALSO
;       A FUNCTION HAS BEEN ADDED TO RETURN THE BUFTAGS TO "SEE" WHICH SECTORS
;       HAVE BEEN SUCCESFULLY READ. A FUNCTION TO RETURN OMNI FIRMWERE VERSION
;       NUMBER (STARTING 11H FOR 1.1 "AA11.ASM") TO HOST HAS BEEN ADDED.
;
;KK 22 DEC 86
;       BUG IN BOOT FIXED. LOCDPB WAS USING SEEKTRK (WHICH IS SEEK CYLINDER)
;       AND R/W IN BOOT WAS USING CPS$TRACK. THIS CAUSED LOCDPB TO REPORT THE
;       DENSITY OF THE WRONG TRACK OF 8 INCH IN WHICH THE BOOT CYLINDER IS OF
;       A DIFFERENT DENSITY. FAILURE WAS ON THE SECOND TRACK.
;       I DON'T LIKE THE WAY POWER ON INITIALIZATION WAS DONE. THE ORDER HAS
;       BEEN CHANGED. TABLES WERE BEING ACCESSED BEFORE THEY WERE BEING INIT-
;       IALIZED - I THINK.
;
;KK 30 DEC 86
;       COMPLETED TIMER FOR PREAD/PWRITE TO CHECK DRQ FROM 765 TO SEE IF
;       WE HAVE A 5 INCH WITHOUT A DISK. ON TIMEOUT THE 765 IS RESET
;       (REQUIRES HARDWARE MOD.) AND AN ERROR CODE IS FAKED AND THEN
;       EXECUTION RESUMES AN "BEGIN$0" WHICH RESETS STACK AND WAITS FOR
;       HOST COMMANDS (MAIN IDLE LOOP). LOCDPB NOW ALSO TIMES OUT AND FAKES
;       AN ERROR CODE. THE DRQ CHECK IS DONE BY A RIM INSTRUCTION. THIS ALSO
;       REQUIRES A HARDWARE MODIFICATION AND PROBABLY RENDERS THIS FIRMWARE
;       INCOMPATABLE WITH UNMODIFIED BOARDS. IT IS A SIMPLE ONE THOUGH.
;       SIMPLY CONNECT SID OF 8085 TO DRQ OF 765 - NO CUT TRACES.
;
;KK 31 DEC 86 (11:55 P.M. happy new year)
;       FIXED BUG IN FORMAT. WAS NOT SETTING CPS$SIDE TO 0 RESULTING IN
;       INCORRECT NUMBER OF SIDES REPORT. WAS FORMATTING DOUBLE SIDED
;       DISKS SINGLE SIDED SOMETIMES DEPENDING ON THE VALUE THAT HAPPENED
;       TO BE IN CPS$SIDE.
;       ALSO FORCED 2 SIDED TRUE ON ALL 5 INCH AND FIXED A STACK PROBLEM
;       AT PRW10, THE RETURN ROUTINE OF PHYSICAL R/W.

        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       1.  EQUATES
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       CONFIGURING EQUATES (done with SET to reduce size of ".SYM" file)
;
VERSION EQU     11H     ;KK VERSION 1.1 (AA11.ASM)
INITRETRY SET   0503H   ;RETRYC1=3, RETRYC2=5
;
INITLOCCONT SET 08FH    ;SET FOR 8" AT POWERUP/RESET
;INITSPEC SET   SPEC8   ;SET FOR 8" AT POWERUP/RESET
;
FALSE   SET     0
TRUE    SET     NOT FALSE
DODEBUG EQU     FALSE   ;KK SET FOR DEBUG IN CP/M AT 100H
;
RIM     EQU     20H             ;** INSTRUCTION EQUATE **
SIM     EQU     30H             ;** INSTRUCTION EQUATE **
SIMMSK  EQU     0101$1011B      ;KK DEFAULT SIM VALUE
;
;          MSB                                                      LSB
;       +-------+-------+-------+-------+-------+-------+-------+-------+
;SIM    |  SOB  |  USE  |  NOT  | RESET |  USE  | MASK  | MASK  | MASK  |
;INST   |       |  SOB  |  USED |RST 7.5| MASKS | 7.5   | 6.5   | 5.5   |
;       +-------+-------+-------+-------+-------+-------+-------+-------+
;
;          MSB                                                      LSB
;       +-------+-------+-------+-------+-------+-------+-------+-------+
;RIM    |  SIB  |  7.5  |  6.5  |  5.5  | 0: DI |  7.5  |  6.5  |  5.5  |
;INST   |       |ACTIVE |ACTIVE |ACTIVE | 1: EI |MASKED |MASKED |MASKED |
;       +-------+-------+-------+-------+-------+-------+-------+-------+
;
;KK     SOB:    RESET NEC
;       SIB:    NOT USED?
;       5.5:    ?? ON SOME SORT OF JUMPER ??
;       6.5:    WD1001 INTERRUPT
;       7.5:    NEC CHIP INTERRUPT
;
;
;                                                       LSB
;STATUS +-------+-------+-------+-------+-------+-------+-------+-------+
;       |       |       |       |       | FROM  |0: ON  |0: ON  |  TO   |
;4018H  |       |       |       |       | HOST  |1: OFF |1: OFF | HOST  |
;I/O 58 +-------+-------+-------+-------+-------+-------+-------+-------+
;                                        | SW# 6   SW# 4 |
;       FROM HOST           TO HOST      |   SWITCH IS   |
;       0: DATA NOT READY   0: OK TO SEND | ABOVE VOLTAGE |
;       1: DATA READY       1: HOST HAS NOT +---REGULATOR---+
;                              TAKEN OUR LAST BYTE
;
```

```
;
;LOCAL     MSB                                                       LSB
;CONTROL+-------+-------+-------+-------+-------+-------+-------+-------+
;       |  DMA  | HEAD  | 0: 8" |  NOT  |    4-BIT S100 DMA PRIORITY    |
;C006H  |DIREC'N| LOAD  | 1: 5" | USED  |                               |
;I/O C6 +-------+-------+-------+-------+-------+-------+-------+-------+
;
;          0: DMA DIRECTION: S100 ==> OUR RAM
;             ENABLE WAIT STATE SYNC WITH NEC
;
;          1: DMA DIRECTION: OUR RAM ==> S100
;             DISABLE WAIT STATE SYNC WITH NEC
;
;
;
;S100      MSB                                                       LSB
;CONTROL+-------+-------+-------+-------+-------+-------+-------+-------+
;       |       |       |       |       |       |       |       |       |
;C007H  |       |       |       |       |       |       |       |       |
;I/O C7 +-------+-------+-------+-------+-------+-------+-------+-------+
;


;       HARDWARE EQUATES
;
;       (MOST ARE DONE WITH "SET" TO KEEP THEM OUT OF
;        THE MAC ".SYM" FILE -- THE SYMBOLS ARE NOT CHANGED)

WDDATA          SET     4000H   ;HARD DISK DATA TRANSFER PORT
WDDATAIO        SET     40H
WDERROR         SET     4001H   ;HARD DISK ERROR REGISTER
WDERRORIO       SET     41H
WDSECTCNT       SET     4002H   ;HARD DISK NUMBER OF SECTORS??
WDSECTCNTIO     SET     42H
WDSECT          SET     4003H   ;HARD DISK SECTOR NUMBER
WDSECTIO        SET     43H
WDLOTRK         SET     4004H   ;HARD DISK TRACK NUMBER (LOW 8 BITS)
WDLOTRKIO       SET     44H
WDHITRK         SET     4005H   ;HARD DISK TRACK NUMBER (HIGH ORDER BITS)
WDHITRKIO       SET     45H
WDSPEC          SET     4006H   ;HARD DISK SPECS (STEP RATE ETC)
WDSPECIO        SET     46H
WDSTAT          SET     4007H   ;HARD DISK STATUS
WDSTATIO        SET     47H
WDCMD           SET     4007H   ;HARD DISK COMMAND
WDCMDIO         SET     47H

WDFMTCMD        SET     50H
WDWRITECMD      SET     30H
WDREADCMD       SET     20H
HDSECS          SET     16
HDSKEW          SET     2       ;VALUE FOR SKEW FOR FORMAT
;
;       HARD DISK SELECTION DEFERRED INTO OS INIT CODE
;
HDRECAL SET     TRUE            ;PULL COND ASSEM IF PROBLEM RESOLVED
HDRECAL SET     FALSE           ;LEAVE IN FOR SA712

;
;       VARIOUS EQUATES THAT APPLY TO HOST SOFTWARE
;
LEN$DPB         SET     15
LEN$DPBEX       SET     17
LEN$BPB         SET     15
```

```
;
;       HARDWARE PORT EQUATES
;
NECSTAT         SET     4008H
NECSTATIO       SET     48H
NECDAT          SET     4009H
NECDATIO        SET     49H

SYSDATIN        SET     4010H           ;DATA TO/FROM S100 SYSTEM
SYSDATINIO      SET     50H

SYSSTAT         SET     4018H           ;STATUS REGISTER FOR XFERS TO/FROM HOST
SYSSTATIO       SET     58H
SYSOUTSTAT      SET     08H
SYSINSTAT       SET     01H

RAMORG          EQU     8000H
RAMEND          EQU     RAMORG+2800H    ;LBA+1 OF RAM

LOCDMAHI        SET     0C000H          ;LOCAL RAM DMA ADDRESS BITS 15-8
LOCDMAHIIO      SET     0C0H
LOCDMALO        SET     0C001H          ;LOCAL RAM DMA ADDRESS BIT 7 ONLY
LOCDMALOIO      SET     0C1H
SYSDMAEX        SET     0C002H          ;S100 SYSTEM DMA ADDRESS BITS 23-16
SYSDMAEXIO      SET     0C2H
SYSDMAHI        SET     0C003H          ;S100 SYSTEM DMA ADDRESS BITS 15-8
SYSDMAHIIO      SET     0C3H
SYSDMALO        SET     0C004H          ;S100 SYSTEM DMA ADDRESS BITS 7-0
SYSDMALOIO      SET     0C4H
SYSDATOUT       SET     0C005H          ;DATA TO S100 SYSTEM
SYSDATOUTIO     SET     0C5H
LOCCONT         SET     0C006H          ;LOCAL CONTROL
LOCCONTIO       SET     0C6H
DMASTAT         SET     0C007H          ;S100 CONTROL FOR DMA XFER
DMASTATIO       SET     0C7H

SYSSTATWR       SET     080H            ;** SET INTO DMASTAT TO CONTROL
SYSSTATRD       SET     091H            ;** DMA TRANSFERS TO/FROM HOST

;       NOTES ON THE ABOVE

;       LOCDMALO IS ONLY BIT 7.  IE OUR SIDE OF DMA
;       TRANSFERS ARE DONE ON 128-BYTE BOUNDRIES

;       SEE BLOCK AT START OF LISTING FOR LOCCONT AND DMASTAT

;****************************************
;       NEC 765 (INTEL 8272) OPCODES    *
;****************************************

READDATA        SET     26H             ;WITH "SKIP DELETED DATA ADDRESS MARK"
READID          SET     0AH
WRITEDATA       SET     05H
FORMAT          SET     0DH
RECAL           SET     07H
READINT         SET     08H
SPECIFY         SET     03H
DRIVESTAT       SET     04H
SEEKCMD         SET     0FH
;
        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
;
;       2.  "RST" AREA
;           RST 0: POWER ON/RESET JMP
;           RST 4..7 JMP TO THE ONE-BYTE NEC/HOST READ/WRITE ROUTINES
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$


        ORG     0
        IN      SYSDATINIO      ;CLEAR JUNK IN LATCHES
        JMP     INIT

        ORG     20H             ;RST 4 ADDRESS
        JMP     SENDCHAR        ;GO OUTPUT <A> TO HOST
PUTHOST EQU     4

        ORG     28H             ;RST 5 ADDRESS
        JMP     GETCHAR         ;GO GET A BYTE FROM HOST
GETHOST EQU     5

        ORG     30H             ;RST 6 ADDRESS
        JMP     SENDNEC         ;GO OUTPUT <A> TO NEC
PUTNEC  EQU     6


        ORG     38H             ;RST 7 ADDRESS
        JMP     GETNEC1         ;GO GET DATA FROM NEC
GETNEC  EQU     7

        ORG     3CH
RST75:  DI
        JMP     NECSERV
;
        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       3.  FINISH RESET PROCESSING
;           IDLE LOOP
;           COMMAND DECODE
;           COMMAND TABLE
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

        IF DODEBUG
        ORG     100H
        ENDIF
INIT:
        LXI     SP,STACK
        MVI     A,INITLOCCONT
        OUT     LOCCONTIO       ;SET LOCAL CONTROL FOR 8"
        STA     CURLCONT        ;SAVE FOR USE BY OTHERS
;
;       IF POWER ON THEN INIT EVERYTHING
;
;       WE CAN TELL BECAUSE RAM DOES NOT COME UP AS 1234H
;
        LHLD    H1234
        MVI     A,12H
        XRA     H               ;<A>=0 IF <H>=12H
        MOV     B,A
        MVI     A,34H
        XRA     L               ;<A>=0 IF <L>=34H
        ORA     B               ;<A>=0 IF <HL>=1234H
        JNZ     INITALL         ;IF POWER ON
;
;       A PROBLEM HAS BEEN REPORTED IN THE ABOVE: IF THE SYSTEM IS POWERED
;       DOWN FOR 5 TO 10 SECONDS, THE OMNIDISK STATIC RAM WILL BEGIN TO
;       FAIL -- BUT THERE IS NO WAY TO BE SURE THAT OUR 1234H WILL GO
;       BEFORE SOME OTHER RAM LOCATATION.
;
        CALL    RESET$NEC       ;KK DO A RESET OF NEC AND A DOSPEC ETC.
        XRA     A
        STA     DEBUGB

        LXI     H,-1            ;100% JUNK SO AS TO AVOID A MATCH LATER
        SHLD    BUFTRKS

BEGINGENSTAT:   STA     GENSTAT

;       ....                    ;FALL INTO COMMAND DECODE

;*****************************************************************************
;                                                                            *
;               COMMAND DECODE                                               *
;                                                                            *
;*****************************************************************************

BEGIN$0 LXI     SP,STACK

BEGIN:
        LXI     H,BEGIN         ;** PUSH "RTN" ADDRESS
        PUSH    H               ;** FOR COMMAND ROUTINES
        LXI     H,MODE
        MOV     A,M             ;##
        ANI     8FH             ;## RESET BOOT/SYSWRT FLAG IN MODE BYTE
        MOV     M,A             ;##

        LXI     H,22222         ;22222 = 4000 * 250 / 45
;                                                   45 T-STATES/LOOP
;                                              250 MS IS OUR 1/4 SEC TIMER
;                                        4000 T-STATES / MS
;
GETCMD1 IN      SYSSTATIO       ;10 <===T-STATES FOR INSTRUCTION
        RRC                     ; 4
        JC      GETCMD2         ; 7     IF SOMETHING FROM HOST
        DCX     H               ; 6
        MOV     A,L             ; 4
        ORA     H               ; 4
        JNZ     GETCMD1         ;10     IF NOT DONE TIMING 1/4 SEC
;                               ---
;                                45 T-STATES / LOOP
;
        LXI     H,TIME$WRITE
        DCR     M
        JZ      FWT             ;IF TIME TO FORCE WRITES
        INX     H               ;<HL>=TIME$HLOAD
        DCR     M
        RNZ                     ;IF NOT TIME TO UNLOAD HEADS
;
;       IT IS NOW TIME TO UNLOAD THE HEADS
;
;UNLOAD:
        LXI     H,CURLCONT
        MOV     A,M
        ANI     NOT 40H         ;NOT HEAD LOAD BIT
```

```
        MOV     M,A
        OUT     LOCCONTIO
        ANI     20H
        RZ                      ;IF 8" DRIVES ACTIVE
        LDA     RLINE5
        ORA     A
        RNZ                     ;IF 5" DRIVES HAVE READY LINE
        LXI     H,FD500+PDTFLG
        LXI     D,FD501-FD500
        MVI     C,4
UNL1    MOV     A,M
        ANI     NOT 02          ;SET "DROP READY" CONDITION
        MOV     M,A
        DAD     D               ;NEXT 5" DEVICE TABLE
        DCR     C
        JNZ     UNL1
        RET
;
;
GETCMD2 IN      SYSDATINIO
        INR     A               ;(TO PUT FF COMMAND INTO LIST)
        CPI     CMDLIMIT
        RNC                     ;IF INVALID COMMAND TREAT AS NOP
        MVI     B,0
        MOV     C,A
        LXI     H,CMDTBL
        DAD     B
        DAD     B               ;<HL> = ADDRESS W/I CMDTBL
        MOV     E,M
        INX     H
        MOV     D,M
        XCHG                    ;<HL> = PROCESS ADDRESS
        PCHL                    ;GO DO IT -- NOTE: <B>=0

CMDTBL:
        DW      INITALL         ;-1  FF  POWER-ON RESET
        DW      BOOT            ;0   00  BOOT DISK
        DW      SYSWRT          ;1   01  WRITE SYSTEM (INVERSE OF BOOT)

        DW      SETMODE         ;2   02  SET PORT OR DMA DATA XFER (+DMA PRI)
        DW      FMTTRK          ;3   03  FORMAT TRACK
        DW      DLDS            ;4   04  DEFINE LOGICAL DEVICE SET
        DW      GLDS            ;5   05  GET LOGICAL DEVICE SET
        DW      XFWT            ;6   06  FORCE PENDING WRITES
        DW      RBOOTPH         ;7   07  RESET (BOOT PROM) PHANTOM

        DW      SETSKEW         ;8   08  SET SKEW BYTE INTO DEVICE TABLE
        DW      SETDRV          ;9   09  SELECT (LOGICAL) DRIVE
        DW      SETTRK          ;10  0A  LOGICAL SEEK
        DW      SETSEC          ;11  0B  SET SECTOR
        DW      SETDMA          ;12  0C  SET DMA ADDRESS
        DW      READ            ;13  0D  READ LOGICAL SECTOR
        DW      XWRITE          ;14  0E  WRITE LOGICAL SECTOR

        DW      SETHEAD         ;15  0F  SELECT HEAD (ONLY USED ON HARD DISK)
        DW      MOVMEM          ;16  10  MOVE MEMORY WITH DMA
        DW      GETDPB          ;17  11  "READ" CP/M DPB FROM SELECTED DRV/TRK
        DW      DEFMEMD         ;18  12  DEFINE MEMORY DISK
        DW      SEPSEEK         ;19  13  FORCE MOTION SEEK
        DW      TESTIOIN        ;20  14  DEBUG READ FROM BUFFER
        DW      TESTIOOUT       ;21  15  DEBUG WRITE TO BUFFER
        DW      GETGENSTAT      ;22  16  GET GENERAL STATUS
        DW      GETEXTSTAT      ;23  17  GET LONG (EXTENDED) STATUS OF R/W ERR
        DW      SETRETRY        ;24  18  SET RETRY LOGIC COUNTERS
        DW      GETDPBEX        ;25  19  GET DPB EXTENDED (CP/M 3.0)
        DW      GETBPB          ;26  1A  GET MS-DOS BPB
        DW      GETMCSTAT       ;27  1B  GET MEDIA CHANGE STATUS
        DW      SETFDP          ;28  1C  SET FLOPPY DISK PARAMETERS
        DW      PFREAD          ;29  1D  PHYSICAL FORMAT READ
        DW      PFWRITE         ;30  1E  PHYSICAL FORMAT WRITE
        DW      HDPARM          ;31  1F  SET HD PARM (#HDS,PCOMP,STP,CTLR TYP)
        DW      HDTBLS          ;32  20  SET HD TABLES (DEV, OFFSET, OS BLOCK)
        DW      ERRIGNORE       ;33  21  IGNORE FD ERR & MARK BUFFER VALID
        DW      SETIHDP         ;34  22  SET INDIVIDUAL HD PARMS
        DW      GETIHDP         ;35  23  GET INDIVIDUAL HD PARMS AND TABLES
;KK
        DW      SETRAMDPB       ;36  24  SET FLOPPY DPB AND PARAMETERS
        DW      GETRAMDPB       ;37  25  GET FLOPPY DPB AND PARAMETERS
        DW      DORID           ;38  26  READ I.D. INFO TO EXTENDED STATUS
        DW      INDIVSTEP       ;39  27  SET INDIVIDUAL STEP RATES
        DW      SETTPI          ;40  28  SET TRACK DENSITY STATUS
        DW      GETTPI          ;41  29  INVERSE OF ABOVE
        DW      GETVERS         ;42  2A  RETURN VERSION NUMBER
        DW      DUMPTAGS        ;43  2B  RETURN BUFFER TABS TO HOST

CMDLIMIT EQU    ($-CMDTBL)/2

        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;        4.  POWER-ON INIT
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

INITALL:
        DI
        XRA     A
        LXI     H,RAMORG
        LXI     D,RAMEND-RAMORG
IA1     MOV     M,A
        INX     H
        DCR     E
        JNZ     IA1             ;IF MORE RAM TO ZERO
        DCR     D
        JNZ     IA1             ;IF MORE RAM TO ZERO
;
;       MOVE POWER-UP CONSTANTS TO RAM
;
        LXI     D,PUCONST
        LXI     H,PUDEF
        MVI     C,PULEN
PUMOV   LDAX    D
        MOV     M,A
        INX     H
        INX     D
        DCR     C
        JNZ     PUMOV           ;IF MORE POWER-UP CONSTANTS TO MOVE TO RAM

        LXI     H,DEBUGC
        SHLD    DEBUGA
;
;       SET ALL DEVICE TABLES TO UNDEFINED AND
;       INIT FLOPPY DISK SKEW TABLES
;
        LXI     B,12            ;<BC>=PHYS DEV CODE - 10H
IA2     LXI     H,PDTFINDERS
```

```
        DAD     B
        DAD     B
        MOV     E,M
        INX     H
        MOV     D,M             ;<DE>=PHYSICAL DEVICE TABLE ADDRESS
        MVI     A,0FFH          ;** MARK PHYSICAL DEVICE
        STAX    D               ;** TABLE AS UNDEFINED
        LXI     H,LENPDTH
        DAD     D               ;<HL>=PDT ADRS FOR 1ST SKEW CONTROL BYTE
        XCHG                    ;...TO <DE>

        MOV     A,C             ;8"=0..3; 5"=4..7; NOT FLOPPY=8..12
        CPI     8
        JNC     IA4             ;IF NOT FLOPPY DISK DEVICE TABLE
        LXI     H,STD8+5        ;** ASSUME
        MVI     B,NFMT8         ;** 8" DRIVE
        CPI     4
        JC      IA3             ;IF 8" DRIVE
        LXI     H,STD5+5        ;ELSE...
        MVI     B,NFMT5         ;...SETUP FOR 5" DRIVE
IA3:
        MOV     A,M             ;SKEW FACTOR FROM FDRELTAB...
        STAX    D               ;...TO DEVICE TABLE
        INX     D               ;<DE>=ADRS OF NEXT SKEW CTL BYTE IN DEV TABLE
        PUSH    D
        LXI     D,LENFDREL
        DAD     D               ;<HL>=ADRS OF SKEW BYTE IN NEXT FDREL SLOT
        POP     D
        DCR     B
        JNZ     IA3             ;IF MORE SKEW CONTROL BYTES TO INIT

IA4     DCR     C
        JP      IA2             ;IF MORE SKEW TABLES TO INIT
;
;       NOW READ THE SWITCH BITS AND SETUP A: DISK
;
        IN      SYSSTATIO
        ANI     6               ;ISOLATE OUR SWITCH BITS
        RLC                     ;SHOULDN'T THIS BE RRC?
        ORI     10H             ;FORM PHYSICAL DEVICE CODE FOR BOOT DEVICE
        STA     LOGDEVTBL       ;SETUP LOGICAL-TO-PHYSICAL MAPPING FOR A:
        XRA     A               ;** FIND PHYSICAL DEVICE TABLE
        CALL    MLP             ;** ADDRESS FOR A: DEVICE
        MVI     M,0             ;PHYS-TO-LOG MAPPING TO PDT TO FINISH A: SETUP

        CALL    HDINIT          ;INIT FOR HARD DISK
        JMP     INIT            ;NOW DO NORMAL RESET PROCESSING

PUCONST DB      6,20            ;WRITE AND HEAD LOAD DELAYS (250MS TICKS)
        DB      00FH,00FH       ;8/5 STEP RATE (16MS & 32MS)
        DB      0FH,0FH,0FH     ;KK EXTENDED FOR INDIVIDUALIZED STEP RATES
        DB      0FH,0FH,0FH
        DB      200,200         ;8/5 HEAD LOAD TIME IN MS
        DB      50,50           ;8/5 HEAD SETTLE TIME (MS DELAY AFTER SEEK OK)
        DW      INITRETRY       ;READ/WRITE RETRY COUNTS FOR FLOPPY
        DW      1234H
        DB      2               ;HD # HEADS (SET BY OS INIT CODE)
        DB      100             ;PRE COMP CYLINDER (FOR WRITES ABOVE THIS CYL)
        DB      0FH             ;HARD DISK STEP RATE (INIT TO S...L...O...W)
        DB      020H            ;HARD DISK CTLR W/O ECC; 512-BYTE SECTORS
        DB      (JMP)           ;USED IN BOOT/SYSWRT PROCESS
PULEN   EQU     $-PUCONST
;
        PAGE
```

```
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       5.  VARIOUS SHORT COMMAND PROCESSORS
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

;
;       GET MEDIA CHANGE STATUS
;
GETMCSTAT:
        CALL    MAKECUR
        DW      MCS1            ;GO HERE IF HARD DISK OR MEMORY DISK
        LHLD    CURPDTADRS
        INX H ! INX H           ;FORM ADDRESS OF FLOPPY DISK FLAGS BYTE
        MOV     A,M             ;FETCH FLOPPY DISK FLAGS BYTE
        ANI     3               ;ISOLATE BITS OF INTREST
        DCR     A               ;CONVERT TO -1, 0, +1 FOR MS-DOS USE
        RST     PUTHOST
        RET

MCS1    MVI     A,1             ;+1 (NO MEDIA CHANGE) IS ALLWAYS HARD/MEM STAT
        RST     PUTHOST
        RET

;
;       RESET BOOT PROM PHANTOM
;
RBOOTPH: OUT    SYSDMAEXIO      ;STROBE, NOT CONTENTS, RESETS BOOT PROM PHANTOM
        RET

;
;       DEBUG READ/WRITE RAM
;
TESTIOIN:
        CALL    TIOADRLEN
TESTIOIN1:
        RST     GETHOST
        MOV     M,A
        INX     H
        DCX     D
        MOV     A,D
        ORA     E
        JNZ     TESTIOIN1
        RET

TESTIOOUT:
        CALL    TIOADRLEN
TESTIOOUT1:
        MOV     A,M
        RST     PUTHOST
        INX     H
        DCX     D
        MOV     A,D
        ORA     E
        JNZ     TESTIOOUT1
        RET

TIOADRLEN:                      ;GET ADDRESS AND LENGTH FOR DEBUG READ/WRITE
        RST     GETHOST
        MOV     L,A
        RST     GETHOST
        MOV     H,A             ;HL=ADDRESS OF BLOCK TO READ OR WRITE
```

```
        RST     GETHOST
        MOV     E,A
        RST     GETHOST
        MOV     D,A             ;HL=LENGTH OF BLOCK TO READ OR WRITE
        ORA     E
        RNZ                     ;LENGTH<>0 THEN THIS IS READ OR WRITE
        POP     PSW             ;ELSE...KILL RETURN ADDRESS
        PCHL                    ;AND OFF WE GO. . . .

;       GETGENSTAT - DELIVER GENSTAT BYTE TO HOST
;
;       THE GENERAL STATUS BYTE IS CONFIGURED AS FOLLOWS:
;
;       11xxxxxx - ERROR DETECTED IN PHYSICAL FORMAT
;       10xxxxxx - ERROR DETECTED IN PHYSICAL READ
;       01xxxxxx - ERROR DETECTED IN PHYSICAL WRITE
;       00010000 (10H) - LOGICAL DISK NOT DEFINED
;       00010001 (11H) - ERROR ON FMTTRK PARAMETER LIST
;       00010010 (12H) - CANT READ ID IN RPI (CALLED FROM GETDPB)
;       00010011 (13H) - RETRY FAILS TO RECOVER WRONG CYLINDER IN RPI
;       00010100 (14H) - DISK IS OF UNKNOWN DENSITY
;       more?
;
GETGENSTAT:
        LDA     GENSTAT
        JMP     SENDCHAR

;       GETEXTSTAT - DELIVER EXTENDED STATUS LIST TO HOST
;
;       EXTENDED STATUS LIST INCLUDES:
;
;       BYTES   DISCRIPTION
;       -----   -------------------------------------
;         1     GENERAL STATUS
;         1     PHYSICAL SECTOR IN ERROR
;         7     NEC765/INTEL8272 FDC 7-BYTE RESULT STATUS
;
GETEXTSTAT:
        LXI     H,GENSTAT
        MOV     A,M             ;GENERAL STATUS HEADS LIST
        MVI     M,0             ;REQUEST FOR EXTENDED STAT ZEROS GENERAL STATUS
        RST     PUTHOST         ;SEND GENERAL STATUS TO HOST
        INX     H
        MVI     E,LENESL-1      ;ADDITIONAL LENGTH TO SEND TO HOST
        JMP     GETDPB2         ;SEND <E> BYTES TO HOST
;
;KK SET INDIVIDUAL STEP RATES, HOST CALL
;
INDIVSTEP:
        CALL    GETFPYHST       ;KK GET FLOPPY DEVICE FROM HOST
        PUSH    PSW             ;KK IF BAD DEVICE CY=1, SAVE
        LXI     H,FD$STEP$RATE
        JNC     INDIVSTEP1      ;KK IF GOOD DEVICE
        LXI     H,MISCBUF       ;KK TRASH INPUT ON BAD DEVICE
INDIVSTEP1:
        CPI     14H             ;KK 5" OR 8"?
        PUSH    PSW             ;KK SAVE FLAGS
        ANI     7
        MOV     E,A
        MVI     D,0
        DAD     D               ;KK <HL> = ADDRESS OF SELECTED STEP RATE
        RST     GETHOST         ;KK STEP RATE FROM HOST
        DCR     A
        RAL ! RAL ! RAL
        MOV     B,A             ;KK TEMP SAVE
        POP     PSW             ;KK AGAIN, 5" OR 8"?
        MOV     A,B             ;KK RESTORE
        JNC     INDIVSTEP2      ;KK 5"
        RAL
INDIVSTEP2:
        ANI     0F0H
        CMA
        MOV     M,A
        POP     PSW             ;KK ERROR = (CY = 1)
        MVI     A,0
        RAL                     ;KK IF CY THEN <A>:= 1
        STA     GENSTAT
        JMP     DOSPEC


SETFDP:                         ;SET FLOPPY DISK PARAMETERS
        LXI     H,WRITE$DELAY
        CALL    GC2             ;PICK UP DELAY-AFTER-WRITE AND STAY-LOADED TIME

        RST     GETHOST         ;8" STEP TIME
        DCR     A
        RAL ! RAL ! RAL ! RAL
        ANI     0F0H
        CMA
        MVI     E,4             ;KK 4 8" DRIVES
SETFDP8:
        MOV     M,A             ;SAVE 8" STEP TIME IN "SPECIFY" FORMAT
        INX     H
        DCR     E
        JNZ     SETFDP8

        RST     GETHOST         ;5" STEP TIME
        DCR     A
        RAL ! RAL ! RAL
        ANI     0F0H
        CMA
        MVI     E,4             ;KK 4 5" DRIVES
SETFDP5:
        MOV     M,A             ;SAVE 5" STEP TIME IN "SPECIFY" FORMAT
        INX     H
        DCR     E
        JNZ     SETFDP5

        CALL    GC4             ;GET HEAD LOAD TIMES AND HEAD SETTLE TIMES
;       ...                     ;NOW DO THE SPECIFY
;
;       DOSPEC -- ISSUE SPECIFY COMMAND
;
DOSPEC:
        LDA     CURPHYSDEV      ;KK USE CURRENT DEVICE
        ANI     7
        MOV     C,A             ;KK ..FOR STEP TABLE OFFSET
        MVI     B,0
        LXI     H,FD$STEP$RATE  ;SETUP FOR 8" SPECIFY
        DAD     B
;KK     LDA     CURLCONT
;KK     ANI     20H
;KK     JZ      DOSPEC1         ;IF 8"
;KK     INX     B               ;ELSE...SETUP FOR 5" SPECIFY
DOSPEC1 MVI     A,SPECIFY       ;NEC SPECIFY COMMAND
        DI
        RST     PUTNEC
;KK     LDAX    B               ;STEP RATE WORD
```

```
        MOV     A,M             ;KK FROM TABLE INSTEAD
        RST     PUTNEC
        MVI     A,2             ;MIN (2MS) HEAD SETTLE TIME, DMA
        RST     PUTNEC
        RET

ERRIGNORE:                      ;IGNORE LAST FD ERROR
        LDA     PIOERSECT
        MOV     L,A
        LDA     RES$R
        CMP     L
        RNZ                     ;IF LAST ERROR SECTOR <> SECTOR IN RESULT VECT
        DCR     L
        RM                      ;IF NO ERROR WE DONT FIX IT
        MVI     H,BUFTAG/100H
        MOV     A,M
        ORI     80H             ;SET "SECTOR VALID" TAG IN BUFFER
        MOV     M,A
        RET

SETHEAD:
        LXI     H,SEEKHD
        JMP     GC1

SETMODE:LXI     H,MODE          ;MODE: DBXXPPPP D=1 ==> DMA, PPPP=DMA PRIORITY
        CALL    GC1             ;       B=1 ==> BOOT/SYSWRT IN PROGRESS
        ANI     0FH             ;PRIORITY BITS ONLY
        MOV     B,A
        MOV     A,M             ;CURLCONT (FOLLOWS MODE BYTE IN RAM)
        ANI     0F0H            ;PULL PREVIOUS PRIORITY BITS
        ORA     B               ;MERGE PRIORITY BITS FROM HOST
        MOV     M,A             ;UPDATE CURLCONT IN RAM
        OUT     LOCCONTIO
        RET

HDTBLS: CALL    GHDADRS         ;GET BASE ADRS OF DESIRED UNIT
        LXI     D,PDTHDEV       ;BEGIN BY GETTING DEVICE CODE
        DAD     D
        MVI     B,3
        CALL    GC              ;SELECT CODE AND "HARDWARE OFFSET" TO DEV TABLE
;
;       ON RETURN <HL> = ADDRESS OF LENGTH OF HARD DISK OS BLOCK LENGTH
;
HDT3    RST     GETHOST         ;GET LENGTH OF FOLLOWING "OS TABLE"
        MOV     B,A             ;** SETUP FOR GC,
        INR     B               ;** (GET COUNT) LOOP
        JMP     GCA             ;NOTE: BOTH COUNT AND OS TABLE GO TO DEV TABLE

SETIHDP:
        CALL    GHDADRS         ;GET BASE OF DEV TBL OF DESIRED UNIT
        LXI     D,PDTHHEADS
        DAD     D               ;FORM ADDRESS W/I DEVICE TABLE
        MVI     B,3             ;GET #HEADS, PRE-COMP CYL AND STEP RATE
        JMP     GC

GETIHDP:
        CALL    GHDADRS
        LXI     D,PDTHHEADS
        DAD     D
        MVI     C,PDTHOSLEN-PDTHHEADS+1 ;XFER # OF HEADS .. LEN BYTE
PIP1    MOV     A,M
        RST     PUTHOST
        INX     H
        DCR     C
        JNZ     PIP1            ;IF NOT DOWN TO LENGTH BYTE
        MOV     E,A             ;LENGTH BYTE
        JMP     GETDPB2         ;GO SEND TO HOST

;
;       SUBROUTINE TO GET ADDRESS W/I HARD DISK DEVICE TABLE
;
;       ENTRY:  HOST IS ABOUT TO SEND LOGICAL UNIT
;
GHDADRS:
        RST     GETHOST         ;GET LOGICAL UNIT OF FOLLOWING HARD DISK DATA
        CALL    MLP             ;MAP LOGICAL TO PHYSICAL
        JC      GHA1            ;IF BAD LOGICAL DEVICE
        ANI     NOT 3
        CPI     18H
        RZ                      ;IF HARD DISK
GHA1    LXI     H,0000          ;FOR "NOT HARD DISK" TRASH DATA "INTO" PROM
        RET


SETDRV: LXI     H,SEEKDEV
        JMP     GC1

SETTRK: LXI     H,SEEKTRK       ;LOW ORDER BITS OF TRACK (CYLINDER)
        JMP     GC2

SETSEC: LXI     H,SEEKSEC
        JMP     GC1

SETDMA: LXI     H,SEEKDMALO
        JMP     GC3             ;GET DMA ADRS BITS 7-0, BITS 15-8, BITS 23-16

SETRETRY:
        LXI     H,SEEKRETRY1    ;GET FIRST AND SECOND
        JMP     GC2             ;R/W RETRY CONTROL BYTES

GC6:    INR     B
        INR     B
GC4:    INR     B
GC3:    INR     B
GC2:    INR     B
GC1:    INR     B
GC:     RST     GETHOST         ;GET PARAMETER FORM HOST
GCA     MOV     M,A
        INX     H
        DCR     B
        JNZ     GC              ;IF MORE PARAMETER BYTES LEFT
        RET
;
        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;        6.  THE ONE-BYTE NEC/HOST READ/WRITE ROUTINES
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

SENDNEC:
        PUSH    PSW
        CALL    WAIT16
SENDNEC2:
        IN      NECSTATIO       ;MAIN NEC STATUS
        ADD     A
```

```
        JNC     SENDNEC2        ;IF NOT READY TO TRANSFER DATA
;       CM      ERROR           ;IF NEC WANTS TO SEND US DATA
        POP     PSW
        OUT     NECDATIO
        RET

GETNEC1:
        CALL    WAIT16
GETNEC2:
        IN      NECSTATIO       ;MAIN NEC STATUS
        ADD     A
        JNC     GETNEC2         ;IF NOT READY TO TRANSFER DATA
;       CP      ERROR           ;IF NEC WANTS US TO SEND DATA
        IN      NECDATIO
        RET

GETCHAR:
        IN      SYSSTATIO
;       ANI     SYSINSTAT
;       JZ      GETCHAR
        RRC
        JNC     GETCHAR         ;IF NO DATA FROM HOST THEN WE WAIT
        IN      SYSDATINIO      ;ACCEPT DATA FROM HOST
        RET

SENDCHAR:
        PUSH    PSW
SENDCHAR1:
        IN      SYSSTATIO
        ANI     SYSOUTSTAT
        JNZ     SENDCHAR1
        POP     PSW
HOSTOUT OUT     SYSDATOUTIO
        RET
;
        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       7.  BOOT/SYSWRT ORGANIZER
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

;       BOOT PROCESS ------
;
;       ACCEPT BOOTLEN FROM HOST
;       FOR CYL = 0 TO WHATEVER
;         SET SIDE = 1
;           RPI
;           BOOTIO (READ)
;         SET SIDE = 2
;           IF ONLY ONE SIDE, NEXT CYL
;           RPI
;           BOOTIO (READ)
;       NEXT CYL
;       (DOES NOT "FALL THRU" BECAUSE BOOTIO RETURNS CONTROL TO IDLE LOOP)
;
;       BOOTIO ------
;
;       CALL "BSKEW" TO SETUP SKEW=1 IN SKEW LIST
;       IF SIDE = 1, THEN SEEKSEC=0
;       IF SIDE = 2, THEN SEEKSEC=PDTLSS (LOGICAL SECTORS/SIDE)
;       LOOP PDTLSS TIMES
;         CALL ARG (IE CALL READ OR CALL WRITE)
;         IF ERROR, THEN ABORT
;         DECERMENT LENGTH TO TRANSFER BY 80H
;         IF ZERO (OR NEGATIVE) THEN EXIT BACK TO IDLE
;         ADD 1 TO SEEKSEC
;       END LOOP
;
;
SYSWRT: LXI     H,WRITE         ;** SET BOOTIO
        CALL    BOOT0           ;** TO WRITE
        LDA     GENSTAT
        ORA     A
        RZ
        JMP     GETGENSTAT

BOOT:   LXI     H,READ          ;** SET BOOTIO
BOOT0:  SHLD    BOOTSUB         ;** TO READ
        XRA     A
        STA     GENSTAT         ;AT LEAST WE START W/ A CLEAN SLATE
        STA     PFIOFLAG        ;FORCE LOGICAL 128-BYTE SECTORS FOR BOOT/SYSWRT
        RST     GETHOST
        STA     BOOTLEN         ;LENGTH OF DATA TO READ (# OF 128-BYTE BLOCKS)
        CALL    FWT             ;FORCE WRITE TAGS
        LXI     H,MODE
        MOV     A,M
        ORI     40H             ;SET BOOT/SYSWRT FLAG IN MODE
        MOV     M,A
        LXI     H,0
        SHLD    BOOTTRK         ;KK KEEP OF BOOT TRACK BECAUSE..
BOOT1   SHLD    SEEKTRK         ;KK ..LOCDPB USES CYLINDER AND R/W USES TRACK
        XRA     A
        STA     CPS$SIDE        ;"CHECK BOTH SIDES" FOR LOCDPB
        CALL    MAKECUR
        DW      BOOT5           ;IF NOT FLOPPY DISK
        CALL    LOCDPB          ;DO READID, LOCATE DPB, SETUP DEVICE TABLE
        MOV     A,H
        ORA     L
        RZ                      ;IF ID UNREADABLE
        MOV     B,M             ;LOGICAL SECTORS / LOGICAL CYLINDER (FROM DPB)
        PUSH    B               ;SAVE FOR AFTER MAKECUR
;
        CALL    MAKECUR         ;SETUP LDT/PDT
        DW      0000            ;IF NOT FLOPPY DISK (BUT IT ALLWAYS IS HERE)
        LHLD    BOOTTRK         ;KK USE THIS FOR R/W
        SHLD    SEEKTRK         ;KK
        LHLD    CURPDTSKEW      ;<HL> = ADRS OF START OF SKEW LIST
        LDA     CURPDTNST
        MOV     D,A             ;<D> = NUMBER OF (PHYSICAL) SECTORS/TRACK
        MVI     E,1             ;<E> = SKEW FACTOR
        CALL    BSKEW           ;SETUP FOR NO SKEW
        POP     B               ;<B> = LOG SECTS/LOG TRACK FROM DPB
BOOT2   XRA     A
        STA     SEEKSEC         ;STARTING LOGICAL SECTOR
BOOT3   PUSH    B
        CALL    JMPBIO          ;CALL READ OR WRITE
        POP     B
        LXI     H,GENSTAT       ;SAVES 1 BYTE OVER "LDA GENSTAT" IF ERROR
        MOV     A,M
        ORA     A
        JZ      BOOT4           ;IF NO ERROR READING OR WRITING BOOT SECTORS
;
;       ERROR READING OR WRITING BOOT SECTORS
;
        XRI     40H             ;IS ERROR ECC/CRC ON HARD DISK?
```

```
        RNZ                     ;NO...THEN QUIT
        MOV     M,A             ;ZERO GENSTAT TO ALLOW READ/WRITE TO WORK
        LXI     H,HDCTYPE
        MOV     A,M
        XRI     80H             ;CHANGE HARD DISK CONTROLLER TYPE
        MOV     M,A
        PUSH    B
        CALL    JMPBIO          ;RETRY READ OR WRITE
        POP     B
        LDA     GENSTAT         ;CHECK ERROR CONDITION
        ORA     A
        RNZ                     ;IF STILL BAD
BOOT4   LXI     H,BOOTLEN
        DCR     M               ;REDUCE BLOCKS REMAINING TO READ/WRITE
        JZ      FWT             ;IF ALL DONE WITH BOOT READ/WRITE
        LXI     H,SEEKSEC
        INR     M               ;BUMP LOGICAL SECTOR
        DCR     B               ;LOGICAL SECTORS REMAINING ON TRACK
        JNZ     BOOT3           ;IF MORE LOGICAL SECTORS TO TRANSFER
        CALL    FWT             ;ELSE FORCE WRITE TAGS & DO NEXT CYLINDER
        LHLD    BOOTTRK         ;KK INCRIMENT
        INX     H               ;KK
        SHLD    BOOTTRK         ;KK
        XCHG
        LHLD    CURPDTADRS      ;KK SEE IF TWO SIDED
        INX H ! INX H           ;KK ADDRESS OF FLAGS BYTE
        MOV     A,M
        XCHG
        ANI     4               ;KK DOUBLE SIDED BIT
        JZ      BOOT4A          ;KK SINGLE SIDED
        MOV     A,H             ;KK  DIV 2 IF DOUBLE SIDED (FOR LOCDPB)
        RAR
        MOV     H,A
        MOV     A,L
        RAR
        MOV     L,A
BOOT4A:
;       LHLD    SEEKTRK
;       INX     H
        JMP     BOOT1           ;GO BACK FOR NEXT TRACK

BOOT5   MVI     B,64            ;BOTH HARD AND MEMORY DISK HAVE
        JMP     BOOT2           ;64 128-BYTE SECTORS/TRACK
;
        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;        8.  FLOPPY DISK SEEK STUFF
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

SEPSEEK:
;       CALL    WAITNBUSY
        RST     GETHOST         ;GET TRACK TO SEEK TO
        MOV     B,A
        RST     GETHOST         ;GET LOGICAL DRIVE NUMBER...
        JMP     SEEK2           ;...AND ENTER SEEK STUFF
;
;       SEEK - SEEKS CURRENT DRIVE TO GIVEN TRACK
;
SEEK$CPS: ;KK ADDED
        LDA     CPS$TRACK       ;KK USE CPS$TRACK
        JMP     SEEK1
SEEK:
        LDA     SEEKTRK         ;DESTINATION TRACK (KK CYLINDER)
SEEK1:
        MOV     B,A
        LDA     CURPDTLDN       ;LOGICAL DEVICE NUMBER -- DONT GET FROM SEEKDEV
SEEK2:
        CALL    SMS             ;START MOTION SEEK
;       ...                     ;FALL INTO WAITNBUSY

;       WAITNBUSY - WAIT UNTIL NOT BUSY
;
;       IE WAIT UNTIL ALL SEEKS ARE DONE
;
WAITNBUSY:
        CALL    WAIT16          ;DEBUG ADD DELAY....IN LOOP???
        LHLD    TRHLOAD         ;<L>=TIME TO HEADS FULLY LOADED
                                ;<H>=EXTRA TIME DELAY AFTER SEEK DONE
WNB1    IN      NECSTATIO
        ANI     0FH             ;IS ANYBODY STILL BUSY?
        JZ      WNB5            ;IF SEEK COMPLETE
        MOV     A,L
        ORA     A
        JZ      WNB1            ;IF HEADS ALL READY LOADED - JUST WAIT FOR SEEK
WNB2    DCR     L               ;REDUCE TIME REMAINING FOR HEADS TO LOAD
WNB3    MVI     A,250           ;TICK LENGTH IS 1MS
WNB4    DCR     A               ; 4<===T-STATES IN INSTRUCTION
        ORA     A               ; 4     (DELAY ALLOWS ONE BYTE COUNTER IN LOOP)
        JNZ     WNB4            ;10     IF MORE DELAY TO MAKE 1MS TICKS
        JMP     WNB1            ;--
                                ;16 T-STATES / LOOP
;
WNB5    SHLD    TRHLOAD         ;WILL STORE 0000 BEFORE WE EXIT THIS MESS
        MOV     A,H
        ORA     L
        RZ                      ;IF BOTH DELAYS ARE MET
        XRA     A
        CMP     H
        JZ      WNB6            ;IF HEADS HAVE SETTLED AFTER SEEK
        DCR     H               ;REDUCE TIME REMAINING FOR HEADS TO SETTLE
WNB6    CMP     L
        JNZ     WNB2            ;IF HEADS HAVE NOT YET LOADED
        JMP     WNB3            ;HEADS HAVE LOADED -- WAIT FOR SETTLE

SMS:
        CALL    MLP
        RC                      ;IF INVALID LOGICAL ADDRESS
        PUSH    H               ;SAVE PDT ADRS
        MOV     C,A             ;PHYSICAL DEVICE CODE (10H - 17H)
        CALL    LOAD            ;LOAD HEADS (REQ'D TO SEEK -- XXYY STRAPPED)
        POP     H               ;PDT ADRS
        INX H ! INX H           ;PDT FLAGS ADDRESS
        MOV     A,B             ;TARGET TRACK
        STA     RTRY$TRACK      ;KK SAVE FOR RWRETRY
        ORA     A
        JZ      SMS$RECAL       ;IF SEEKING TO TRACK ZERO THEN DO UNCOND RECAL
;
        MOV     A,M             ;PDT FLAGS
        ADD     A
        JNC     SMS1            ;IF RECAL IS NOT REQUIRED
        PUSH    H
        PUSH    B
        CALL    SMS$RECAL       ;START RECAL
        CALL    WAITNBUSY       ;WAIT UNTIL RECAL IS DONE BEFORE GOING ON
```

```
        POP     B
        POP     H
SMS1:
        PUSH    H               ;KK SAVE
        LHLD    CURFPYTAB       ;KK SEE IF DOUBLE STEP FLAGGED (48TPI IN 96TPI
        MOV     A,M             ;KK ..UNIT)
        ANI     10H
        JZ      SMS1A
        MOV     A,B
        ADD     A               ;KK *2 TRACK FOR DOUBLE STEP
        MOV     B,A
SMS1A:
        POP     H               ;KK
        INX     H               ;ADDRESS OF TRACK NUMBER BYTE
        MOV     A,B             ;ONCE AGAIN, OUR TARGET TRACK
        CMP     M
        RZ                      ;IF WE ARE ALLREADY ON THE DESIRED TRACK
        MOV     M,A             ;POST DESIRED TRACK NUMBER IN PDT
        MVI     A,SEEKCMD       ;SEEK COMMAND FOR NEC
        DI
        RST     PUTNEC
        MOV     A,C             ;DEVICE CODE
        ANI     03H             ;MASK OFF ONLY DRIVE SELECT BITS
        RST     PUTNEC
        MOV     A,B             ;TRACK TO GO TO
SMS3:
        RST     PUTNEC
        EI
        LXI     H,FD$SETTLE
        LDA     CURLCONT
        ANI     20H
        JZ      SMS4
        INX     H
SMS4:
        MOV     A,M             ;PICK UP HEAD SETTLE TIME
        STA     TRHSETL         ;"TIME REMAINING FOR HEAD SETTLE"
;       CALL    WAIT16          ;WE NEED TO GIVE "BUSY" TIME TO COME UP
;
WAIT16:
        XTHL                    ;** XTHL TAKES 16 T-STATES.
        XTHL                    ;** AT 4MHZ THIS IS 4USEC
        XTHL                    ;** EACH OR 16USEC FOR THE
        XTHL                    ;** FOUR WE HAVE HERE.
        XTHL ! XTHL ! XTHL ! XTHL ;EXTRA DELAY IS DEBUG STUFF
        RET                     ; (CALL/RET ADD ANOTHER 6USEC)
;
SMS$RECAL:
        PUSH    H
        CALL    SMS$RECAL1
SMS$RECAL0:
        IN      NECSTATIO
        ANI     0FH             ;KK IS ANYBODY STILL BUSY?
        JNZ     SMS$RECAL0      ;KK IF SEEK NOT COMPLETE
        POP     H
SMS$RECAL1:                     ;KK SUBROUTINE SO IT CAN BE CALLED TWICE
        DI
        MOV     A,M             ;PDT FLAGS
        ANI     7FH             ;RESET "RECAL REQUIRED" FLAG
        MOV     M,A
        INX     H               ;FORM ADDRESS OF TRACK NUMBER BYTE
        XRA     A
        MOV     M,A             ;TRACK NUMBER (ZERO) ==> PDT
        MVI     A,RECAL         ;RECALLABRATE COMMAND
        RST     PUTNEC
        MOV     A,C             ;DEVICE CODE
        ANI     03H             ;MASK OFF ONLY DRIVE SELECT BITS
        CALL    SMS3            ;KK HEAD SETTLE AND WAIT STUFF
        RET
;
        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       9.  FLOPPY DISK HEAD LOAD
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

LOAD:   LHLD    WRITE$DELAY     ;** INIT TIMERS USED TO COUNT DOWN FOR
        SHLD    TIME$WRITE      ;** FORCE-WRITE AND HEAD-UNLOAD DELAYS
        LXI     H,CURLCONT
        MOV     A,M
        ORI     40H             ;HEAD LOAD BIT
        CMP     M
        RZ                      ;IF HEADS WERE ALLREADY LOADED
        MOV     M,A
        OUT     LOCCONTIO
        ANI     20H             ;CHECK IF 5" OR 8"
        LXI     H,TRHLOAD       ;ADRS OF TIME REM FOR HEAD LOAD (1 MS TICKS)
        LXI     D,FD$LOAD$TIME
        LDAX    D
        MOV     M,A             ;SET FOR 8"
        RZ                      ;IF 8"
        INX D ! LDAX D          ;ELSE...
        MOV     M,A             ;    ...SET FOR 5"
        RET
;
        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       10.  FORMAT TRACK MAINLINE AND FLOPPY DISK FORMAT
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

FMTTRK:
        XRA     A
        STA     GENSTAT
        LDA     SEEKSEC         ;KK SAVE IN CASE NEEDED (NEW TO VERS 1.1)
        STA     VFYSEC          ;KK
        LXI     H,FMT$NUM
        CALL    GC2             ;GET FORMAT NUMBER AND FORMAT CONTROL
        CALL    MAKECUR         ; (CALLS FWT AS REQUIRED)
        DW      MD$FMT
        LXI     D,STD8          ;** SETUP FOR 8"
        MVI     B,NFMT8         ;** RELATION TABLE
        LDA     CURLCONT
        ANI     20H
        JZ      FMT1            ;IF 8"
        LXI     D,STD5          ;** SETUP FOR 5.25"
        MVI     B,NFMT5         ;** RELATION TABLE
FMT1    LXI     H,FMT$CNTL
        MOV     A,M
        CPI     20H
        JNZ     FMT3            ;IF NOT GETTING FORMAT TYPES
        XCHG                    ;FIRST RELATION ADDRESS TO <HL>
FMT2    LXI     D,103H          ;TRANSFER 3 BYTES TO HOST
```

```
        CALL    INBUFREAD3      ;SEND TO HOST
        LXI     D,LENFDREL-3
        DAD     D               ;<HL> POINTS TO TOP OF NEXT FDREL ENTRY
        DCR     B
        JNZ     FMT2            ;IF NOT DONE SENDING ALL GROUPS
        RET

;...JUST THINKING
;
;          DB   OC$FMTV         FORMAT AND VERIFY
; FMT$NUM  DB   FORMAT NUMBER
; FMT$CNTL DB   0010 0000 READ LIST OF DENSITIES
;               0001 0000 CHECK # SIDES READ ONLY
;               0001 0001 CHECK # SIDES USING FORMAT
;               0000 xxVF F=1 FOR FORMAT, V=1 FOR VERIFY, VF=00 ILLEGAL
;                    00 USE TWO HEADS, NORMAL
;                    01 USE HEAD ZERO ONLY (IF HARD DISK INIT TRACK TO E5'S)
;                    10 USE HEAD ONE ONLY
;                    11 USE HEAD ZERO NORMAL + KILL HEAD 1 USING H=81H IN CHRN
;KK             1XXX XXXX USE SEEKSEC FOR FIRST SECTOR ON VERIFY ONLY

FMT3
;       AT THIS POINT --
;
;       <A>  = FMT$CNTL
;       <B>  = 1 + MAX VALID NUMBER FOR FMT$NUM
;       <DE> = ADDRESS OF FIRST FDREL SLOT OF SELECTED SIZE
;       <HL> = ADDRESS OF FMT$CNTL
;
        XRI     10H
        MOV     C,A             ;(USED IF WE GOTO FMT10)
        ANI     NOT 01
        JZ      FMT10           ;IF TO CHECK # SIDES
        XCHG
        DCX     D
        LDAX    D               ;FMT$NUM
        MVI     E,11H           ;ERROR CODE FOR INVALID FORMAT NUMBER
        CMP     B
        JNC     LDPB$XX         ;IF INVALID FORMAT NUMBER
        MOV     C,A             ;FOR USE AS LOOP COUNTER
        LXI     D,LENFDREL
        JMP     FMT5

FMT4    DAD     D               ;FORM ADDRESS OF NEXT FDREL SLOT
FMT5    DCR     C
        JP      FMT4            ;IF NOT DONE WITH FMT$NUM * LENFDREL
        SHLD    FMT$RELADRS     ;SAVE ADDRESS OF FDREL SLOT

        CALL    SETDENS         ;SETUP STUFF FOR VERIFY (NOTE: <A>=FMT NUMBER)
        XRA     A
        STA     SEEKSEC         ;TO GET BETTER STUFF OUT OF CPS
        CALL    CPS             ;MORE SETUP FOR VERIFY
        CALL    SEEK

        MVI     A,128           ;** MIN REV TIME=166 MS.  HERE WE ADD A 128 MS
        STA     TRHLOAD         ;** OF DELAY AFTER SEEK TO ASSURE HEAD SETTLE.
        CALL    WAITNBUSY       ;DO THE DELAY

        LDA     SEEKTRK
        STA     BUFTRACK        ;SETUP FOR VERIFY
        LDA     FMT$CNTL
        ANI     0CH             ;ISOLATE HEAD GAMES STUFF
        CPI     8               ;START ON SECOND HEAD?
        MVI     A,0             ;ASSUME START ON FIRST HEAD
        JNZ     FMT7            ;IF OUR ASSUMPTION IS GOOD
FMT6    INR     A               ;ELSE...START ON SECOND HEAD
FMT7    STA     BUFSIDE
        LDA     FMT$CNTL
        ANI     1
        CNZ     FMTSUB          ;IF DOING FORMAT
        RC                      ;IF FORMAT ERROR

        LDA     FMT$CNTL
        ANI     2
        CNZ     FMTVFY          ;IF DOING VERIFY
        RC                      ;IF VERIFY ERROR

        LDA     CURPDTFLG
        ANI     4
        RZ                      ;IF NOT DOUBLE SIDED

        LDA     BUFSIDE
        ORA     A
        RNZ                     ;IF FINISHED WITH SECOND SIDE

        LDA     FMT$CNTL
        ANI     0CH
        JZ      FMT6            ;IF DOING BOTH SIDES NORMALLY
        CPI     0CH
        RNZ                     ;IF NOT TO KILL BACK SIDE
        MVI     A,80H
        JMP     FMT6            ;GO KILL BACK SIDE


;*****************************************************************************
;
;       CHECK NUMBER OF SIDES
;
;       <C> = 0 THEN DO SIDE CHECK READ ONLY
;       <C> = 1 THEN CHECK WITH FORMAT OPERATION
;
;       RETURNS TO HOST --
;
;       2: DISK IS DOUBLE SIDED
;       1: DISK IS SINGLE SIDED
;       0: DON'T KNOW (NOT READY OR OTHER ERROR)
;
;       METHOD: IF NOT READY RETURN 0
;               IF DRIVE SAYS SINGLE-SIDED THEN RETURN 1
;               IF 8" RETURN 2 (REMEMBER, 8" KNOWS # SIDES)
;               (NOW ONLY 5" LEFT)
;               IF READ ONLY METHOD CALL LOCDPB
;                 THEN IF ERROR RETURN 0
;                 ELSE RETURN # SIDES FROM DEVICE TABLE
;               (NOW 5" TO CHECK WITH FORMAT OPERATION)
;
FMT10:  MVI     A,DRIVESTAT     ;NEC "SENSE DRIVE STATUS" COMMAND
        RST     PUTNEC
        LDA     CURPHYSDEV
        MOV     B,A             ;SAVE FOR DEVICE TYPE CHECK
        ANI     3               ;UNIT NUMBER (0-3)
        RST     PUTNEC
        RST     GETNEC          ;ST3 -- OUR USE: 20H=READY, 08H=2-SIDED
        ANI     028H
        ADI     0E0H            ;WILL CARRY IF READY
        JNC     FMT10$0         ;IF DRIVE SAYS "NOT-READY"
        JZ      FMT10$1         ;IF DRIVE SAYS "SINGLE-SIDED"
        MOV     A,B             ;CURPHYSDEV
```

```
        ANI     04H
        JZ      FMT10$2         ;IF 8" (& NOT SINGLE-SIDED) THEN DOUBLE-SIDED
        DCR     C
        JZ      FMT12           ;IF TO DO # SIDES CHECK WITH FORMAT OPERATION
FMT11   XRA     A               ;KK START WITH SIDE 0
        STA     CPS$SIDE        ;KK
        CALL    LOCDPB          ;ELSE...DO SIDE CHECK READ-ONLY
        MOV     A,L
        ORA     H
        JZ      FMT10$0         ;IF ERROR READING DISK
        LHLD    CURPDTADRS
        INX H ! INX H           ;FORM ADRS OF FLAGS BYTE
        MOV     A,M
        ANI     04H             ;CHECK DOUBLE SIDED FROM DEVICE TABLE
        JZ      FMT10$1         ;IF SINGLE SIDED
FMT10$2 MVI     A,2             ;ELSE...DOUBLE SIDED
        RST     PUTHOST
        RET

;       CHECK NUMBER OF SIDES WITH FORMAT OPERATION
;
;       GENERAL METHOD:
;
;       1.  USE "STD5" FORMAT
;       2.  FORMAT USING SECOND HEAD FIRST
;           IF ERROR RETURN 1
;       3.  FORMAT USING FIRST HEAD SECOND
;       4.  DO READ-ONLY STYLE NUMBER-OF-SIDES CHECK
;
FMT12:
        CALL    SEEK
        MVI     A,128           ;** MIN REV TIME=166 MS.  HERE WE ADD A 128 MS
        STA     TRHLOAD         ;** OF DELAY AFTER SEEK TO ASSURE HEAD SETTLE.
        CALL    WAITNBUSY       ;DO THE DELAY
        LXI     H,STD5
        SHLD    FMT$RELADRS
        MVI     A,1
        STA     BUFSIDE
        CALL    FMTSUB          ;FORMAT BACK SIDE
        JC      FMT10$1         ;IF ERROR ASSUME ONLY ONE SIDE
        LXI     H,BUFSIDE
        DCR     M               ;SET FOR FRONT SIDE
        CALL    FMTSUB
        JMP     FMT11           ;GO BACK AND DO READID STUFF DENSITY CHECK

FMT10$0 XRA     A
        RST     PUTHOST
        RET

FMT10$1 MVI     A,1
        RST     PUTHOST
        RET

;*******************************************************************************
;
;       FMTSUB -- PERFORM PHYSICAL FORMAT
;
;       ENTRY:  FMT$RELADRS = ADDRESS OF FD RELATATION ENTRY
;               BUFSIDE = HEAD NUMBER (0 OR 1)
;
FMTSUB:
        LXI     H,FMTSUB2
        SHLD    NECINTADRS      ;POST INT ADRS FOR WHEN FORMAT TRACK IS DONE
        MVI     A,0C0H          ;** USED IF ERROR TO INDICATE TO HOST
        STA     OPTYPE          ;** THAT ERROR WAS DURING FORMAT OP
        LHLD    FMT$RELADRS     ;ADDRESS OF FDREL SLOT
        MOV     A,M             ;POINT TO FIRST ENTRY IN TABLE
        ANI     040H            ;MASK OFF SIZE DATA
        ORI     FORMAT          ;= NEC FORMAT COMMAND (THE 40H SETS FM/MFM)
        DI
        RST     PUTNEC
        LDA     BUFSIDE         ;<A> = HEAD NUMBER (0 OR 1)
        MOV     C,A             ;SAVE FOR LATER USE AS "H" OF "CHRN"
        ADD A ! ADD A
        MOV     B,A
        LDA     CURPHYSDEV
        ANI     3               ;STRIP OFF DEVICE TYPE (LEAVE ONLY UNIT NUMBER)
        ORA     B
        RST     PUTNEC          ;SEND (UNIT, HEAD) INFO
        INX     H
        MOV     A,M             ;GET N  --  BYTES/SECTOR = (N+7)**2
        MOV     E,A             ;SAVE FOR LATER USE AS "N" OF "CHRN"
        RST     PUTNEC          ;SEND SECTOR SIZE
        INX     H
        MOV     A,M             ;GET "SC" IE NUMBER OF SECTORS
        MOV     D,A             ;STORE FOR LATER
        RST     PUTNEC          ;SEND NUMBER OF SECTORS
        LDA     SEEKTRK
        MOV     B,A
        INX     H               ;(SKIP OVER GAP LENGTH USED FOR NON-FORMAT R/W)
        INX     H
        MOV     A,M             ;GET GPL ASSOSIATED WITH FORMATTING
        RST     PUTNEC          ;SEND FORMAT GAP LENGTH

;KK     MVI     H,1             ;RECORD NUMBER FOR NEXT (FIRST) ID TO FORMAT
        MVI     A,1             ;KK DO THIS INSTEAD TO ADD RECORD BIAS
        MOV     H,C             ;KK SIDE NO. FOR SECBIAS
        CALL    SECBIAS         ;KK
        PUSH    PSW             ;KK TEMP SAVE
        MOV     A,C             ;KK ADD SIDE BIAS FOR SIDE I.D.
        CALL    SIDEBIAS
        MOV     C,A             ;KK SAVE
        POP     PSW             ;KK RESTORE SECTOR WITH BIAS APPLIED
        MOV     H,A             ;KK USE FOR STARTING SECTOR I.D.
        MVI     A,0E5H          ;FILL BYTE...(INITIATE EXECUTION MODE)...
        RST     PUTNEC          ;...IS LAST BYTE OF COMMAND STRING
;
        EI
        LDA     CURLCONT
        ANI     07FH
        OUT     LOCCONTIO       ;ENABLE WAIT GENERATION
        OUT     LOCDMAHIIO      ;SET INTO WAIT

FMTSUB1 MOV     A,B             ;CYL
        OUT     NECDATIO
        MOV     A,C             ;HEAD
        OUT     NECDATIO
        MOV     A,H             ;RECORD NUMBER
        INR     H
        OUT     NECDATIO
        MOV     A,E             ;BYTES/SECTOR CODE CALLED "N"
        OUT     NECDATIO
        DCR     D               ;SECTORS REMAINING
        JNZ     FMTSUB1         ;IF ANOTHER SECTOR TO FORMAT
        HLT                     ;WAIT FOR NEC INTERRUPT
;*******************************************************************************
```

```
;
;       FMTVFY - VERIFY READ AFTER FORMAT
;
;       ENTRY:  FMT$RELADRS = ADDRESS OF FD RELATATION ENTRY
;               FMT$NUM = FORMAT NUMBER AS GIVEN BY HOST
;               BUFSIDE = HEAD NUMBER (0 OR 1)
;
FMTVFY:
        LHLD    FMT$RELADRS     ;ADDRESS OF FDREL SLOT
        LDA     FMT$NUM         ;FORMAT NUMBER
        CALL    SETDENS         ;SETUP STUFF FOR VERIFY
        CALL    CPS             ;MORE SETUP FOR VERIFY

        LDA     SEEKTRK
        STA     BUFTRACK        ;SETUP FOR VERIFY

        MVI     B,1             ;FOR VERIFY: FIRST SECTOR TO READ
        LDA     FMT$CNTL        ;KK SEE IF WE USE SEEKSEC FOR FIRST SECTOR
        ANI     80H             ;KK (IT WAS SAVED INTO VFYSEC ABOVE)
        JZ      FMTVFY1
        LDA     VFYSEC          ;KK USE SEEKSEC FOR FIRST SECTOR
        MOV     B,A
FMTVFY1:
        LDA     CURPDTNST
        MOV     C,A             ;FOR VERIFY: LAST SECTOR TO READ
        CALL    PREAD           ;PHYSICAL READ

FMTSUB2 LDA     GENSTAT
        ADI     0FFH
        RET                     ;RETURN WITH CARRY SET IF ERROR

        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       11.  CALCLUATE PHYSICAL SECTOR FOR FLOPPY DISK
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

CPS:
        LDA     SEEKTRK
        MOV     B,A             ;KK TEMP SAVE
        LXI     H,CPS$SIDE
        MVI     M,0             ;KK SIDE 0 DEFAULT
        LDA     CURPDTFLG       ;KK GET NUMBER OF SIDES
        ANI     04H             ;KK DOUBLE SIDED BIT
        JZ      CPSK1           ;KK SINGLE SIDED
        MOV     A,B             ;KK TRACK:= TRACK DIV 2
        ORA     A               ;KK CY:= 0
        RAR
        MOV     B,A
        MVI     A,0             ;KK SIDE:= TRACK MOD 2, PICK UP CARRY
        RAL
        MOV     M,A             ;KK SAVE SIDE IN CPS$SIDE
CPSK1:
        LDA     CURPDTNBC       ;KK GET NUMBER OF CYLINDERS TO CHECK OVERFLOW
        DCR     A               ;KK TRACK IS [0..NBC-1]
        CMP     B               ;KK COMPARE WITH TRACK
        JNC     CPSK1A
        INR     A               ;KK ADJUST BACK TO NBC
        MOV     C,A
        MOV     A,B
        SUB     C               ;KK OVERFLOW, SUBTRACT CURPDTNBC
        MOV     B,A
        INR     M               ;KK SELECT SIDE 1
CPSK1A:
        MOV     A,B
        STA     CPS$TRACK
        MVI     H,0             ;KK
;       MOV     H,A             ;(ZERO FOR USE LATER)
        LDA     CURPDTNST       ;NUMBER OF PHYSICAL SECTORS PER TRACK
        MOV     B,A
        LDA     SEEKSEC         ;NOTE: STARTS AT ZERO
        MOV     L,A             ;<HL>=SEEKSEC
        MOV     D,H             ;(ZERO)
        LDA     PFIOFLAG
        ORA     A               ;READY FOR TEST FOR PHYSICAL FORMAT I/O
        LDA     CURPDTPSS       ;PHYS SECTOR SIZE (0=128, 1=256, ETC)
        JNZ     CPS$PF          ;IF PHYSICAL FORMAT MODE
        ORA     A
        JZ      CPSS0           ;IF SECTOR SIZE IS 0 (128)
        SUI     2
        JM      CPSS1           ;IF SECTOR SIZE IS 1 (256)
        JZ      CPSS2           ;IF SECTOR SIZE IS 2 (512)
                                ;ELSE...SECTOR SIZE IS 3 (1024)


CPSS3:                          ;3: 8*128
        DAD H ! DAD H ! DAD H ! DAD H ! DAD H
        MOV     A,H
        MOV     H,D             ;(ZERO)
        DAD H ! DAD H ! DAD H
        LXI     D,1024          ;SECTOR SIZE
        JMP     CPS2

CPSS2:                          ;2: 4*128
        DAD H ! DAD H ! DAD H ! DAD H ! DAD H ! DAD H
        MOV     A,H
        MOV     H,D             ;(ZERO)
        DAD H ! DAD H
        LXI     D,512           ;SECTOR SIZE
        JMP     CPS2

CPSS1:                          ;1: 2*128
        DAD H ! DAD H ! DAD H ! DAD H ! DAD H ! DAD H ! DAD H
        MOV     A,H
        MOV     H,D             ;(ZERO)
        DAD H
        LXI     D,256           ;SECTOR SIZE
        JMP     CPS2

CPS$PF: MVI     L,40H           ;<HL>=0040H
        MOV     D,H             ;(ZERO) FOR FUTURE <H>=0
CPSPF1  DAD     H
        DCR     A
        JP      CPSPF1
        XCHG                    ;<DE>=PHYSICAL SECTOR LENGTH
        LDA     SEEKSEC
        JMP     CPS2

CPSS0:                          ;0: 1*128
        MOV     A,L
        MOV     H,D             ;(ZERO)
        LXI     D,128           ;SECTOR SIZE
;       ...     FALL INTO CPS2

;       AT THIS POINT:
```

```
;
;               <A> = PHYSICAL SECTORS FROM START OF TRACK (B4 INTERLEAVE)
;               <B> = NUMBER OF PHYSICAL SECTORS PER TRACK
;               <DE>= PHYSICAL SECTOR SIZE
;               <H> = # OF 128-BYTE OFFSETS FROM PHYS SECTOR START
CPS2:
        PUSH    D               ;PHYS SECTOR SIZE
        MOV     C,H             ;128-BYTE OFFSETS FROM START OF PHYS SECTOR
;KK     CMP     B
;KK     JC      CPS3            ;IF NOT ON SECOND SIDE
;KK     SUB     B               ;SECOND SIDE...
;KK     LXI     H,CPS$SIDE
;KK     INR     M               ;BUMP TO SECOND SIDE

;KK CPS3
        MVI     D,0             ;<DE> = PHYS SECTOR BEFORE INTERLEAVE
        MOV     E,A
        LHLD    CURPDTSKEW
        DAD     D               ;<HL> = ADDRESS WITHIN SKEW LIST
        MOV     A,M
        STA     CPS$SECTOR      ;DESIRED PHYSICAL SECTOR - SKEW IS DONE!!

;       NOW FORM ADDRESS WITHIN BUFFER THAT CONTAINS
;       (OR WILL CONTAIN) OUR DESIRED LOGICAL SECTOR

        POP     H               ;NUMBER OF BYTES IN SECTOR
        SHLD    CPS$PSECT       ;LEAVE AROUND FOR THE USE OF OTHERS
        XCHG
        LXI     H,BUFFER
        DCR     A
        JZ      CPS5            ;IF WE DESIRE THE FIRST PHYSICAL SECTOR
CPS4    DAD     D               ;STEP BY PHYSICAL SECTOR SIZE
        DCR     A
        JNZ     CPS4
                        ;<HL>   ;KEEP AT IT UNTIL WE COUNT ALL SECTORS
CPS5    MOV     A,C             ;START OF SECTOR ADDRESS (? SAVE IT ?)
        RAR                     ;NUMBER OF 128-BYTE OFFSETS FROM <HL>
        MOV     D,A
        MVI     A,0             ;WE WANT THAT CARRY BIT
        RAR
        MOV     E,A             ;<DE> = <C> * 128
        DAD     D
        SHLD    CPS$BUFADR
        RET                     ;DONE (AT LAST)
;
        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       12.  FLOPPY DISK READ/WRITE
;            READ MAINLINE
;            WRITE MAINLINE
;            PHYS READ/WRITE AND COMMON SUBROUTINE
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

PFREAD:
        MVI     A,0FFH          ;SET PHYSICAL FORMAT FLAG
        JMP     READ1

READ:
        XRA     A               ;RESET PHYSICAL FORMAT FLAG
READ1   STA     PFIOFLAG
        LDA     GENSTAT
        ORA     A
        RNZ                     ;IF PENDING ERROR CODE WE ABORT
        CALL    SRC             ;SET RETRY COUNTERS
        LDA     UNLOAD$DELAY    ;** SET TIMER TO KEEP HEADS
        STA     TIME$HLOAD      ;** LOADED IF ALLREADY LOADED
        CALL    MAKECUR         ;ASSURE SEEKDEV THE CURRENT DEVICE
        DW      MD$READ         ;IF NOT FLOPPY DISK
        CALL    CPS             ;CALCULATE PHYSICAL SECTOR
;
;       CHECK IF REQUESTED TRACK IS IN BUFFER
;
READ2:
        LHLD    CPS$TRKS        ;REQUESTED TRACK/SIDE FROM CPS
        XCHG
        LHLD    BUFTRKS         ;TRACK/SIDE IN BUFFER
        MOV     A,H
        XRA     D
        MOV     C,A
        MOV     A,L
        XRA     E
        ORA     C
        JZ      READ3           ;IF TRACK IS IN BUFFER
        CALL    FWT             ;ELSE...FORCE WRITE TAGS
        CALL    VBT             ;VOID BUFFER TAGS
        CALL    SEEK$CPS
        LHLD    CPS$TRKS        ;** MAKE REQUESTED TRACK AND SIDE
        SHLD    BUFTRKS         ;** THE CURRENT TRACK AND SIDE
        LDA     CURPDTNST       ;NUMBER OF SECTORS
        MOV     C,A
        MVI     B,1             ;(<B>, <C>) = (FIRST, LAST) SECTOR TO READ
        JMP     READ6           ;GO READ FULL TRACK
;
READ3:
        CALL    BUFCHK          ;CHECK IF REQUESTED SECTOR IS IN BUFFER
        JC      INBUFREAD       ;IF IT IS IN THE BUFFER GO TRANSFER TO HOST
;
;       REQUESTED TRACK IS "IN BUFFER" BUT SECTOR IS NOT "IN BUFFER"
;
        CALL    SEEK$CPS        ;ASSURE THAT WE ARE ON THE RIGHT TRACK
        LDA     CPS$SECTOR
        MOV     B,A             ;READ WILL START WITH THIS SECTOR
        MVI     H,BUFTAG/100H
        MOV     L,A
        DCR     L               ;<HL> POINTS TO BUFTAG BYTE FOR CPS$SECT
        LDA     CURPDTNST       ;NUMBER OF PHYSICAL SECTORS ON TRACK
        MOV     C,A
READ4   MOV     A,M             ;FETCH BUFFER TAG
        ADD     A               ;CHECK IF IN BUFFER
        JC      READ5           ;IF SECTOR IS IN BUFFER WE DONT RE-READ
        INR     L
        MOV     A,C             ;MAX SECTOR ON TRACK...
        CMP     L               ;...VSS # OF SECTOR WHOSE TAG WE JUST CHECKED
        JNZ     READ4           ;IF MORE TAGS CONTINUE LOOP
READ5   MOV     C,L
;
;       AT THIS POINT (<B>, <C>) = (FIRST, LAST) SECTOR TO READ
;
READ6:
        CALL    PREAD
        LDA     PIOERSECT       ;PHYSICAL I/O ERROR SECTOR (=0 IF NO ERROR)
        MOV     L,A
        LDA     CPS$SECTOR      ;ONLY SECTOR WE NEED RIGHT NOW
        XRA     L               ;WAS ERROR (IF ANY) ON OUR SECTOR?
```

```
        JNZ     READ2           ;NO...GET SECTOR FROM BUF - OR ISSUE SHORT READ

;       NEEDED SECTOR HAD ERROR -- TRY TO RECOVER

        CALL    RWRETRY         ;R/W RECOVERY
        DW      READ2           ;RETRY ADDRESS
        RET                     ;EXIT IF UNRECOVERED ERROR

;****** FRONT-END TO WRITE
;
PFWRITE:
        MVI     A,0FFH          ;SET PHYSICAL FORMAT MODE
        JMP     XWRITE1
;
XWRITE:
        XRA     A               ;RESET PHYSICAL FROMAT MODE
XWRITE1 STA     PFIOFLAG        ;PHYSICAL FORMAT I/O FLAG
        RST     GETHOST         ;GET TYPE OF WRITE
        STA     RWTYPE          ;0=NORMAL     (* BOTH OF THESE MUST BE      *)
                                ;1=DIRECTORY  (* HANDLED AS UPDATE-IN-PLACE *)
                                ;2=SEQUENTIAL (DONT WORRY ABOUT PARTIAL SECTOR)
        CALL    WRITE
        LDA     GENSTAT
        ORA     A
        RZ                      ;IF WRITE COMPLEATED W/O ERROR
        JMP     GETGENSTAT

;       WRITE - WRITE DATA TO DISK
;
;
;       FOR NOW, THIS CODE LOOKS STRANGELY LIKE THE CODE FROM READ.
;       LATER WE WILL MAKE IT SOMEWHAT SMARTER
;
;       STRATING TO CHANGE -- ADDED RWTYPE=2 STUFF
;
WRITE:
        LDA     GENSTAT
        ORA     A
        RNZ                     ;IF OUTSTANDING ERROR CODE WE EXIT W/O TRYING
        CALL    SRC             ;SET RETRY COUNTERS
        LDA     WRITE$DELAY     ;** SET TIMER TO ASSURE
        STA     TIME$WRITE      ;** TIMELY ENTRY TO PRW
        CALL    MAKECUR         ;ASSURE SEEKDEV THE CURRENT DEVICE
        DW      MD$WRITE        ;IF NOT FLOPPY DISK
        CALL    CPS             ;CALCULATE PHYSICAL SECTOR
;
;       CHECK IF REQUESTED TRACK IS IN BUFFER
;
WRITE2:
        LHLD    CPS$TRKS        ;REQUESTED TRACK/SIDE FROM CPS
        XCHG
        LHLD    BUFTRKS         ;TRACK/SIDE IN BUFFER
        MOV     A,H
        XRA     D
        MOV     C,A
        MOV     A,L
        XRA     E
        ORA     C
        JZ      WRITE3          ;IF TRACK IS IN BUFFER
        CALL    FWT             ;ELSE...FORCE WRITE TAGS
        CALL    VBT             ;VOID BUFFER TAGS
        CALL    SEEK$CPS
        LHLD    CPS$TRKS        ;** MAKE REQUESTED TRACK AND SIDE
        SHLD    BUFTRKS         ;** THE CURRENT TRACK AND SIDE
        LDA     RWTYPE
        CPI     2
        JZ      SEQWRITE        ;IF SEQUENTIAL WRITE, PREREAD IS NOT NESSESSARY
        LDA     CURPDTNST       ;NUMBER OF SECTORS
        MOV     C,A
        MVI     B,1             ;(<B>, <C>) = (FIRST, LAST) SECTOR TO READ
        JMP     WRITE6          ;GO READ FULL TRACK
;
WRITE3:
        CALL    BUFCHK          ;CHECK IF REQUESTED SECTOR IS IN BUFFER
        JC      INBUFWRITE      ;IF IT IS IN THE BUFFER GO GET FROM HOST
        LDA     RWTYPE
        CPI     2
        JZ      SEQWRITE        ;IF SEQUENTIAL WRITE, PREREAD IS NOT NESSESSARY
;
;       REQUESTED TRACK IS "IN BUFFER" BUT SECTOR IS NOT "IN BUFFER"
;
        CALL    SEEK$CPS        ;ASSURE THAT WE ARE ON THE RIGHT TRACK
        LDA     CPS$SECTOR
        MOV     B,A             ;READ WILL START WITH THIS SECTOR
        MVI     H,BUFTAG/100H
        MOV     L,A
        DCR     L               ;<HL> POINTS TO BUFTAG BYTE FOR CPS$SECT
        LDA     CURPDTNST       ;NUMBER OF PHYSICAL SECTORS ON TRACK
        MOV     C,A
WRITE4  MOV     A,M             ;FETCH BUFFER TAG
        ADD     A               ;CHECK IF IN BUFFER (READTAG=80H)
        JC      WRITE5          ;IF SECTOR IS IN BUFFER WE DONT RE-READ
        INR     L
        MOV     A,C             ;MAX SECTOR ON TRACK...
        CMP     L               ;...VSS # OF SECTOR WHOSE TAG WE JUST CHECKED
        JNZ     WRITE4          ;IF MORE TAGS CONTINUE LOOP
WRITE5  MOV     C,L
;
;       AT THIS POINT (<B>, <C>) = (FIRST, LAST) SECTOR TO READ
;
WRITE6:
        CALL    PREAD
        LDA     PIOERSECT       ;PHYSICAL I/O ERROR SECTOR (=0 IF NO ERROR)
        MOV     L,A
        LDA     CPS$SECTOR      ;ONLY SECTOR WE NEED RIGHT NOW
        XRA     L               ;WAS ERROR (IF ANY) ON OUR SECTOR?
        JNZ     WRITE2          ;NO...GET SECTOR FROM BUF - OR ISSUE SHORT READ
;
;       NEEDED SECTOR HAD ERROR -- TRY TO RECOVER
;
        CALL    RWRETRY         ;R/W RECOVERY
        DW      WRITE2          ;RETRY ADDRESS
        RET                     ;EXIT IF UNRECOVERED ERROR

;*****************************************************************************
;
;       PREAD - PHYSICAL READ
;
;       ENTRY:  <B> FIRST SECTOR TO READ
;               <C> LAST SECTOR TO READ
;
;       EXIT:   SECTORS THAT HAVE BEEN READ ARE MARKED VALID IN BUFTAG
;
PREAD:
        LHLD    BUFTRKS
        CALL    PRW             ;PHYSICAL READ/WRITE
        DB      READDATA        ;NEC READ OPCODE
        DB      80H             ;USED IF ERROR TO INDICATE ERROR ON READ
```

```
        JZ      PREAD0          ;KK DO PROGRAM I/O FOR MINI
;
;       DCX     D               ;KK WE READ OUR FIRST BYTE BEFORE READ LOOP
        LXI     B,0             ;KK TIMER
PRL1:                           ;KK     T-STATES
        DCX     B               ;KK  5
        MOV     A,B             ;KK  5
        ORA     C               ;KK  4
        JZ      RDY$ERR         ;KK 10
;       IN      NECSTATIO       ;KK 10
        DB      RIM             ;KK     BOARD PATCH: 8085 SID/NEC DRQ
        ADD     A               ;KK  4
        JNC     PRL1            ;KK 10  NO DMA REQUEST, LOOP
                                ;-----
                                ;   48  TOTAL T-STATES, 12 MICRO SEC
;       IN      NECDATIO        ;KK OUR FIRST BYTE, GET IT QUICK
;       MOV     M,A             ;KK (THIS DIDN'T WORK, NO DMACK* )
;       INX     H
;
;       PHYS READ LOOP (MOSTLY) FROM BRENT
;
PREAD0:
        LDA     CURLCONT        ;  13  INITIAL VALUE
        ANI     07FH            ;   7  SET MSB TO ZERO
        OUT     LOCCONTIO       ;  10  ENABLE WAIT GENERATION
        OUT     LOCDMAHIIO      ;  10  SETS 8085 INTO WAIT
        LXI     B,NECDAT        ;  10  DATA ADDRESS FOR NEC DATA
                                ;----
                                ;  50  TOTAL T-STATES, 12.5 MICRO SEC

PREAD1  LDAX    B               ;WAIT FOR AND THEN GET DATA BYTE
        MOV     M,A             ;SAVE
        INX     H
        DCX     D               ;BYTE COUNTER
        MOV     A,D
        ORA     E
        JNZ     PREAD1          ;IF NOT DONE
;
;       NEXT-TO-LAST BYTE HAS BEEN READ FROM LAST SECTOR
;
        OUT     LOCDMALOIO      ;SET TC TO NEC765

        LDAX    B               ;GET THE LAST BYTE OF THE LAST SECTOR
        MOV     M,A             ;SAVE BYTE
        INX     H               ;FOR INT PROCESS TO SHOW HOW FAR WE GOT
        JMP     PWRITE2         ;TURN OFF READY SYNC AND WAIT FOR INTERRUPT

;*****************************************************************************
;
;       PWRITE - PHYSICAL WRITE
;
;       ENTRY:  <B> FIRST SECTOR TO WRITE
;               <C> LAST SECTOR TO WRITE
;
;       EXIT:   SECTORS THAT HAVE BEEN WRITTEN ARE MARKED AS
;               IF THEY HAD JUST BEEN READ IN THE BUFTAG LIST
;
PWRITE:
        LHLD    BUFTRKS
        CALL    PRW             ;PHYSICAL READ/WRITE
        DB      WRITEDATA       ;NEC WRITE OPCODE
        DB      40H             ;USED IF ERROR TO INDICATE ERROR ON WRITE
        JZ      PWRITE0         ;KK DO PROGRAM I/O FOR MINI
;
        LXI     B,0             ;KK TIMER
PWL1:                           ;KK     T-STATES
        DCX     B               ;KK  5
        MOV     A,B             ;KK  5
        ORA     C               ;KK  4
        JZ      RDY$ERR         ;KK 10
;       IN      NECSTATIO       ;KK 10
        DB      RIM             ;KK     BOARD PATCH: 8085 SID/NEC DRQ
        ADD     A               ;KK  4
        JNC     PWL1            ;KK 10  NO DMA REQUEST, LOOP
                                ;-----
                                ;   48  TOTAL T-STATES, 12 MICRO SEC
;
;       PHYS WRITE LOOP (MOSTLY) FROM BRENT
;
PWRITE0:
        LDA     CURLCONT        ;INITIAL VALUE
;       CALL    DEBUG
        ANI     07FH            ;SET MSB TO ZERO
        OUT     LOCCONTIO       ;ENABLE WAIT GENERATION
        OUT     LOCDMAHIIO      ;SETS 8085 INTO WAIT
        LXI     B,NECDAT        ;DATA ADDRESS FOR NEC DATA
        MOV     A,M             ;FIRST BYTE TO WRITE
PWRITE1 STAX    B               ;DATA BYTE ==> NEC
        INX     H
        DCX     D               ;BYTE COUNTER
        MOV     A,D
        ORA     E
        MOV     A,M             ;NEXT BYTE TO WRITE
        JNZ     PWRITE1         ;IF NOT ON LAST BYTE
;       NEXT-TO-LAST BYTE HAS BEEN WRITTEN TO LAST SECTOR
;
PWRITE2 OUT     LOCDMALOIO      ;SET TC TO NEC765
        OUT     NECDATIO        ;SEND THE LAST BYTE OF THE LAST SECTOR

PWRITE3 LDA     CURLCONT
        OUT     LOCCONTIO       ;TURN OFF READY SYNC (IE SET MSB ON)
        HLT                     ;WAIT FOR INTERRUPT

;*****************************************************************************
;
;       PRW - PHYSICAL READ/WRITE  (MOST CODE IS COMMON TO BOTH READ AND WRITE)
;
;       ENTRY:  <B> FIRST SECTOR TO R/W
;               <C> LAST SECTOR TO R/W
;               <H> HEAD NUMBER (0 OR 1)
;               <L> TRACK NUMBER
;               CALL    PRW
;               DB      READ OR WRITE OPCODE
;
PRW:
;       CALL    DEBUG
        SHLD    PRWA            ;SAVE TRACK AND HEAD NUMBERS
        CALL    LOAD            ;LOAD HEAD(S) AND RESET HEAD LOAD TIMER
        CALL    WAITNBUSY       ;ASSURE THAT HEADS ARE LOADED
        LXI     H,2
        DAD     SP
        SHLD    SPSAVE          ;SP TO RESTORE AFTER NEC INTERRUPT

        XTHL                    ;<HL> POINT TO DB FOLLOWING CALL
        LDA     CURPDTFLG
        ANI     40H             ;MFM BIT ONLY
        ORA     M               ;NEC COMMAND
        INX     H
```

```
        DI
        RST     PUTNEC          ;OUTPUT TO NEC
        MOV     A,M             ;$ PICK UP AND SAVE CODE USED TO INDICATE
        STA     OPTYPE          ;$ WHAT WE WERE DOING IF WE GET AN ERROR
        INX     H               ;** RESTORE RETURN
        XTHL                    ;** ADDRESS TO STACK
;
;       POST THE ADDRESS WE WANT TO GO TO WHEN THE READ/WRITE IS DONE
;
        LXI     H,PRW10
        SHLD    NECINTADRS
;
;       FIND ADDRESS OF WHERE TO START DATA TRANSFER
;
        LHLD    CPS$PSECT       ;NUMBER OF BYTES IN SECTOR (FROM CPS)
        XCHG                    ;<DE> = NUMBER OF BYTES IN PHYSICAL SECTOR
        LXI     H,BUFFER
        MOV     A,B             ;SECTOR NUMBER TO FIND IN BUFFER
        DCR     A
        JZ      PRW2            ;IF FIRST PHYSICAL SECTOR IS WANTED
PRW1    DAD     D               ;ADD LENGTH OF SECTOR TO (ACCUMLATED) BASE
        DCR     A
        JNZ     PRW1            ;CONTINUE LOOP UNTIL WE FIND R/W START ADDRESS
                                ;<HL> = MAIN BUFFER ADDRESS OF PHYS SECT IN <B>
                                ;<DE> = NUMBER OF BYTES IN EACH SECTOR
PRW2    PUSH    H               ;ADDRESS OF WHERE TO PUT FIRST SECTOR
;
;       FIND LENGTH OF DATA TRANSFER
;
        LXI     H,-1
        MOV     A,C             ;LAST SECTOR TO READ
        SUB     B               ;LESS FIRST SECTOR TO READ (= NUMBER OF SECT-1)
PRW3    DAD     D               ;ADD NUMBER OF BYTES IN SECTOR
        DCR     A
        JP      PRW3            ;FORM TOTAL DESIRED DATA LENGTH
        PUSH    H               ;LEN TO READ LESS 1 BYTE
;
;       SEND REST OF COMMAND STUFF TO NEC
;
        LHLD    PRWA            ;<H>=HEAD NUMBER; <L>=TRACK
        MOV     A,H             ;HEAD NUMBER (0 OR 1)
        ADD     A
        ADD     A               ;HEAD FLAG TO BE OR'D WITH DEVICE NUMBER
        MOV     E,A
        LDA     CURPHYSDEV
;       CALL    DEBUG
        ANI     3               ;** DEVICE NUMBER
        ORA     E               ;** PLUS SIDE
        RST     PUTNEC          ;** TO NEC
        MOV     A,L
        RST     PUTNEC          ;"C" (TRACK NUMBER) TO NEC
        MOV     A,H
        CALL    SIDEBIAS        ;KK ADD SIDE BIAS
        RST     PUTNEC          ;"H" HEAD (SIDE) AS IN ID FIELD
        MOV     A,B
        STA     PIOFIRSTR       ;SAVE FIRST RECORD THAT WE ARE TRYING TO R/W
        CALL    SECBIAS         ;KK ADD SECTOR OFFSET (BIAS), <H> = SIDE NO.
        RST     PUTNEC          ;"R" FIRST RECORD (SECTOR) TO R/W
        LHLD    CURPDTADRS
        LXI     D,PDTPSS
        DAD     D               ;<HL>=ADRS OF PDTPSS ("N", PHYS SECT SIZE 0-3)
        MOV     A,M
        MOV     B,A             ;SAVE "N" TO HELP FORM "DTL"
        RST     PUTNEC          ;"N" LENGTH OF SECTOR (0=128, 1=256,,,)
        MOV     A,C
        STA     PIOLASTR        ;SAVE LAST RECORD WE ARE TRYING TO R/W
;       CALL    DEBUG
        CALL    SECBIAS         ;KK ADD SECTOR OFFSET (BIAS), <H> = SIDE NO.
        RST     PUTNEC          ;"EOT" NUMBER OF RECS/TRK (LAST SECTOR)
        INX H ! INX H           ;<HL>=ADRS OF PDTGPL (R/W GAP LENGTH)
        MOV     A,M
        RST     PUTNEC          ;"GPL"
        DCR     B               ;<B> WAS "N"
        MVI     A,0FFH
        JP      PRW4            ;IF SECTOR SIZE > 128
        MVI     A,80H
PRW4    RST     PUTNEC          ;"DTL" 80H IF "N"=0 ELSE 0FFH
;
;       ALLOW INTS AFTER LAST COMMAND BYTE HAS BEEN SENT TO NEC
;
        EI
        POP     D               ;LEN TO READ LESS 1 BYTE
        POP     H               ;BUFFER ADDRESS
        LDA     CURLCONT        ;KK FLAG 8 OR 5 INCH
        ANI     20H             ;KK
        RET                     ;BACK TO TRANSFER DATA

;
;       COME HERE FROM READ ENDING INTERRUPT
;
PRW10:
        PUSH    H               ;LBA+1 READ FROM OR WRITTEN TO NEC
        LHLD    PRWA            ;KK SECUNBIAS NEEDS THIS SIDE NO. IN <H>
        LDA     RES$R           ;KK UNAPPLY BIAS IN CASE OF ERROR
        CALL    SECUNBIAS       ;KK ..OR ANYTHING
        STA     RES$R

        LDA     PIOFIRSTR       ;FIRST RECORD WE TRIED TO READ OR WRITE
        MOV     L,A
        LDA     RES$ST0
        ANI     0C0H            ;CHECK "IC", INTERRUPT CODE
        STA     PIOERSECT       ;=0 IF NORMAL COMPLEATION
        JZ      PRW12           ;IF NO ERROR
        LDA     RES$R           ;RECORD NUMBER THAT GAVE ERROR
        STA     PIOERSECT
        DCR     A               ;= LAST RECORD READ OR WRITTEN W/O ERROR
        JMP     PRW13

PRW12   LDA     PIOLASTR        ;LAST RECORD WE READ (ALL WERE READ OK)
PRW13   SUB     L               ;<A> = 1 LESS THAN NUMBER OF RECORDS READ OK
        JM      PRW15           ;IF NO RECORD WAS READ OK
        MOV     C,A
        DCR     L
        MVI     H,BUFTAG/100H
PRW14   MVI     M,READTAG       ;SET DATA VALID
        INX     H               ;NEXT TAG ADDRESS
        DCR     C
        JP      PRW14           ;IF MORE RECORDS TO SET TAGS FOR

PRW15   POP     D               ;KK RESTORE <HL> (ALMOST)
        LHLD    SPSAVE
        SPHL
        XCHG                    ;KK TOLD'JA
        RET
;
        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
;
;       13.  FLOPPY DISK READ-PHYSICAL-ID (FOR DENSITY INFO)
;            PLUS GETDPB AND SETDENS
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

;       GETDPB - GET DPB FOR CP/M
;       GETDPBEX - GET EXTENDED DPB FOR CPM 3.0
;       GETBPB - GET BPB FOR MS-DOS
;
GETBPB:
        LXI     D,LEN$BPB+256*LEN$DPBEX
        JMP     GETDPB1
;
GETDPBEX:
        LXI     D,LEN$DPBEX
        JMP     GETDPB1
;
GETDPB:
        LXI     D,LEN$DPB
GETDPB1
        PUSH    D
        XRA     A
        STA     CPS$SIDE        ;START ON HEAD ZERO
        CALL    LOCDPB          ;LOCATE DPB BY DOING READID AND FDRELTAB SEARCH
        POP     D               ;RESTORE OFFSET/LENGTH WORD
;KK
;KK CURRAMDPB IS ASSIGNED IN LOCDPB
;KK
        PUSH    H               ;KK <HL> = FDREL ADDRESS
        LHLD    CURRAMDPB       ;KK RAMDPB VALID INSTEAD?
        MOV     A,H
        ORA     L
        INX     H               ;KK ADVANCE PAST LENGTH BYTE
        JNZ     GETDPB1A        ;KK IF CURRAMDPB VALID
        XTHL                    ;KK NOT RAMDPB, RESTORE FDREL ADDRESS
GETDPB1A:
        POP     B               ;KK TRASH OTHER ADDRESS TO FIX STACK
        JNZ     GETDPB2         ;KK IF NOT FDREL ADDRESS

        MOV     A,H
        ORA     L
        RZ                      ;IF ID UNREADABLE
        MOV     C,D             ;OFFSET
        MVI     B,0             ;<BC> = OFFSET INTO DPB
        DAD     B
GETDPB2
        MVI     D,1             ;<D> SET TO XFER <E> BYTES TO HOST
        JMP     INBUFREAD3      ;GO TRANSFER TO HOST
;
;       CP/M GETDPB, GETDPBEX OR MS-DOS GETBPB
;       ENCOUNTER OTHER THAN FLOPPY DISK
;
RPINFD: POP D ! POP D           ;(WE ARE NOT GOING TO RETURN)
        LXI     H,MD$LOSB       ;ADRS OF LEN OF OS BLOCK FOR MEMORY DISK
        JNZ     RPINFD1         ;IF OS BLOCK ON MEMORY DISK IS BEING REQUESTED
                                ;ELSE...HARD DISK
        LXI     D,PDTHOSLEN     ;OFFSET TO LENGTH OF OS BLOCK
        LHLD    CURPDTADRS
        DAD     D               ;= ADDRESS OF LENGTH OF OS TABLE
RPINFD1 MOV     E,M
        INX     H               ;ADDRESS OF OS BLOCK
        JMP     GETDPB2         ;TRANSFER TO HOST

LDPB$YY MVI     E,13H           ;ERROR CODE FOR UNRECOVERED WRONG CYLINDER
        LXI     H,RETRYC2
        DCR     M
        JP      LDPB1           ;IF WE GET TO DO IT AGAIN
                                ;ELSE...EXIT WITH ERROR
LDPB$XX
        MOV     A,E             ;ERROR CODE...
        STA     GENSTAT         ;...TO GENERAL STATUS
        LXI     H,0             ;RETURN <HL>=0 FOR FAILURE
        RET


;       LOCDPB - LOCATE DPB FOR DEVICE
;
;       USES LOGICAL DEVICE IN SEEKDEV AND TRIES TO
;       READ AN ID FROM THAT DEVICE
;
;       FIRST TRIES SINGLE DENSITY (FM) AND THEN
;       IF FAILURE, TRIES DOUBLE DENSITY (MFM).
;
;       THEN SETS UP THE PHYSICAL DEVICE TABLE WITH
;       GAP LENGTHS AND SKEW TABLES
;
LOCDPB:
        LXI     H,0             ;KK DEFAULT RAMDPB = NONE
        SHLD    CURRAMDPB
        CALL    SRC             ;JUST USED TO SET RETRYC2
        CALL    FWT             ;FORCE WRITE TAGS BEFORE WE GET TO FAR ALONG
        MVI     A,0FFH
        STA     CURPHYSDEV      ;FORCE MAKE "MAKECUR" MOVE LDT/PDT (WHY??)
                                ;(PART OF THE REASON: TO VOID BUFFER TAGS)
        CALL    MAKECUR         ;MAKE "SEEKDEV" THE CURRENT UNIT
        DW      RPINFD          ;IF NOT FLOPPY DISK
;
;       ASSURE DEVICE TABLE FLAGS ARE CURRENT
;
        MVI     A,DRIVESTAT     ;NEC "SENSE DRIVE STATUS" COMMAND
        RST     PUTNEC
        LDA     CURPHYSDEV
        MOV     B,A             ;KK SAVE
        ANI     3               ;UNIT NUMBER (0-3)
        RST     PUTNEC
        RST     GETNEC          ;ST3 -- OUR USE: 40H=WR PROTECT, 08H=2-SIDED
        ANI     48H
        MOV     C,A             ;KK
        MOV     A,B             ;KK IF 5 INCH THEN FORCE 2 SIDED FLAG TRUE
        ANI     20H             ;KK THIS CURPHYSDEV
        MOV     A,C             ;KK RESTORE ST3
        JZ      LDP0            ;KK IF 8 INCH
        ORI     08H             ;KK 5 INCH, 2 SIDED
LDP0:
        RAR                     ;FLAGS FOR PDT ARE DOWN-SHIFTED ONE BIT
        MOV     C,A
        LHLD    CURPDTADRS
        INX H ! INX H           ;FORM ADDRESS OF FLAGS BYTE
        MOV     A,M
        ANI     NOT (24H+03H)   ;CLEAR "WRITE PROTECT" & "2 SIDED" FLAGS
                                ;ALSO CLEAR BITS FOR MEDIA CHANGE STATUS
        ORA     C               ;SET ABOVE FLAGS FROM ST3
        MOV     M,A             ;STORE BACK TO PDT FLAGS BYTE
        LXI     D,PDTNBH-PDTFLG ;OFFSET TO PDT NUMBER-OF-HEADS BYTE
        DAD     D
        ANI     4               ;ISOLATE "2 SIDED" BIT
```

```
        RAR ! RAR               ;SHIFT INTO LSB OF <A>
        INR     A               ;<A> = NUMBER OF HEADS (1 OR 2)...
        MOV     M,A             ;...POST TO DEVICE TABLE

;
;KK     IF DPB IN RAM THEN USE IT
;
        LHLD    CURFPYTAB       ;KK ^CURRENT-FLOPPY-TABLE-ENTRY
        MOV     A,M             ;KK 80H = TABLE HAS DPB
        RLC
        JNC     LDPB1           ;KK NO RAM DPB, SKIP
;
        RRC                     ;KK BACK INTO POSITION
        ANI     04H             ;KK DOUBLE SIDED BIT
        MOV     B,A             ;KK SAVE
        INX     H               ;KK POINT TO RELNUM
                                ;KK SETDENS WANTS THIS "RELATIONSHIP NO."
        MOV     E,M             ;KK USE FOR COUNTER TO SCAN FDRELTAB
        REPT    5
        INX     H               ;;KK POINT TO SKEW
        ENDM
        MOV     D,M             ;KK TEMP SAVE
        INX     H               ;KK POINT TO DPB LENGTH BYTE AND SAVE POINTER
        SHLD    CURRAMDPB       ;KK FOR ANYONE ELSE
        LHLD    CURPDTADRS      ;KK SAVE DOUBLE SIDED BIT IN PDT
        INX     H               ;KK FORM FLAGS ADDRESS
        INX     H
        MOV     A,M
        ANI     NOT 04H         ;KK RESET DOUBLE SIDED
        ORA     B               ;KK COMBINE DOUBLE SIDED BIT FROM FPYTAB
        MOV     M,A
;
        LDA     CURPHYSDEV      ;KK 5" OR 8"?
        LXI     H,STD8          ;KK 8" FDRELTAB
        ANI     4               ;KK 8" = 10H..13H
        JZ      LDPB0           ;KK IF 8" UNIT
        LXI     H,STD5          ;KK 5" FDRELTAB
LDPB0:
                                ;KK LDPB9 MOVES <E> TO <A> FOR SETDENS
        MOV     A,E             ;KK USE FOR COUNTER
        LXI     B,LENFDREL
LDPB0A:
        DCR     A
        JM      LDPB0B          ;KK EXIT WHEN <HL> = RELATIONSHIP ADDRESS
        DAD     B
        JMP     LDPB0A
;
LDPB0B:
        INX     H               ;KK LDPB9 DECRIMENTS THIS
        PUSH    PSW             ;KK GET READY FOR LDPB9
        PUSH    H
;
;KK COMMENTED OUT TO DISSALLOW RAMDPB'S SKEW TO DESTROY EXISTING SKEW
;
;       LHLD    CURPDTADRS      ;KK MAKE SURE NOT ZERO SKEW
;       MOV     A,E             ;KK REL NUM
;       ADI     LENPDTH
;       MOV     C,A
;     I ABOVE (NOT NEEDED)
;       DAD     B
;       MOV     A,D             ;KK GET SKEW
;       ORA     A
;       JZ      LDPB9           ;KK IF ZERO LEAVE CURRENT SKEW ALONE
;       MOV     M,A             ;KK SKEW CONTROL BYTE
        JMP     LDPB9
;KK END

LDPB1   XRA     A
        CALL    SEEK1           ;RECALLABRATE
        CALL    SEEK            ;READ ID'S FROM SEEKTRK (USER SPECIFIED TRACK)
;
;       NOW READ ID'S FROM THE DISKETTE
;
;       WE ARE GOING TO LOOP READING ID'S UNTIL WE READ THE MAXIMUM
;       ID NUMBER TWICE.  (BY DOING IT THIS WAY, WE ARE NOT DEPENDANT
;       ON HAVING THE ID'S IN ORDER ON THE TRACK.  ID'S SHOULD BE IN
;       ORDER, OR THE FULL-TRACK READ/WRITE OPERATIONS WILL TAKE
;       MUCH LONGER.  NOTE: THEY WILL CONTINUE TO WORK.)
;
        LXI     B,0000          ;<B>=SINGLE DENS (FM) ;<C>=MAX RECORD SO FAR
        CALL    RID$CPS         ;KK READ ID USING CPS$SIDE
        JZ      LDPB2A          ;IF SINGLE DENSITY IS CORRECT
        MVI     B,40H           ;<B>=DOUBLE DENSITY (MFM)
LDPB2   CALL    RID$CPS         ;READ ID
        MVI     E,12H           ;ERROR CODE FOR CANT FIND ID
        JNZ     LDPB$XX         ;IF CANT READ ID
LDPB2A:
;
;       IF WE ARE NOT ON THE CORRECT TRACK, BEGIN CORRECTIVE PROCEDURES
;
        LDA     RES$C           ;TRACK NUMBER (CYLINDER) FROM ID WE JUST READ
        LXI     H,SEEKTRK
        CMP     M
        JNZ     LDPB$YY         ;IF WE ARE ON THE WRONG CYLINDER
;
;       ASSURE THAT WE SEE THE MAX RECORD NUMBER TWICE
;
        LDA     RES$R           ;** RECORD NUMBER OF ID JUST FOUND
        CMP     C               ;** VSS MAX REC NUM FOUND SO FAR
        JC      LDPB2           ;IF JUST READ REC NUM IS LESS THAN CURRENT MAX
        MOV     C,A             ;ELSE...UPDATE CURRENT MAX
        JNZ     LDPB2           ;IF MAX HAS BEEN INCREASED
                                ;ELSE...MAX RECORD HAS BEEN READ TWICE
        LDA     CURLCONT
        ANI     20H             ;=0 IF 8" IS CURRENT
        MOV     E,A             ;** SETUP FOR
        MVI     A,8             ;** 8" UNIT
        JZ      LDPB3           ;IF 8" UNIT
        MVI     E,LOW -NFMT8    ;ELSE...
        MVI     A,5             ;...SETUP FOR 5" UNIT

LDPB3   ORA     B
        MOV     B,A             ;<D> = TYPE BYTE AS IN "FDRELTAB"

;       AT THIS POINT:
;
;       <B>     DISKETTE SIZE (8 OR 5) + 40H IF DOUBLE DENSITY
;       RES$N   SECTOR SIZE ("N" OF "CHRN")
;       <C>     NUMBER OF SECTORS PER TRACK
;       <E>     COUNTER WHICH WILL BECOME RELATIVE FORMAT NUMBER

;
;       BEGIN SCAN OF FDRELTAB
;
        LXI     H,FDRELTAB
        MOV     A,M
LDPB4   CMP     B
        JNZ     LDPB6           ;IF NOT ON CORRECT SIZE/DENSITY SECTION
```

```
        INX     H
        LDA     RES$N           ;SECTOR SIZE FROM DISK (0=128 1=256 2=512 ETC)
        CMP     M
        JNZ     LDPB5           ;IF SECTOR SIZE IS WRONG
        INX     H
        MOV     A,M             ;NUMBER OF SECTORS/TRACK FROM FDRELTAB
        CMP     C               ;VSS NUMBER FROM DISK
        DCX     H
        JZ      LDPB7           ;IF WE FOUND OUR MATCH
LDPB5   DCX     H
LDPB6   PUSH    B
        LXI     B,LENFDREL
        DAD     B               ;ADVANCE TO NEXT FLOPPY DISK RELATIONSHIP
        POP     B
        INR     E               ;BUMP RELATIONSHIP NUMBER
        MOV     A,M             ;** FIRST BYTE OF NEXT FDREL ENTRY...
        CPI     0FFH            ;** ...OR OUR END OF TABLE FLAG
        JNZ     LDPB4           ;IF NOT AT END OF TABLE
        MVI     E,14H           ;ERROR CODE = BAD LOGICAL FORMAT
        JMP     LDPB$XX         ;ENTRY NOT IN TABLE

;
;       CORRECT RELATIONSHIP HAS BEEN FOUND
;
LDPB7:
        PUSH    PSW
        PUSH    H
        MOV     A,B             ;=5 OR 8 (+40H IF MFM)
        RRC
        JNC     LDPB9           ;IF 8" DRIVE
;
;       FOR 5.25" FLOPPYS ONLY WE NEED TO READ ID FROM SIDE TWO
;       TO FIND OUT IF WE HAVE A DOUBLE-SIDED DISKETTE MOUNTED
;
        LXI     H,CPS$SIDE
        INR     M               ;SIDE: 0 ==> 1
        CALL    RID$CPS         ;READ ID (THIS TIME FROM SIDE 2)
        LHLD    CURPDTADRS
        INX H ! INX H           ;FORM ADDRESS OF FLAGS BYTE
        JNZ     LDPB8           ;IF READ ERROR (IE NOT DOUBLE SIDED)
        LDA     RES$H
        CPI     1
        JZ      LDPB9           ;IF SECOND SIDE ID HAS SECOND SIDE HEAD NUMBER
;
;       DISK IS ONLY SINGLE-SIDED -- RESET DOUBLE-SIDED BIT IN FLAGS
;
LDPB8   MOV     A,M
        ANI     NOT 4           ;CLEAR DOUBLE-SIDED BIT
        MOV     M,A
        XRA     A
        STA     GENSTAT         ;KILL POSSIBLE ERROR FROM FAILED SIDE 2 READID
LDPB9:
        LHLD    CURPDTADRS
        INX H ! INX H           ;FORM ADDRESS OF FLAGS BYTE
        MOV     A,M             ;FLOPPY DISK FLAGS BYTE
        ORI     2               ;SAY "MEDIA NOT CHANGED" (FROM LAST GETDPB)
        MOV     M,A
        POP     H
        POP     PSW
        DCX     H               ;<HL> = RELATIONSHIP ADDRESS
        MOV     A,E             ;RELATIONSHIP NUMBER
;       ...                     ;FALL INTO SETDENS

;       SETDENS - SET DENSITY RELATED INFO INTO PDT/LDT

;
;       ENTRY:  <A>  RELATIONSHIP NUMBER
;               <HL> RELATIONSHIP ADDRESS
;               USE CURRENT (FLOPPY) DRIVE
;
SETDENS:
        XCHG
        LHLD    CURPDTADRS
        PUSH    H               ;SAVE PDT ADDRESS
        ADI     LENPDTH         ;OFFSET TO SKEW BYTE = REL NUMB + LEN OF HEADER
        MOV     C,A
        MVI     B,0             ;<BC> = PDT OFFSET TO SKEW CONTROL BYTE
        DAD     B               ;<HL> = ADRS OF SKEW CONTROL BYTE
        MOV     C,M             ;SKEW CONTROL BYTE
        LXI     H,RAMDPBSKEW    ;KK RAMDPB SKEW VALID?
        MOV     A,M
        ORA     A
        JZ      SETDEN0         ;KK NO
        MOV     C,A             ;KK YES, USE IT INSTEAD
SETDEN0:
        POP     H
        INX     H
        INX     H
        DI
        MOV     A,M
        ORI     2               ;SET "READY FOR R/W" BIT
        ANI     NOT 40H         ;DROP MFM BIT
        MOV     M,A             ;RESTORE TO PDT
        LDAX    D
        ANI     40H             ;ISOLATE MFM BIT FROM RELATIONSHIP TABLE
        ORA     M
        MOV     M,A             ;MFM SET IN PDT FLAGS
        EI
        PUSH    PSW             ;SAVE PDT FLAGS FOR DOUBLE SIDED TEST
        INX     H               ;SKIP TRACK NUMBER
        INX     H
        INX     D
        LDAX    D               ;SECTOR SIZE    2**(N+7) = BYTES/SECTOR
        MOV     M,A             ;...STORE INTO PDT
        INX     H
        INX     D
        LDAX    D               ;PHYSICAL SECTORS PER TRACK
        MOV     M,A             ;...STORE INTO PDT
        MOV     B,A             ;(SAVE FOR BSKEW)
        INX     H
        INX     D
        LDAX    D               ;R/W GAP LENGTH
        MOV     M,A             ;...STORE INTO PDT
        INX     H
        INX     D               ;<DE>=ADRS OF FMT GAP LENGTH
        INX     D               ;<DE>=ADRS OF DEFAULT SKEW FACTOR
        INX     D               ;<DE>=ADRS OF SINGLE SIDED DPB
        PUSH    D
        LDAX    D               ;LOGICAL SECTORS PER SIDE (FROM SS DPB)
        MOV     M,A             ;...STORE INTO PDT
        INX     H
        PUSH    H               ;<HL>=ADRS OF PDT NUMBER OF PHYSICAL HEADS
        LHLD    CURPDTSKEW      ;ADRS OF SKEW LIST
        MOV     E,C             ;SKEW FACTOR
        MOV     D,B             ;SECTORS/TRACK
        CALL    BSKEW           ;BUILD SKEW LIST AND ...STORE INTO PDT

        POP     D               ;ADRS OF PDT NUMBER OF PHYSICAL HEADS
        POP     H               ;ADRS OF SINGLE SIDED DPB
```

```
        LXI     B,LEN$DPBEX+LEN$BPB  ;LENGTH ADDED TO <HL> IF DOUBLED SIDED
                                ;NOTE: <B>=0 IS USED IN MAKING # OF HEADS
        POP     PSW             ;PDT FLAGS
        ANI     4
        JZ      SETDEN1         ;IF SINGLE SIDED
        DAD     B               ;ADVANCE <HL> TO DOUBLE SIDED DPB
        INR     B               ;FOR NUMBER OF HEADS BYTE
SETDEN1:
        MOV     A,B
        INR     A               ;NUMBER OF HEADS
        STAX    D               ;...STORE INTO PDT
        INX     D
        LDA     CURLCONT        ;CHECK...
        ANI     20H             ;...FLOPPY SIZE
        MVI     A,TRACKS8       ;NUMBER OF TRACKS ON 8" FLOPPY
        JZ      SETDEN2         ;IF 8" FLOPPY
        MVI     A,TRACKS5       ;ELSE...NUMBER OF TRACKS ON 5" FLOPPY
SETDEN2 STAX    D               ;...STORE INTO PDT
        PUSH    H
        CALL    MOVECUR         ;MOVE NEW LDT/PDT TO CUR... AREA
        POP     H
        RET


;       RID - READ ID FROM FLOPPY DISK
;
;KK     GETS DEVICE FROM RID$DEVICE
;       GETS HEAD NUMBER FROM CURPHYSDEV
;       GETS DENSITY FROM <B> = 00H/40H FOR FM/MFM
;
;       USES <A>, <HL>
;       UNCHANGED: <BC>, <DE>
;
;       RETURNS ZERO PSW STATUS IF READID OPERATION WORKED
;
RID$CPS:
        LDA     CPS$SIDE        ;KK RID USING CPS$SIDE
        STA     RID$SIDE
RID:
        MOV     A,B
        ANI     40H             ;DENSITY BIT
        ORI     READID          ;NEC "READ ID" COMMAND
        DI
        RST     PUTNEC
;
;       POST THE ADDRESS WE WANT TO GO TO WHEN THE READID IS DONE
;
        LXI     H,RID2
        SHLD    NECINTADRS

        LDA     CURPHYSDEV      ;DEVICE CODE
        ANI     03H             ;MASK OFF ONLY DRIVE SELECT BITS
        MOV     L,A             ;TEMP SAVE
        LDA     RID$SIDE        ;KK INSTEAD OF CPS$SIDE
        ANI     1               ;(JUST TO BE SURE)
        ADD     A
        ADD     A               ;SIDE BIT TO HEAD SELECT POSITION
        ORA     L               ;DEVICE SELECT + SIDE
        RST     PUTNEC
        EI

;       HERE WE WOULD NORMALLY JUST "HLT" BUT BECAUSE OF THE FAKED
;       READY LINE ON 5" FLOPPYS, WE NEED A TIMEOUT.
;
;       THE TIMEOUT IS SET FOR 500MS (2 000 000 T-STATES).
;       IT SHOULD ONLY "GO OFF" IF READ ID IS ATTEMPTED
;       ON A NOT-READY OR NON-EXISTANT DRIVE.

        LXI     H,588           ;588=2000000/(14*256)
RID1    DCR     A
        JNZ     RID1            ;BASIC TIMEOUT LOOP IS 14 T-STATES/LOOP
        DCR     L
        JNZ     RID1
        DCR     H
        JP      RID1            ;IF STILL NOT TIMED OUT
;
;       HIT "TC" AND ALLOW NEC TO INTERRUPT US
;
;       DEBUG ***** IT STILL HUNG.
;       AND AS IF THAT AINT ENOUGH, WHEN WE INSERT A 5" DISK IT TAKES OFF OK!
;
;       NOW WHAT, BUCKY?
;
;       PROBLEM IS NEC CHIP.  ONLY WAY OUT IS TO SEND "RESET" TO THE CHIP.
;       THIS REQUIRES HARDWARE CHANGES ON THE BOARD.  CODE CHANGES WILL WAIT.
;
;       OUT     LOCDMALOIO      ;SEND "TC" TO NEC FDC
;       HLT                     ;WAIT FOR INTERRUPT
;
        CALL    RESET$NEC       ;KK
        MVI     A,40H           ;KK ABNORMAL TERMINATION CODE
        STA     RES$ST0         ;KK SIMULATE RES$ST0
;
;       RETURN FROM ID READING INTERRUPT
;
RID2:   LDA     RES$ST0         ;MAIN RESULT VECTOR BYTE
        ANI     0C0H            ;ONLY KEEP ERROR STATUS STUFF
        RET
;
;       HOST CALL TO READ I.D.
;
DORID:
        RST     GETHOST
        STA     RID$SIDE
        RST     GETHOST         ;KK DENSITY
        PUSH    PSW
        MVI     A,0FFH
        STA     CURPHYSDEV
        CALL    MAKECUR
        DW      DORIDEX9        ;KK IF NOT FLOPPY
        CALL    SEEK
        POP     B               ;KK RESTORE DENSITY TO <B> FOR RID
        CALL    RID
;       ORA     A
;DORIDEX:
;       MVI     A,0             ;KK CY= ERROR
;       RAL                     ;KK IF CY THEN <A>:= 1
        STA     GENSTAT
        RET
DORIDEX9:
        POP     B               ;KK FIX STACK
        RET

        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       14.  FLOPPY DISK FORCE WRITE (OF MODIFIED SECTORS)
```

```
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

;       XFWT - HOST CALL TO FORCE WRITE TAGS
;
;       PARAMETER:
;
;         0  NORMAL
;         1  VOID PROBLEM SECTOR, ELSE NORMAL
;         2  DROP ALL BUFFER TAGS (ALL PENDING WRITES LOST)
;
XFWT    XRA     A
        STA     GENSTAT         ;RESET GENERAL STATUS
        RST     GETHOST         ;ACCEPT PARAMETER
        ORA     A
        JZ      FWT             ;IF SIMPLE FORCE WRITE TAGS
        DCR     A
        JZ      XFWT1           ;IF TO DROP JUST ONE WRITE TAG
        DCR     A
        JZ      VBT             ;IF TO DROP ALL BUFFER TAGS
        JMP     FWT             ;ELSE BAD ... DO THE SIMPLE FORCE WRITE

XFWT1   LHLD    FWTA
        MOV     A,M             ;FETCH BUFFER TAG FOR PROBLEM SECTOR
        ANI     NOT WRITETAG
        MOV     M,A             ;RESTORE W/O "HAS BEEN MODIFIED" FLAG
;       JMP     FWT

;
;       FWT - FORCE WRITE TAGS IN FLOPPY AND HARD DISK BUFFER
;
FWT:
        LDA     GENSTAT
        ORA     A
        RNZ                     ;IF UNPROCESSED ERROR UPON ENTRY
        CALL    WDFWT
        LDA     BUFSIDE
        ANI     NOT 1           ;ONLY VALID SIDES ARE 0 AND 1
        RNZ                     ;SO...IF NOT VALID BAIL OUT
        CALL    SRC             ;SET RETRY COUNTERS
FWT0:
        LXI     H,BUFTAG
        LDA     CURPDTNST       ;NUMBER OF SECTORS PER TRACK
        MOV     C,A             ;FOR LOOP COUNTER
;
;       BEGIN BY SCANNING FOR THE FIRST WRITE TAG
;
FWT1    MOV     A,M
        CPI     READTAG+WRITETAG ;MUST HAVE "VALID" AND "WRITE" ONLY SET
        JZ      FWT2            ;IF WE HAVE A WRITE TAG
        INX     H
        DCR     C
        JNZ     FWT1            ;IF MORE TAGS TO LOOK THRU
        LXI     H,0             ;** NO ERROR -- POINT FWTA AT ROM
        SHLD    FWTA            ;** TO AVOID CHANCE OF MISTAKE
        RET                     ;DONE IF NO (REMAINING) WRITE TAGS

FWT2    SHLD    FWTA            ;ADDRESS OF FIRST WRITE TAG
                                ;  IF WE FAIL TO WRITE AND RWRETRY CANT HELP
                                ;  US THEN (THE RECORD REFERENDED BY) THIS
                                ;  TAG IS THE (FIRST) UNWRITEABLE ONE.
        INX     H
        MOV     B,L             ;<B>=FIRST RECORD TO WRITE

;       NEXT SCAN FOR LAST CONTIGEOUS WRITE TAG

FWT3    MOV     A,M
        CPI     READTAG+WRITETAG ;MUST HAVE "VALID" AND "WRITE" ONLY SET
        JNZ     FWT4            ;IF OFF END OF CONTIGEOUS WRITE TAGS
        INX     H
        DCR     C
        JNZ     FWT3            ;IF MORE TAGS TO LOOK THRU
        DCX     H
FWT4    MOV     C,L             ;<C>=LAST RECORD TO WRITE
        PUSH    B               ;SAVE FIRST AND LAST RECORD NUMBERS
        LDA     BUFTRACK
        CALL    SEEK1           ;ASSURE THAT WE ARE OVER DESIRED TRACK
        POP     B               ;RESTORE FIRST AND LAST RECORD NUMBERS
        CALL    PWRITE          ;PERFORM PHYSICAL WRITE
        LDA     PIOERSECT
        LXI     H,PIOFIRSTR
        CMP     M
        JNZ     FWT             ;IF NO ERROR OR ERROR NOT ON FIRST RECORD
        CALL    RWRETRY         ;GO DO RETRY STUFF
        DW      FWT0            ;RETRY ADDRESS
;
;       FALL THRU HERE IF UNRECOVERED ERROR
;
;       THIS IS A VERRRRRY BAD SPOT TO BE IN BECAUSE THE WRITE OPCODE
;       SENT GOOD GENERAL STATUS BACK TO THE HOST.  NOW THE HOST IS
;       DOING SOMETHING ELSE AND WE WOULD LIKE TO BRING THIS CURRENT
;       PROBLEM TO ITS ATTENTION.
;
;       OUR GENERAL ATTACK ON THE PROBLEM IS THIS --
;
;       1.  THE HOST NEEDS TO BE INVOLVED (OR AT LEAST ABLE TO BE INVOLVED).
;       2.  HOST INVOLVMENT NEEDS TO BE SIMPLE, YET NO IMPORTANT CAPABILITY
;           DARE BE LOST TO IT.
;       3.  HOST NEEDS TO BE ABLE TO RETRY "FWT" OPERATION BOTH WAYS --
;           PURE RETRY OR SKIP THE OFFENDING SECTOR AND TRY AGAIN.
;
;               ---- METHOD ----
;
;       4.  FWT SIMPLY RETURNS WHEN IT FAILS.  (GENSTAT IS LEFT NON-ZERO.)
;       5.  FWT RETURNS WITHOUT EVEN TRYING IF GENSTAT IS NON-ZERO UPON ENTRY.
;       6.  LIKEWISE, NEITHER READ NOR WRITE EVEN TRY TO OPERATE IF GENSTAT
;           IN NON-ZERO UPON ENTRY.  EACH ROUTINE MEERLY EXITS, PRESENTING
;           GENSTAT IN ITS OWN PARTICULAR WAY.
;       7.  AN EXTERNAL FWT CALL SUPPORTS FORCING ALL WRITE TAGS.
;           A MODIFIER TO THIS CALL ALLOWS GIVING UP ON THE SECTOR
;           THAT HN EXTERNAL FWT CALL SUPPORTS FORCING ALL WRITE TAGS.
;           A MODIFIER TO THIS CALL ALLOWS GIVING UP ON THE SECTOR
;           THAT HAS FAILED TO WRITE.

        RET                     ;THIS PART OF THIS MESS IS SIMPLE
;
;
;       DUMPTAGS - HOST CALL TO RETURN BUFTAGS TO HOST
;
;         RETURNS
;         B1            COUNT OF TAGS (SECTORS)
;         B2..BN-1      BUFTAGS
;
DUMPTAGS:
        LDA     CURPDTNST       ;KK USE NUMBER OF PHYSICAL..
        MOV     E,A             ;KK ..SECTOR/TRACK FOR COUNT
        RST     PUTHOST         ;KK GIVE LENGTH TO HOST
```

```
        LXI     H,BUFTAG        ;KK POINT TO SOURCE
        MVI     D,1             ;KK READY FOR INBUFREAD3
        MOV     A,E             ;KK DON'T GO IF ZERO TAGS
        ORA     A
        RZ
        JMP     INBUFREAD3      ;KK READ <E> BYTES TO HOST
;
;
        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       15.  NEC INTERRUPT SERVICE WITH ERROR CODE GENERATOR AND RESET$NEC
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;KK     WAIT FOR INTERUPT. IF NEVER OCCURS THEN ASSUME 5 INCH NOT READY
;KK     THEN RESET FDC. (FOLLWING CODE BORROWED FROM RID ROUTINE)
;FDC$WAIT:
;       HERE WE WOULD NORMALLY JUST "HLT" BUT BECAUSE OF THE FAKED
;       READY LINE ON 5" FLOPPYS, WE NEED A TIMEOUT.
;
;       THE TIMEOUT IS SET FOR 500MS (2 000 000 T-STATES).
;       IT SHOULD ONLY "GO OFF" ON A NOT-READY OR NON-EXISTANT DRIVE.
;
;       EI
;       LXI     D,588           ;588=2000000/(14*256)
;FDC$WAIT1:
;       DCR     A
;       JNZ     FDC$WAIT1       ;BASIC TIMEOUT LOOP IS 14 T-STATES/LOOP
;       DCR     E
;       JNZ     FDC$WAIT1
;       DCR     D
;       JP      FDC$WAIT1       ;IF STILL NOT TIMED OUT
;       ...                     ;FALL THROUGH TO RESET
RDY$ERR:
;
;KK     RESET NEC, POST ERROR STATUS AND RESTART
;
        PUSH    H
        PUSH    D
        PUSH    B
        CALL    CLRSTAT
        CALL    RESET$NEC
        MVI     A,48H           ;KK 1= NOT READY
        STA     RES$ST0         ;KK FAKE NOT READY ST0 STATUS
        CALL    SEC             ;KK SET ERROR CODES
        LDA     PRWA+1          ;KK HEAD NUMBER
        MOV     L,A             ;KK FOR SECBIAS
        MVI     A,1             ;KK FAKE SECTOR 1 ERROR
        CALL    SECBIAS         ;KK PRW10 UNAPPLIES THIS
        STA     RES$R           ;KK STORE BIAS SECTOR FOR PRW10
        POP     B
        POP     D
        POP     H
;       XTHL                    ;KK REPLACE RTN ADDRESS
;       LHLD    NECINTADRS      ;KK WITH SPECIFIED
;       XTHL                    ;KK PROCESS ADDRESS
;       RET
 JMP    BEGIN$0
;
;KK     CLEAR EXTENDED STATUS
;
```

```
CLRSTAT:
        LXI     H,GENSTAT
        MVI     B,9
CLRSTAT1:
        MVI     M,0
        INX     H
        DCR     B
        JNZ     CLRSTAT1
        RET

;
RESET$NEC:
        DI
        CALL    WAIT16
        MVI     A,0FFH          ;KK SET SERIAL OUTPUT BIT
        DB      SIM             ;KK TO RESET NEC (REQUIRES HARDWARE CHANGE)
        CALL    WAIT16          ;KK GIVE IT A CHANCE TO DO ITS THING
        MVI     A,SIMMSK        ;10H: ACK 7.5 (NEC), 0BH: MASK 6.5(WD) AND 5.5
        DB      SIM
        EI
;
;       FIRST DELAY 20 MS WITH INTRRUPTS ENABLED TO GIVE NEC FDC
;       A CHANCE TO DO ITS THING
;
        LXI     H,2963  ;20 MS * 4000 T-STATES PER MS / 27 T-STATES/LOOP
RESNEC1:
        DCX     H       ; 6
        MOV     A,H     ; 4
        ORA     L       ; 4
        JNZ     RESNEC1 ;13     ;IF DELAY NOT DONE
                        ;--
                        ;27 T-STATES/LOOP  20*4000/27 = 2963 LOOPS
        JMP     DOSPEC  ;ISSUE FLOPPY DISK SPECIFY COMMAND AND RETURN

;
;       NEC INTERUPT SERVICE
;
NECSERV:
        PUSH    H
        PUSH    D
        PUSH    B
        PUSH    PSW
        LDA     CURLCONT
        OUT     LOCCONTIO       ;TURN OFF READY SYNC (MSB SET ALLWAYS)
        CALL    WAIT16          ;DEBUG ADD DELAY...
        IN      NECSTATIO       ;THIS IS THE NEC MAIN STATUS REGISTER
        ANI     10H
        JZ      NECINT2         ;IF R/W COMMAND IS NOT IN PROGRESS
;
;       EXECUTION PHASE OF R/W COMMAND IS OVER
;       BEGIN RESULT PHASE BY ACCEPTING THE 7-BYTE RESULT TABLE
;
NECINT0:
        IN      NECSTATIO       ;KK SEE IF FDC READY
        ANI     20H             ;KK
        JNZ     NECINT0         ;KK
        LXI     H,RES$ST0
        MVI     C,7
NECINT1 RST     GETNEC
        MOV     M,A
        INX     H
        DCR     C
        JNZ     NECINT1         ;IF MORE RESULT PHASE TO INPUT
NECINT1A:
```

```
        CALL    SEC             ;SET ERROR CODE
        POP     PSW
        POP     B
        POP     D
        POP     H
        XTHL                    ;** REPLACE RTN ADDRESS
        LHLD    NECINTADRS      ;** WITH SPECIFIED
        XTHL                    ;** PROCESS ADDRESS
        EI
        RET

NECINT2 CALL    CRC             ;CHECK READY CHANGE

        CALL    WAIT16          ;DELAY TO ASSURE OK ON NEXT NEC OPERATION
        POP     PSW
        POP     B
        POP     D
        POP     H
        EI
        RET                     ;RETURN TO INTERRUPTED CODE

;
;       BEGIN SENSE INTERRUPT STATUS AFTER SEEK, RECAL OR READY CHANGE
;       (RECURSIVE SUBROUTINE WITH MAIN ENTRY CRC)
;
CRC0    RST     GETNEC          ;FINISH READINT WITH GET (AND TRASH) PCN
CRC:    MVI     A,READINT
        RST     PUTNEC
        RST     GETNEC          ;GET ST0
        CPI     80H
        RZ                      ;IF NO MORE UNITS REQUIRE ATTENTION
        PUSH    PSW             ;SAVE ST0
        CALL    CRC0            ;ALL READINT'S B4 ANY DRIVESTAT (RECURSIVE)
        POP     PSW             ;ST0
        CPI     0C0H
        RC                      ;IF NOT READY CHANGE (MOSTLY SEEK COMPLETE)
;
;       READY CHANGE -- CHANGE DEVICE TABLE IF DROP READY
;
        MOV     C,A             ;SAVE ST0 (THIS IS FROM READINT AFTER PUSH/POP)
        MVI     A,DRIVESTAT     ;NEC "SENSE DRIVE STATUS" COMMAND
        RST     PUTNEC
        MOV     A,C             ;FIRST BYTE OF READINT REPLY
        ANI     3               ;ISOLATE UNIT NUMBER (0-3)
        MOV     C,A             ;SAVE FOR USE IF DROP READY
        RST     PUTNEC
        RST     GETNEC          ;ST3
        ANI     20H
        RNZ                     ;IF UNIT CAME READY
;
;       UNIT DROPPED READY -- DO "ANI NOT 02" TO DEVICE TABLE FLAGS BYTE
;       (ALSO, IF 5" DRIVE THEN ALL 5" DRIVES HAVE READY LINES)
;
        MVI     B,0             ;<BC> = UNIT NUMBER
        LXI     H,PDTFINDERS
        DAD B ! DAD B           ;<HL> = ADRS OF PDT FINDER ENTRY (IF 8")
        LDA     CURLCONT
        ANI     20H
        JZ      CRC1            ;IF 8" DRIVE
        STA     RLINE5          ;MARK 5" DRIVES AS HAVING READY LINES
        MVI     C,8
        DAD     B               ;<HL> NOW SET AT 5" PDT FINDER
CRC1    MOV     C,M
        INX     H
        MOV     B,M             ;<BC> = DEVICE TABLE ADDRESS
        INX B ! INX B           ;<BC> = ADRS OF FLAGS BYTE
        LDAX    B
        ANI     NOT 02          ;MARK "DROP READY"
        STAX    B
        RET

;       SEC - SET ERROR CODE
;
;       CALLED FROM NECSERV AFTER 7-BYTE RESULT VECTOR IS READ
;
;       IT SHOULD BE NOTED THAT ERROR RECOVERY IS NOT DONE HERE.
;       ERROR RECOVERY IS DONE IN RWRETRY.  WHAT WE ARE DOING HERE
;       IS DEFINING ERROR CODES AND SETTING UP, BUT NOT DOING,
;       ERROR RECOVERY.
;
;       MOST OF THE OPERATION OF SEC AND ITS SUBROUTINE, MEC
;       IS GIVEN BELOW.
;
;     <D>       <L>       <H>     <B>   <C>     <---REGISTER USE---
;    ST 0      ST 1      ST 2    ERROR
;  76543210  76543210  76543210  CODE
;  --------  --------  --------
;  ....1...  ........  ........    01   DIE     NOT READY
;  ........  ......1.  ........    02   DIE     NOT WRITEABLE
;  ........  ........  ...1....    03   RECAL   WRONG CYLINDER (SEEK ERROR?)
;  ........  ........  ......1.    04   RECAL   BAD CYLINDER (FORMAT ERROR?)
;  ........  ........  .1......    05   SRETRY  CONTROL MARK (DEL DATA AM FOUND)
;  ........  .......1  .......0    06   SRETRY  CANT READ ANY ID ADDRESS MARKS
;  ........  .....1..  ........    07   SRETRY  CANT FIND REQUESTED ID
;  ........  .......1  .......1    08   RETRY   MISSING DATA ADDRESS MARK
;  ........  ..1.....  ..1.....    09   RETRY   CRC ERROR IN DATA
;  ........  ..1.....  ........    10   RETRY   CRC ERROR IN ID
;  ........  ...1....  ........    11   RETRY   OVER RUN (HDW/FIRMWARE ERROR?)
;  ...1....  ........  ........    12   RETRY   DRIVE DETECTED FAULT
;  ..... A L L    E L S E .....    13   RETRY   UNKNOWN ERROR FROM FD CTLR CHIP
;
;
;  <C>   RECOVERY METHOD     HOW IMPLEMENTED
;  ---   ---------------     --------------------------------
;   1    NORMAL RETRY
;   2    SEEK RETRY          RETRYC1=0
;   3    RECAL               RETRYC1=0, RETRYC3=2
;   4    DIE                 RETRYC1=0, RETRYC3=2, RETRYC2=0
;
SEC:
        LDA     RES$ST0
        MOV     D,A
        ANI     0C0H
        STA     GENSTAT         ;STORE DONE AGAIN IF THIS IS NON-ZERO
        RZ                      ;IF NO ERROR THEN WE ARE DONE
        LHLD    RES$ST1         ;<L>=ST1, <H>=ST2
        LXI     B,104H          ;<B>=1, <C>=4
        CALL    MEC             ;MAP ERROR CODE
        LDA     OPTYPE          ;READ=80H, WRITE=40H
        ORA     B               ;MERGE WITH ERROR CODE
        STA     GENSTAT
        DCR     C
        RZ                      ;IF NORMAL RETRY RECOVERY
        XRA     A
        STA     RETRYC1         ;FORCE NEXT RETRY TO SEEK NEXT
        DCR     C
        RZ                      ;IF WE WANTED SEEK RETRY
        MVI     A,2
```

```
        STA     RETRYC3         ;FORCE SEEK TYPE TO BE RECAL
        DCR     C
        RZ                      ;IF WE WANTED RECAL RETRY
        XRA     A
        STA     RETRYC2         ;ELSE WE WANT TO DIE
        RET

;       MEC - MAP ERROR CODE
;
;       UPON ENTRY--
;
;       <B> = 4   RECOVERY METHOD = DROP DEAD
;       <C> = 1   ERROR CODE
;       <D> = ST0
;       <L> = ST1
;       <H> = ST2
;
MEC:
        MOV     A,D             ;ST0
        ANI     8
        RNZ                     ;IF NOT READY (EC=1)
        INR     B               ;EC=2
        MOV     A,L             ;ST1
        ANI     2
        RNZ                     ;IF NO WRITE ENABLE (EC=2)
        INR     B               ;EC=3
        DCR     C               ;<C>=3, RECAL RETRY
        MOV     A,H             ;ST2
        ANI     10H
        RNZ                     ;IF WRONG CYL (EC=3)
        INR     B               ;EC=4
        MOV     A,H             ;ST2
        ANI     2
        RNZ                     ;IF BAD CYL (EC=4)
        INR     B               ;EC=5
        DCR     C               ;<C>=2, SEEK RETRY
        MOV     A,H             ;ST2
        ADD     A
        RM                      ;IF CONTROL MARK (DELETED DATA MARK) EC=5
        INR     B               ;EC=6
        MOV     A,H             ;ST2
        CMA
        ANA     L               ;ST1
        RRC
        RC                      ;IF CANT READ ANY ID ADDRESS MARK (EC=6)
        INR     B               ;EC=7
        MOV     A,L             ;ST1
        ANI     4
        RNZ                     ;IF CANT FIND REQUESTED ID (EC=7)
        INR     B               ;EC=8
        DCR     C               ;<C>=1, NORMAL RETRY
        MOV     A,L             ;ST1
        ANA     H               ;ST2
        RRC
        RC                      ;IF MISSING DATA ADDRESS MARK
        INR     B               ;EC=9
        MOV     A,L             ;ST1
        ANA     H               ;ST2
        ANI     20H
        RNZ                     ;IF CRC ERROR IN DATA (EC=9)
        INR     B               ;EC=10
        MOV     A,L             ;ST1
        ANI     20H
        RNZ                     ;IF CRC ERROR IN ID (EC=10)
        INR     B               ;EC=11
        MOV     A,L             ;ST1
        ANI     10H
        RNZ                     ;IF OVER RUN (EC=11)
        INR     B               ;EC=12
        MOV     A,D             ;ST0
        ANI     10H
        RNZ                     ;IF DRIVE DETECTED ERROR (EC=12)
        INR     B               ;EC=13
        RET                     ;UNKNOWN RESPONSE FROM FD CTLR CHIP
;
        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       16.  ALL HARD DISK ACCESS CODE
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$


HDPARM: CALL    FWT             ;JUST TO BE VERY SURE
        SHLD    WDCYL           ;RAM ADRS ASSURES NO MATCH ON CYLINDER
        LXI     H,HDHEADS       ;SET #HDS, PRE-COMP, HD STEP RATE, HD CTLR TYPE
        CALL    GC4
;
;       COPY #HEADS, PRECOMP AND STEP RATE TO ALL 4 HARD DISK TABLES
;
HDINIT:
        LXI     B,HDHEADS       ;SOURCE OF COPY
        LXI     H,HD00+PDTHHEADS ;FIRST COPY DESTINATION
        LXI     D,LENHDDT       ;SPACE BETWEEN DESTINATIONS
        CALL    HDPSUB          ;COPY NUMBER OF HEADS...
        CALL    HDPSUB          ;...PRECOMP CYL AND...
        CALL    HDPSUB          ;...STEP RATE
;
HDI1    IN      WDSTATIO
        ANI     80H
        JNZ     HDI1            ;IF STILL BUSY AFTER SYSTEM RESET
;       XRA     A               ;<A>=0 FROM ABOVE SO ";" IN COL 1 IS OK
        OUT     WDHITRKIO
        OUT     WDLOTRKIO

        IF HDRECAL
        MVI     A,1FH           ;RECAL WITH 7.5mS STEP RATE
        OUT     WDCMDIO
HDI2    IN      WDSTATIO
        ANI     80H
        JNZ     HDI2            ;IF RECAL IS NOT DONE
        ENDIF

        INX     H
;       MOV     A,M             ;HARD DISK STEP RATE...
        XRA     A               ;DEBUG...I DONT THINK WE NEED THIS SECTION
        ORI     70H             ;...PLUS SEEK COMMAND
        OUT     WDCMDIO         ;SEEK TO ZERO TO DEFINE STEP RATE FOR DRIVE
HDI3    IN      WDSTATIO
        ANI     80H
        JNZ     HDI3            ;IF SEEK IS NOT DONE
        EI
        RET

HDPSUB:                         ;SUBROUTINE OF HDPARM
        PUSH    H
```

```
        LDAX    B               ;FETCH PARAMETER TO MOVE TO HARD DISK DEV TABLE
        MOV M,A ! DAD D         ;...TO HD00 TABLE
        MOV M,A ! DAD D         ;...TO HD01 TABLE
        MOV M,A ! DAD D         ;...TO HD02 TABLE
        MOV     M,A             ;...TO HD03 TABLE
        POP     H
        INX     H
        INX     B
        RET

HARD$READ:
        LDA     MODE            ;IS IT DMA
        RLC                     ;IF YES WE MUST USE BUFFER FOR TRANSFER
        JC      HARD$READ1
        LDA     PFIOFLAG        ;IS IT A PHYSICAL READ
        ORA     A               ;IF YES GO DO FAST READ
        JNZ     FASTREAD        ;GO DO DIRECT IO TO WD CONTROLLER
HARD$READ1:
        CALL    WDCK            ;ARE WE STILL IN OUR CURRENT BUFFER?
        CNZ     WDPREAD         ;IF DATA NOT IN BUFFER THEN DO PHYS READ
        RC                      ;IF ERROR...
        CALL    WDBUFFCAL       ;HL=BUFF ADDRESS
        SHLD    CPS$BUFADR
        JMP     INBUFREAD       ;TRANSFER DATA TO HOST

HARD$WRITE:
        LDA     MODE            ;IS IT DMA
        RLC                     ;IF YES WE MUST USE BUFFER FOR TRANSFER
        JC      HARD$WRITE1
        LDA     PFIOFLAG        ;IS IT A PHYSICAL WRITE
        ORA     A               ;IF YES GO DO FAST WRITE
        JNZ     FASTWRITE       ;GO DO DIRECT IO TO WD CONTROLLER
HARD$WRITE1:
        CALL    WDCK            ;ARE WE STILL IN OUR CURRENT BUFFER?
        CNZ     WDPREAD         ;IF NOT SEQ WRITE THEN PRE-READ IS REQUIRED
        RC                      ;IF ERROR...
        MVI     A,0FFH          ;** MARK BUFFER AS...
        STA     WDBMOD          ;** ...HAVING BEEN MODIFIED
        CALL    WDBUFFCAL       ;HL=BUFF ADDR
        SHLD    CPS$BUFADR
        JMP     INBUFWRITE      ;FETCH WRITE DATA FROM HOST

;       WDPREAD  --  WD PHYSICAL READ
;
;       OPERATION:
;       1.  ASSURE ANY MODIFIED BUFFER IS WRITTEN BACK TO DISK
;       2.  ISSUE COMMANDS FOR READ
;       3.  WAIT FOR COMMAND TO FINISH INSIDE WD CTLR
;       4.  TAKE DATA FROM WD CTLR
;       5.  CHECK WD ERROR STATUS
;
WDPREAD:
        CALL    WDFWSET         ;FORCE WRT & SETUP WD CTLR FOR PHYS READ/WRITE
        RC                      ;IF FORCE WRITE ERROR
;
;       ENTRY HERE FROM FORMAT VERIFY
;
WDPR0   LXI     B,WDCMD         ;SETUP WD CMD/STS PORT
        MVI     A,WDREADCMD
        STAX    B               ;SEND READ COMMAND
WDPR1   LDAX    B               ;FETCH WD STATUS
        ANI     80H             ;CHECK WD "BUSY"
        JNZ     WDPR1           ;IF WD CONTROLLER IS NOT READY TO DELIVER DATA
        MOV     C,A             ;FORM DATA PORT ADDRESS IN <BC>
        LXI     H,WDBUFF
        MOV     E,C             ;ZERO LOOP COUNTER FOR 256 LOOPS (512 BYTES)

WDPR2   LDAX    B               ;DATA BYTE...
        MOV     M,A             ;...TO WDBUFF
        INX     H
        LDAX    B               ;DATA BYTE...
        MOV     M,A             ;...TO WDBUFF
        INX     H
        DCR     E
        JNZ     WDPR2           ;IF NOT DONE MOVING 512       IN      WDSTATIO
        RRC
        JC      WD$GAVE$ERROR   ;IF WD CTLR GAVE ERROR READING SECTOR
;
;       MARK SECTOR AS ACTIVE AND UNCHANGED
;
        LXI     D,SEEKDEV
        LXI     H,WDDEV
        MVI     C,5             ;BYTES TO MOVE
WDPR3   LDAX    D
        MOV     M,A
        INX     H
        INX     D
        DCR     C
        JNZ     WDPR3           ;IF MORE BYTES TO MOVE
        MOV     M,C             ;ZERO TO WDBMOD
        RET                     ;NOTE: CARRY MUST BE CLEAR

;       WDFWT  --  WD FORCE WRITE SECTOR IF SECTOR IS MODIFIED
;
;       RETURNS WITH CARRY SET IF WD ERROR
;
WDFWT:
        LDA     WDBMOD          ;WD BUFFER MODIFIED FLAG
        ORA     A
        RZ                      ;IF WRITE IS NOT NESSESSARY
WDPWT:
;
;       BUTLER WANTS TO TRY WITH THIS SECTION OUT
;
        LDA     PFIOFLAG        ;WE NEED TO SEE IF PHYSICAL IO
        ORA     A               ;IF SO THEN DO NOT DIVIDE
        LDA     WDSEC
        JNZ     WDPWT1B
        ANI     NOT 03
        RRC ! RRC
WDPWT1B OUT     WDSECTIO
;
WDPWT1A MVI     A,WDWRITECMD
        OUT     WDCMDIO         ;WRITE CMD TO WD CTLR
        LXI     B,WDDATA        ;DATA PORT ADDRESS TO <BC>
        MOV     E,C             ;ZERO LOOP COUNTER FOR 256 LOOPS (512 BYTES)
        LXI     H,WDBUFF
WDPWT1  MOV     A,M             ;DATA BYTE...
        STAX    B               ;...TO WD CTLR
        INX     H
        MOV     A,M             ;DATA BYTE...
        STAX    B               ;...TO WD CTLR
        INX     H
        DCR     E
        JNZ     WDPWT1          ;IF NOT DONE MOVING 512 BYTES
WDPWT2  IN      WDSTATIO
        RLC                     ;BUSY STATUS ==> CARRY
        JC      WDPWT2          ;IF WRITE HAS NOT COMPLEATED
```

```
        IN      WDSTATIO
        RRC
        JC      WD$GAVE$ERROR
        XRA     A
        STA     WDBMOD          ;MARK SECTOR AS = WHAT WE NOW HAVE ON DISK
        RET

FASTREAD:
        CALL    WDFWSET         ;FORCE ANY PENDING WRITE AND SETUP FOR READ
        RC                      ;IF FORCE WRITE ERROR
        MVI     A,0FFH
        STA     WDSEC           ;VOID WDBUFFER FLAG
        MVI     A,WDREADCMD
        STAX    B               ;ISSUE READ COMMAND
FASTR1A:
        LDAX    B               ;FETCH WD STATUS
        ANI     80H             ;CHECK WD "BUSY"
        JNZ     FASTR1A         ;IF WD CONTROLLER IS NOT READY TO DELIVER DATA
        STA     GENSTAT         ;CLEAR ERROR FLAG
        MOV     C,A             ;FORM DATA PORT ADDRESS IN <BC>
        LXI     H,SYSDATOUT
        MOV     E,C             ;ZERO LOOP COUNTER FOR 256 LOOPS (512 BYTES)
        MVI     D,SYSOUTSTAT

FASTR1: IN      SYSSTATIO
        ANA     D
        JNZ     FASTR1          ;IF HOST HAS NOT ACCEPTED LAST BYTE
        LDAX    B               ;DATA BYTE FROM WD CTLR...
        MOV     M,A             ;...SEND TO HOST
FASTR2: IN      SYSSTATIO
        ANA     D
        JNZ     FASTR2          ;IF HOST HAS NOT ACCEPTED LAST BYTE
        LDAX    B               ;DATA BYTE FROM WD CTLR...
        MOV     M,A             ;...SEND TO HOST
        DCR     E
        JNZ     FASTR1          ;IF NOT DONE MOVING 512 BYTES
FASTR3: IN      SYSSTATIO       ;WAIT FOR HOST TO TAKE LAST BYTE TO SEND
        ANA     D
        JNZ     FASTR3
        IN      WDSTATIO
        RRC
        JC      WD$GAVE$ERROR   ;IF WD CTLR GAVE ERROR READING SECTOR
        RET

FASTWRITE:
        CALL    WDFWSET         ;FORCE PENDING WRITES AND SETUP FOR WRITE
        RC                      ;IF FORCE WRITE ERROR
        XRA     A
        STA     GENSTAT         ;CLEAR ERROR FLAG
        RST     PUTHOST         ;SEND DUMMY ZERO TO HOST
        DCR     A               ;<A> = 0FFH
        STA     WDSEC           ;VOID WD BUFFER TAGS
        MVI     A,WDWRITECMD
        OUT     WDCMDIO         ;WRITE CMD TO WD CTLR
        LXI     B,WDDATA        ;DATA PORT ADDRESS TO <BC>
        MOV     E,C             ;ZERO LOOP COUNTER FOR 256 LOOPS (512 BYTES)
        LXI     H,SYSDATIN
FASTW1  IN      SYSSTATIO
        RAR
        JNC     FASTW1
        MOV     A,M             ;DATA BYTE FROM HOST...
        STAX    B               ;...TO WD CTLR
FASTW2  IN      SYSSTATIO       ;HAS HOST SENT US A BYTE
        RAR
        JNC     FASTW2
        MOV     A,M             ;DATA BYTE FROM HOST...
        STAX    B               ;...TO WD CTLR
        DCR     E
        JNZ     FASTW1          ;IF NOT DONE MOVING 512 BYTES
FASTW3  IN      WDSTATIO
        RLC                     ;BUSY STATUS ==> CARRY
        JC      FASTW3          ;IF WRITE HAS NOT COMPLEATED
        IN      WDSTATIO
        RRC
        JC      WD$GAVE$ERROR
        RET

HARD$FMT:
        LDA     FMT$CNTL        ;FORMAT SUB-OPCODE BYTE
        ANI     1
        JZ      HDFMT7          ;IF NOT TO FORMAT THEN CHECK FOR VERIFY
        MVI     A,HDSECS
        OUT     WDSECTCNTIO
        CALL    WDSET
        MVI     A,27            ;** NON-ECC CTLR ADDS 3 TO ** (THIS DOES NOT
        OUT     WDSECTIO        ;** GIVE GAP1 = GAP3 = 30  **  EFFECT ECC CTLR)
        MVI     A,WDFMTCMD
        OUT     WDCMDIO
;
;       BUILD SKEW TABLE FOR PHYSICAL SECTOR ORGANIZATION
;
        LXI     H,WDBUFF
        MOV     C,L             ;INIT SECTOR NUMBER TO ZERO
        MVI     D,HDSECS        ;PHYS SECTORS PER TRACK
        JMP     HDSQ2           ;ENTER LOOP...

HDSQ1   LDA     SEEKSEC         ;**
        ADD     L               ;** ADD SKEW FACTOR TO ADDRESS
        MOV     L,A             ;**
        SUB     D
        JC      HDSQ3           ;IF WE DID NOT WRAP-AROUND
        MOV     L,A             ;ELSE SET ADDRESS-LENGTH
        CMP     B
        JNZ     HDSQ3           ;IF WRAPAROUND DID NOT HIT A PREVIOUS VALUE
        INR     L               ;ELSE...
HDSQ2   MOV     B,L             ;     ...DO ADJUST
HDSQ3   MOV     M,C
        INR     C
        MOV     A,C
        SUB     D
        JNZ     HDSQ1           ;IF NOT DONE

        MOV     L,A             ;<HL> = WDBUFF **NOTE: <L> ALSO IS LOOP CNTR**
HDFMT2:
        XRA     A
        OUT     WDDATAIO        ;ZERO BYTE TO WD CTLR
        MOV     A,M
        OUT     WDDATAIO        ;SKEW BYTE TO WD CTLR
        INR     L
        JNZ     HDFMT2          ;IF MORE DATA TO OUTPUT
HDFMT3:
        IN      WDSTATIO
        RLC
        JC      HDFMT3          ;IF STILL BUSY DOING FORMAT
        IN      WDSTATIO
        RRC
        JC      WD$GAVE$ERROR
HDFMT4:
```

```
;
;       FORMAT DONE OK -- NOW FILL WDBUFF W/ E5'S AND WRITE THE TRACK
;
        LDA     FMT$CNTL
        ANI     4
        JZ      HDFMT7          ;IF NOT TO WRITE E5'S
        LXI     H,WDBUFF
        LXI     B,0E500H        ;<C>=0 FOR 256 LOOPS (FILL ALL 512 BYTES)
HDFMT5:
        MOV     M,B             ;E5 TO BUFFER
        INX     H
        MOV     M,B             ;E5 TO BUFFER
        INX     H
        DCR     C
        JNZ     HDFMT5          ;IF MORE BUFFER TO FILL
;
;       BUFFER FILLED WITH E5'S  --  NOW WRITE TO ALL SECTORS
;
;       MVI     C,0
HDFMT6  PUSH    B
        CALL    WDSET           ;SETUP TO WRITE...
        POP     B
        MOV     A,C
        OUT     WDSECTIO        ;...DEFINE SECTOR TO WRITE...
        PUSH    B
        CALL    WDPWT1A         ;...AND THEN WRITE
        POP     B
        RC                      ;RETURN IF ERROR
        INR     C               ;NEXT SECTOR NUMBER TO WRITE
        MVI     A,HDSECS
        CMP     C
        JNZ     HDFMT6          ;IF MORE SECTORS TO WRITE
;
;       PREHAPS WE WANT TO VERIFY
;
HDFMT7  LDA     FMT$CNTL
        ANI     2
        RZ                      ;IF NOT TO VERIFY
;
;       READ ALL SECTORS
;
        MVI     C,0
HDFMT8  PUSH    B
        CALL    WDSET           ;SETUP FOR READ
        POP     B
        MOV     A,C
        OUT     WDSECTIO        ;...DEFINE SECTOR TO READ...
        PUSH    B
        CALL    WDPR0           ;...AND THEN READ
        POP     B
        RC                      ;RETURN IF ERROR
        INR     C               ;NEXT SECTOR NUMBER TO VERIFY
        MVI     A,HDSECS
        CMP     C
        JNZ     HDFMT8          ;IF MORE SECTORS TO VERIFY
        RET

WDFWSET CALL    WDFWT           ;FORCE ANY PENDING WRITES
        RC                      ;IF ERROR
;       ...                     ;FALL INTO WDSET
WDSET:
;
;       HEAD AND CYLINDER ARE FORMED BY DIVIDING LOGICAL CYLINDER
;       (PLUS OFFSET) BY THE NUMBER OF HEADS.
;
;       FORMAT USED BY WD CONTROLLER:
;
;               ESSDDHHH
;                    HHH  HEAD (REMAINDER AFTER DIVIDE)
;                  DD     DRIVE 0-3 FROM CURPHYSDEV
;                 01      512-BYTE SECTORS      ** (THESE BITS COME
;               1         USE ECC IN DATA FIELD **  FROM HDCTYPE)
;
        LDA     CURPDT+PDTHPCMP
        OUT     WDERRORIO       ;DEFINE PRE-COMP VALUE

        LDA     CURPDT+PDTHDEV  ;000DD000
        LXI     H,HDCTYPE
        ORA     M               ;MERGE W/ "HARD DISK CTLR TYPE" ECC OR NON-ECC
        LHLD    CURPDT+PDTHOFFSET
        XCHG
        LHLD    SEEKTRK         ;GIVEN TRACK...
        DAD     D               ;...PLUS HARDWARE OFFSET (THIS WE DIVIDE)
        MOV     D,A             ;SAVE ESSDD000 FOR USE AFTER DIVIDE
        LDA     CURPDT+PDTHHEADS  ;NUMBER HEADS ON THIS HARD DISK...
        MOV     C,A             ;...IS DIVISOR
        CALL    DIVMQ           ;DIVIDE <HL> BY <C>
;
;       AT THIS POINT...
;
;       <HL> = PHYSICAL CYLINDER NUMBER FOR WD CTLR
;       <A>  = PHYSICAL HEAD NUMBER FOR WD CTLR
;       <D>  = ESSDD000 READY TO BE MERGED WITH HEAD NUMBER
;
        SHLD    WDLOTRK         ;SEND BOTH CYLINDER BYTES TO WD CTLR

        ORA     D               ;MERGE ESSDD000 WITH HEAD NUMBER
        OUT     WDSPECIO        ;SEND TO WD CTLR
;
;       IF PHYSICAL FORMAT I/O MODE THEN
;         WDSECT <== SEEKSEC
;       ELSE
;         WDSECT <== SEEKSEC/4
;
        LDA     PFIOFLAG
        ORA     A
        LDA     SEEKSEC
        JNZ     WDSET1          ;IF IN PHYSICAL FORMAT MODE
        ANI     NOT 3
        RRC ! RRC               ;PHYS SECTOR...
WDSET1  OUT     WDSECTIO        ;...TO WD CTLR
        LXI     B,WDCMD
        RET

WDBUFFCAL:
;
;       RETURNS ADDRESS WITHIN WDBUFF USING "SEEKSEC"
;
;       LOGICAL FRACTION OF PHYSICAL SECTOR COMES
;       FROM LOW 2 BITS OF LOGICAL SECTOR
;
;       MAPPING IS ......XY ==> XY000000
;
        LXI     H,512
        SHLD    CPS$PSECT       ;SETUP BYTES/SECTOR FOR INBUF(READ/WRITE)
        LXI     H,WDBUFF        ;**
        LDA     PFIOFLAG        ;** DO THIS STUFF FOR PHYSICAL
```

```
        ORA     A               ;** FORMAT I/O
        RNZ                     ;**
        LDA     SEEKSEC
        ANI     3
        RRC ! RRC
;
;       LASTLY, FORM ADDRESS OF LOGICAL SECTOR WITHIN WDBUFF
;
        ADD     A               ;Y0000000 & X TO CARRY
        MOV     L,A             ;LOW BYTE OF ADDRESS WITHIN WDBUFF
        RAL                     ;0000000X (AND WE ARE DONE WITH Y)
        ADD     H               ;WDBUFF/100H (EXACT--WDBUFF IS ON 100H BOUNDRY)
        MOV     H,A             ;<HL> = ADDRESS WITHIN WDBUF
        RET

;       WDCK -- CHECK IF HOST REQUEST IS WITHIN WDBUFF
;
;       RETURNS ZERO PSW STATUS IF HOST REQUEST IS WITHIN CURRENT WDBUFF
;
WDCK:
        LXI     H,SEEKDEV
        LXI     D,WDDEV
        LDAX    D               ;WDDEV
        CMP     M               ;SEEKDEV
        RNZ
        INX     D
        INX     H
        LDAX    D               ;WDCYL (LOW BYTE)       NOTE: LOGICAL CYLINDERS
        CMP     M               ;SEEKTRK (LOW BYTE)           ARE CHECKED HERE
        RNZ
        INX     D
        INX     H
        LDAX    D               ;WDCYL (HIGH BYTE)
        CMP     M               ;SEEKTRK (HIGH BYTE)
        RNZ
        INX     D               ;(SKIP WDHEAD
        INX     H               ; AND SEEKHD)
        INX     D
        INX     H
        LDA     PFIOFLAG
        ORA     A
        LDAX    D               ;WDSEC
        JNZ     WDCK1           ;IF DOING 512-BYTE TRANSFERS
        XRA     M               ;VSS SEEKSEC
        ANI     NOT 03          ;DONT CHECK THE BITS WE DONT CARE ABOUT
        RET

WDCK1   CMP     M               ;COMPARE FOR 512-BYTE SECTORS
        RET


WD$GAVE$ERROR:
        IN      WDERRORIO
        STA     GENSTAT
        RET

;*****************************************************************************
;
;       DIVMQ - DIVIDE MEDIUM QUICK (REGISTER ARGUMENTS)
;
;               SUBROUTINE IS USED TO DIVIDE TRACK NUMBER (IN SEEKTRK)
;               BY THE NUMBER OF HEADS TO GIVE PHYSICAL CYL AND HEAD
;               FOR THE HARD DISK
;
;       INPUT:  <HL> 16-BIT DIVIDEND (24-BIT IF <AHL> USED)
;               <C> 8-BIT DIVISOR
;
;       OUTPUT: <B> = 0 (USED AS LOOP COUNTER)
;               <C> 8-BIT DIVISOR (UNCHANGED)
;               <A> 8-BIT REMAINDER
;               <DE> = UNCHANGED (NOT USED)
;               <HL> 16-BIT QUOITIENT
;
;       >>>THIS ROUTINE IS QUICK BECAUSE NO CHECK IS MADE FOR VALID DIVIDE<<<
;
DIVMQ   XRA     A               ;FOR 16-BIT DIVIDEND IN <HL>
DIVMQ1                          ;ENTRY HERE FOR 24-BIT DIVIDEND IN <AHL>
;
;       DELETE THE NEXT SIX LINES AND CHANGE THE
;       LOOP COUNTER TO 16 FOR GENERAL-PURPOSE ROUTINE
;
        DAD H           ;** THE 6 "DAD H" INSTRUCTIONS HERE ASSURE
        DAD H           ;** 6 HIGH-ORDER ZERO BITS IN THE QUOTIENT
        DAD H           ;**
        DAD H ! RAL     ;** THE 3 MISSING "RAL" INSTRUCTIONS ARE OK
        DAD H ! RAL     ;** BECAUSE WE KNOW THAT THERE OUR DIVIDEND
        DAD H ! RAL     ;** HAS AT LEAST 3 HIGH-ORDER ZERO BITS

        MVI     B,10            ;LOOP COUNTER (ONLY 10 BITS IN THIS DIVIDE)
;
;       THIS DIVIDE ROUTINE WORKS FOR ALL DIVISORS FROM 1 TO 80H
;       FOR ALL DIVIDENDS LESS THAN 2000H (AT LEAST 3 HIGH-ORDER
;       ZERO BITS ARE NEEDED) AS LONG AS THE QUOTIENT IS LESS THAN 400H.
;
;       QUOTIENT LESS THAN 400H LIMITS THE DIVIDEND TO LESS THAN
;       400H * <C>.  THIS IS THE LIMITING FACTOR UNLESS <C>, THE
;       DIVISOR, IS GREATER THAN 8.
;
;       OR IN TERMS OF THE HARD DISKS THAT THIS WAS WRITTEN FOR:
;
;               MAX NUMBER OF TRACKS:    8192
;               MAX NUMBER OF CYLINDERS: 1024     IF 1 TO 8 HEADS
;                                  OR  8192/NH  IF NUMBER OF HEADS >8
;
;       EXECUTION TIME:
;
;               6 * DAD          36  (ALL TIME IN T-STATES)
;               3 * RAL          12
;               XRA + MVI        11
;                               ---
;                                59  (NON-LOOP OVERHEAD)
;
;               DAD H            60
;               RAL              40
;               CMP C            40
;               JC               85  50/50 JUMP TAKEN/JUMP NOT TAKEN
;               SUB/INR          40  50% EXECUTED
;               DCR B            40
;               JNZ              97  JUMP TAKEN 9 OUT OF THE 10 TIMES
;                               ---
;                               402  (TOTAL LOOP TIME)
;
;               59+402=461 T-STATES
;               OR ABOUT 115 uSEC AVG EXECUTION TIME AT 4MHZ T-STATE CLOCK
;
;               THE AVG TIME IS EVEN LESS THAN INDICATED BECAUSE WE
;               ARE NOT PRODUCING THE ENTIRE RANGE OF QUOTIENTS AND
;               BECAUSE THE OFTEN USED DIRECTORY IS ON A LOW-NUMBERED
```

```
;               TRACK WHICH DIVIDES SOMEWHAT FASTER.
;
DIVMQ2  DAD     H
        RAL
        CMP     C
        JC      DIVMQ3          ;IF TO DO SUBTRACT
        SUB     C
        INR     L               ;QUOTIENT BIT
DIVMQ3  DCR     B               ;LOOP COUNTER
        JNZ     DIVMQ2          ;IF MORE BITS
        RET
;
        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       17.  ALL DMA STUFF (FOR NORMAL READ/WRITE, MEM DISK AND MOVE MEMORY)
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

;       MOVMEM HOST CALL
;
;       PARAMETERS ARE:
;
;               DESTINATION, 3 BYTES (LO, HI, EX)
;               SOURCE, 3 BYTES (LO, HI, EX)
;               COUNT OF 128-BYTE BLOCKS (1 MOVES 128 BYTES, 0 MOVES 32K)
;
MOVMEM:
        LXI     H,MMDEST        ;** GET BOTH
        CALL    GC6             ;** ADDRESSES
        RST     GETHOST         ;GET LOOP COUNT
        MOV     C,A
MMLOOP  LXI     D,MMFROM        ;HOST SOURCE
        LXI     H,MISCBUF       ;USE OUR LOCAL SCRAP BUFFER
        CALL    DMACOM          ;GET READY FOR DMA XFER
        PUSH    PSW             ;CURRENT CONTENTS OF LOCCONT REGISTER
        ANI     07FH            ;SET BIT 7 LOW LEAVE OTHERS UNCHANGED
        OUT     LOCCONTIO
        MVI     A,SYSSTATRD     ;READY TO DMA READ FROM HOST
        OUT     DMASTATIO       ;FIRE UP DMA (WE HANG 'TIL DONE)
        POP     PSW
        OUT     LOCCONTIO       ;PUT BACK IN READ MODE SO WE CAN TALK TO NEC
        EI
;
MMLOOP1 LXI     D,MMDEST        ;HOST DESTINATION
        LXI     H,MISCBUF       ;USE OUR LOCAL SCRAP BUFFER
        CALL    DMACOM          ;GET READY FOR DMA XFER
        MVI     A,SYSSTATWR     ;READY TO WRITE TO HOST
        OUT     DMASTATIO       ;FIRE UP DMA (WE HANG 'TIL DONE)
        EI                      ;ALLOW NEC INTS AFTER EACH XFER
        DCR     C
        JNZ     MMLOOP          ;IF MORE BLOCKS TO XFER
        RET

DEFMEMD:
        LXI     H,MD$LEP        ;START ADDRESS OF LEN/END STUFF
        MVI     B,8             ;FOUR GROUPS OF 2 BYTES EACH
        CALL    GC              ;GET THOSE 8 BYTES
        CALL    HDT3            ;GET LEN OF OS BLOCK & THEN OS BLOCK
        LXI     H,MD$INT$FLG
        RST     GETHOST         ;FETCH REQUEST FROM HOST
        ORA     A
        RZ                      ;IF NOT TO INIT MEMORY
        JM      DMD1            ;IF TO INIT DESPITE FLAG
        MOV     A,M             ;FETCH MEM-DISK-ALLREADY-BEEN-INIT FLAG
        ORA     A
        RNZ                     ;IF NOT FIRST TIME, WE DONT KILL EXISTING MEM-D
DMD1:   MOV     M,H             ;SET MEM-... FLAG
;
;       INIT PUTS 0E5H IN ENTIRE FIRST 64K (ENTIRE DIRECTORY WOULD BE ENOUGH)
;       THIS WORKS JUST FINE FOR CP/M
;
        LXI     H,MISCBUF       ;NOTE: WE DEPEND UPON MISCBUF=XX80H
DMD2    MVI     M,0E5H
        INR     L
        JNZ     DMD2            ;IF ALL MISCBUF NOT FILLED WITH E5'S
        MOV     H,L             ;(ZERO)
        SHLD    MMDEST          ;ZERO LO&HI BYTES OF THIS 3-BYTE AREA
        LDA     MD$LEP          ;EXTENDED ADDRESS BITS OF FIRST MEM DISK AREA
        STA     MMDEST+2
        MOV     B,H             ;ZERO COUNTER FOR 256 128-BYTES BLOCKS TO FILL
        CALL    DMD3            ;FILL FIRST 32K
;       ...                     ;FALL INTO ROUTINE FOR LAST 32K
DMD3    MVI     C,1             ;SET TO MAKE MMLOOP EXIT AFTER WRITE
        CALL    MMLOOP1         ;FILL BLOCK
        DCR     B
        JNZ     DMD3            ;IF MORE BLOCKS TO WRITE
        RET
;
;       MD$FMT
;
MD$FMT: JZ      HARD$FMT        ;IF HARD DISK
;
;       THE ONLY FORMAT OPCODE MD$FMT RESPONDS TO IS THE "SHOW ME"
;       OPCODE.  SO WE DONT EVEN CHECK.
;
        LDA     MD$LOSB         ;LENGTH OF OPERATING SYSTEM BLOCK
        ADI     10              ;INIT FLG +1, LEN/END BYTES +8, LEN OS BLK +1
        MOV     E,A             ;SET FOR INBUFREAD3
        LXI     H,MD$INT$FLG    ;SEND FROM HERE
        JMP     GETDPB2         ;TRANSFER TO HOST -- <D>=1 THEN JMP INBUFREAD3
;
;
;       MD$READ & MD$WRITE
;
MD$READ:
        JZ      HARD$READ       ;IF HARD DISK
        CALL    MD$COM
        DCX D ! DCX D           ;= LXI D,MD$ADRS
        LXI     H,MISCBUF
        MVI     C,1             ;TRANSFER ONLY ONE 128-BYTE BLOCK
        CALL    DMAREAD1
        LXI     H,MISCBUF
        JMP     INBUFREAD1      ;TRANSFER TO HOST (ALLWAYS LENGTH 80H)

MD$WRITE:
        JZ      HARD$WRITE      ;IF HARD DISK
        LXI     H,MISCBUF
        CALL    INBUFWRITE1     ;GET DATA FROM HOST (LENGTH = 0080H)
        CALL    MD$COM          ;FORM ADDRESS OF MEMORY DISK BLOCK
        DCX D ! DCX D           ;= LXI D,MD$ADRS
        LXI     H,MISCBUF
        MVI     C,1             ;TRANSFER ONLY ONE 128-BYTE BLOCK
        JMP     DMAWRITE1       ;TRANSFER TO MEMORY DISK

;
```

```
;       MD$COM - SET MEMORY DISK ADDRESS INTO MD$ADRS
;
;       ERROR CHECKING (RANGE CHECKING) IS NOT DONE TO SAVE TIME
;
MD$COM: LXI     D,MD$ADRS
        LHLD    SEEKTRK
        DAD H ! DAD H ! DAD H ! DAD H ! DAD H
        LDA     SEEKSEC
        RRC
        MOV     B,A
        ANI     80H
        STAX    D               ;LOW ADDRESS TO MD$ADRS
        INX     D
        XRA     B               ;REMOVE ODD SECTOR BIT FROM SIGN POSN
        ADD     L               ;SAME AS "ORA L" IF GOOD SEC #
        STAX    D               ;HIGH ADDRESS TO MD$ADRS
        INX     D               ;<DE> READY FOR EXTENDED ADRS PART OF MD$ADRS
        MOV     A,H             ;EXT ADRS BEFORE ADD OF MEM DISK BASE
        MVI     B,4             ;NUMBER OF GROUPS OF (BASE, LAST BLOCK + 1)
        LXI     H,MD$LEP        ;ADRS OF FIRST (BASE, LAST BLOCK + 1) GROUP
MDCOM1  ADD     M               ;ADD BASE
        INX     H
        STAX    D               ;EXTENDED ADDRESS TO MD$ADRS
        SUB     M               ;LESS (LAST BLOCK + 1)
        RC                      ;IF ADRS JUST STORED IS CORRECT
        INX     H
        DCR     B
        JNZ     MDCOM1          ;IF MORE GROUPS
        RET                     ;IF OFF END WE DONT TELL

;       DMAREAD - DO 128-BYTE DMA READ FROM S100 HOST
;       DMAWRITE - DO 128-BYTE DMA WRITE TO S100 HOST
;
;       BOTH ENTRY POINTS USE THE HOST-SPECIFIED DMA ADDRESS
;       (BUT SEE NOTE ON REGISTER <DE> BELOW)
;
;       ENTRY:  <HL>  OUR OWN LOCAL RAM ADDRESS FOR USE IN TRANSFER
;                     (NOTE: THE LOW 7 BITS OF <L> ARE IGNORED,
;                      BUT SHOULD BE ZERO)
;
;               <DE>  POINTS TO LOW-ORDER BYTE OF 3-BYTE S100 ADDRESS
;                     (THIS REGISTER IS ONLY SPECIFIABLE IF CALLER USES
;                      THE DMAREAD1 OR DMAWRITE1 ENTRY POINTS)
;
;       EXIT:   JOB DONE
;               <HL> & <BC> UNCHANGED
;               <DE> IS INCREASED BY 2 (POINTS TO EX BYTE OF HOST ADDRESS)
;
;       NOTE:   THE S100 DMA ADDRESS IS ARRANGED IN OUR LOCAL RAM
;               AS "DB  LO,HI,EX"
;
;       NOTE AGAIN: DMAREAD RETURNS VIA GETGENSTAT.  DMAREAD1 RETURNS NORMALLY.
;
DMAREAD:
        LXI     D,GETGENSTAT    ;** FORCE RETURN TO SEND
        PUSH    D               ;** GENERAL STATUS
        CALL    DMACOMA
DMAREAD1:
        CALL    DMACOM          ;GET READY FOR DMA TRANSFER
        PUSH    PSW             ;CURRENT CONTENTS OF LOCCONT REGISTER
        ANI     07FH            ;SET BIT 7 LOW LEAVE OTHERS UNCHANGED
        OUT     LOCCONTIO
        MVI     A,SYSSTATRD
        OUT     DMASTATIO       ;FIRE UP DMA (WE HANG 'TIL DONE)
        DCR     C
        JNZ     DMAREAD1        ;IF MORE BLOCKS TO XFER
        POP     PSW
        OUT     LOCCONTIO       ;PUT BACK IN READ MODE SO WE CAN TALK TO NEC
        EI
        RET

DMAWRITE:
        CALL    DMACOMA
DMAWRITE1:
        CALL    DMACOM          ;GET READY FOR DMA TRANSFER
        MVI     A,SYSSTATWR
        OUT     DMASTATIO       ;FIRE UP DMA (WE HANG 'TIL DONE)
        DCR     C
        JNZ     DMAWRITE1       ;IF MORE BLOCKS TO XFER
        EI
        RET
;
;       DMACOM, DMACOMA -- COMMON CODE FOR DMAREAD & DMAWRITE
;
DMACOMA:
        MOV     C,D             ;# OF 128-BYTE BLOCKS TO XFER
        XCHG
        LDA     SEEKDMALO       ;** FETCH 24-BIT
        LHLD    SEEKDMAHI       ;** HOST ADDRESS
        SHLD    MMFROM+1        ;HOST ADDRESS TO WORK AREA
        LXI     H,MMFROM
        MOV     M,A             ;HOST ADDRESS TO WORK AREA
        XCHG
        RET

DMACOM:
        PUSH    D
        DI                      ;KEEP F.D. INTS FROM MESSING UP DMA XFER
;
;       FIRST SETUP HOST ADDRESS
;
        LDAX    D
        INX     D
        SUI     1               ;REAL SYSDMA STUFF IS 1 HIGHER THAN LOADED
        OUT     SYSDMALOIO      ;SETUP S100 DMA ADDRESS BITS 7-0
        LDAX    D
        INX     D
        SBI     0
        OUT     SYSDMAHIIO      ;SETUP S100 DMA ADDRESS BITS 15-8
        LDAX    D
        SBI     0
        OUT     SYSDMAEXIO      ;SETUP S100 DMA ADDRESS BITS 23-16
;
        CALL    MM80            ;BUMP HOST ADDRESS FOR NEXT TIME
        XCHG                    ;(MM80 DOES NOT XCHG BACK)
;
;       NOW SETUP LOCAL ADDRESS
;
        MOV     A,L
        OUT     LOCDMALOIO
        MOV     A,H
        OUT     LOCDMAHIIO
        LXI     D,0080H
        DAD     D
        POP     D               ;RESTORE ADRS OF HOST ADDRESS BYTES
        LDA     CURLCONT
        RET
```

```
MM80:   XCHG
        DCX     H
        DCX     H               ;FORM ADDDRESS OF LOW BYTE OF HOST ADDRESS
        MOV     A,M
        ADI     80H
        MOV     M,A
        RNC                     ;IF WE DONT HAVE TO DO CARRY STUFF
        INX     H
        INR     M               ;CARRY TO HI BYTE
        RNZ                     ;IF NO CARRY TO EX
        INX     H
        INR     M               ;CARRY TO EX BYTE
        RET
;
        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       18.  BLOCK TRANSFER TO/FROM HOST AND RELATED STUFF
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

;       BUFCHK - CHECK IF GIVEN PHYSICAL SECTOR IS IN MAIN BUFFER
;
;               RETURNS CARRY SET IF FOUND
;               RETURNS <HL> POINTING TO TAG BYTE
;
BUFCHK:
        LDA     CPS$SECTOR
        MOV     L,A
        DCR     L
        MVI     H,BUFTAG/100H   ;<HL> = ADDRESS OF BUFFER FLAG BYTE
        MOV     A,M
        ADD     A               ;MOVE VALID BIT TO CARRY (READTAG=80H)
        RET

;       INBUFREAD - SEND DATA FROM MAIN BUFFER AREA TO HOST
;
;               USES CPS$BUFADR
;
INBUFREAD:
        LHLD    CPS$PSECT
        DAD     H               ;<H> = # OF 128-BYTE BLOCKS TO TRANSFER
        XCHG
        LHLD    CPS$BUFADR
        LDA     PFIOFLAG
        ORA     A
        JNZ     INBUFREAD2      ;IF DOING PHYSICAL MODE IO
INBUFREAD1:
        MVI     D,01            ;ELSE...USE LOGICAL SECTOR LENGTH
INBUFREAD2:
        XRA     A
        STA     GENSTAT         ;STATUS FOR SECTOR IS GOOD OR WE DONT GET HERE
        LDA     MODE
        ADD     A
        JM      INBUFREAD3A     ;IF DOING BOOT
        JC      DMAWRITE        ;IF DMA MODE GO WRITE TO HOST
INBUFREAD3A:
        MVI     E,80H
INBUFREAD3:
        IN      SYSSTATIO
        ANI     SYSOUTSTAT
        JNZ     INBUFREAD3      ;IF HOST HAS NOT TAKEN LAST BYTE
        MOV     A,M
        OUT     SYSDATOUTIO     ;SEND DATA BYTE TO HOST
        INX     H
        DCR     E
        JNZ     INBUFREAD3      ;IF MORE BYTES TO SEND TO HOST
        DCR     D
        JNZ     INBUFREAD3A     ;IF MORE BYTES TO SEND TO HOST
INBUFREAD4:
        IN      SYSSTATIO
        ANI     SYSOUTSTAT
        JNZ     INBUFREAD4      ;IF HOST HAS NOT YET TAKEN OUR LAST BYTE
        RET

;       INBUFWRITE - ACCEPT DATA FROM HOST FOR MAIN BUFFER AREA
;
;       UPON ENTRY <HL> POINTS TO BUFFER TAG BYTE
;
;               USES CPS$BUFADR
;
SEQWRITE: CALL  BUFCHK          ;SPECIAL ENTRY FOR SEQUENTIAL WRITE (RWTYPE=2)
INBUFWRITE:
        MOV     A,M
        ORI     WRITETAG+READTAG  ;MARK AS WRITTEN INTO
        MOV     M,A
        LHLD    CPS$PSECT       ;PHYSICAL SECTOR LENGTH
        DAD     H               ;<H> = # OF 128-BYTE BLOCKS TO TRANSFER
        XCHG
        LHLD    CPS$BUFADR
        LDA     PFIOFLAG
        ORA     A
        JNZ     INBUFWRITE2     ;IF DOING PHYSICAL MODE IO
INBUFWRITE1:
        MVI     D,01            ;ELSE...USE LOGICAL SECTOR LENGTH
INBUFWRITE2:
        XRA     A
        STA     GENSTAT         ;STATUS FOR SECTOR IS GOOD OR WE DONT GET HERE
        LDA     MODE
        ADD     A
        JM      INBUFWRITE3     ;IF DOING PUTSYS (CANT BE DMA)
        JC      DMAREAD         ;IF DMA MODE, GO GET DATA TO BE WRITTEN
INBUFWRITE3:
        XRA     A               ;SEND DUMMY GENERAL STATUS
        RST     PUTHOST         ;BEFORE TAKING DATA
INBUFWRITE4A:
        MVI     E,80H
INBUFWRITE4:
        IN      SYSSTATIO
        RAR
        JNC     INBUFWRITE4     ;IF HOST HAS NOT PRESENTED NEXT BYTE
        IN      SYSDATINIO      ;ACCEPT BYTE FROM HOST
        MOV     M,A             ;STORE INTO BUFFER
        INX     H
        DCR     E
        JNZ     INBUFWRITE4     ;IF MORE BYTES TO ACCEPT FROM HOST
        DCR     D
        JNZ     INBUFWRITE4A    ;IF MORE BYTES TO ACCEPT FROM HOST
        RET
;
;       VBT - VOID BUFFER TAGS
;
VBT:
        LXI     B,40H
        LXI     H,BUFTAG
VBT1    MOV     M,B
```

```
        INX     H
        DCR     C
        JNZ     VBT1            ;IF MORE TAGS TO VOID
        RET
;
        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       19.  SETUPS FOR PHYSICAL DISK CHANGE
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

;
;       MAKECUR - MAKE THE CURRENT LOGICAL DEV IN SEEKDEV THE CURRENT DEVICE
;       MOVECUR - UPDATES CUR LDT/PDT FROM TABLE AREA
;
MAKECUR:
        LDA     SEEKDEV
        CALL    MLP
        JC      BADLU           ;IF BAD LOGICAL UNIT
        MOV     C,A             ;PHYS DEVICE CODE
        XTHL
        CPI     18H
        JNC     C1A             ;IF THIS DEVICE IS NOT FLOPPY DISK
        INX     H               ;** ADJUST RTN ADRS PAST
        INX     H               ;** 'NOT A FLOPPY' ADDRESS
        JMP     C1B

C1A     MOV     A,M             ;> THIS DEVICE IS NOT A FLOPPY
        INX     H               ;> SO WE PICK UP THE
        MOV     H,M             ;> 'NOT A FLOPPY' ADDRESS
        MOV     L,A             ;> AND PUT INTO <HL>
C1B     XTHL                    ;RTN ADRS TO STACK
        LDA     CURPHYSDEV      ;CURRENT PHYS DEVICE
        CMP     C
        JZ      MAKECX          ;IF NO DEVICE CHANGE
        CALL    FWT             ;ELSE...FORCE WRITE TAGS
;
; WE MIGHT BE ABLE TO KEEP THE READ TAGS .... BUT FOR NOW, THEY GO
;
        LXI     H,BUFTAG
        LXI     B,40H           ;<B>=0, <C>=LENGTH OF BUFFER TAGS
MAKEC1  MOV     M,B             ;ZERO IN-BUFFER TAGS
        INX     H
        DCR     C
        JNZ     MAKEC1          ;IF NOT ALL TAGS DONE
        LDA     SEEKDEV
        CALL    MLP             ;MAP LOGICAL TO PHYSICAL
        STA     CURPHYSDEV      ;POST CURRENT PHYSICAL DEVICE
        STA     WDDEV           ;<A>.NE.SEEKDEV -- USED IF HARD DISK
        STA     BUFSIDE         ;>1 TO KEEP FWT FROM ACTING UP
        SHLD    CURPDTADRS      ;CURRENT PHYSICAL DEVICE TABLE ADDRESS
        XCHG
        SHLD    CURLDTADRS      ;CURRENT LOGICAL DEVICE TABLE ADDRESS
;
;       TEST AGAIN FOR FLOPPY...
;
;       IF FLOPPY, CHECK IF 8<==>5 CHANGE
;
        CPI     18H
        JNC     MOVECUR         ;IF NOT FLOPPY
;       ANI     4               ;NON-ZERO IF 5" FLOPPY (DONT NEED TO EXECUTE)
        RAL ! RAL ! RAL
        LXI     H,CURLCONT
        XRA     M
        ANI     20H
        JZ      MOVECUR         ;IF NO 8/5 CHANGE
        XRA     M               ;FORM ORIG CURLCONT WITH 8/5 BIT CHANGED
;       ANI     NOT 40H         ;DROP HEAD LOAD BIT (MAKES 'LOAD' SET TIMERS)
        DI                      ;DI/EI REQ'D TO AVOID INT AFTER STORE &B4 'OUT'
        MOV     M,A             ;UPDATE CURLCONT
        OUT     LOCCONTIO       ;TELL DISK ABOUT CHANGE
;
;       NOW WAIT 16 MS FOR NEC 765 TO RECOVER FROM CLOCK CHANGE
;
        LXI     B,167*16        ;DELAY TIME: 16 MS = 16 * 167 MS PER LOOP
MCDELAY DCX     B               ; 6 <== T-STATES PER LOOP
        MOV     A,B             ; 4
        ORA     C               ; 4
        JNZ     MCDELAY         ;10 IF MORE TIME TO BURN WAITING FOR OUR NEC765
;                               ---
;                                24 T-STATES/LOOP or 4000/24=167 LOOPS PER MS
        MOV     A,M
        ANI     NOT 40H         ;KK DROP HEAD LOAD BIT (MOVED FORM ABOVE)
        MOV     M,A             ;KK (NOW HEADS DON'T UNLOAD AT 5<==>8 CHANGE)
;
        EI                      ;NOW WE GET OUR 'READY CHANGE' INTS
;
;KK     CALL    DOSPEC          ;ISSUE SPECIFY COMMAND UNCONDITIONALLY AT
;KK                             ;..MOVECUR BECAUSE OF INDIVIDUAL STEP RATES
;
;       SET ALL "MUST RECALLABRATE" BITS
;       SET ALL MEDIA CHANGE CONDITIONS TO "DONT KNOW"
;
        LXI     H,PDTFINDERS
        MVI     C,8             ;LOOP COUNTER (ALL 8" + ALL 5")
MAKEC1B MOV     E,M
        INX     H
        MOV     D,M             ;<DE> = PDT ADDRESS
        INX     H
        INX D ! INX D           ;FORM ADDRESS OF PDT FLAGS BYTE
        LDAX    D
        ANI     NOT 03          ;CLEAR MEDIA CHANGE CONDITIONS
        ORI     81H             ;"MUST RECAL" BIT (80H) + "DONT KNOW" (01H)
        STAX    D
        DCR     C
        JNZ     MAKEC1B         ;IF MORE TO DO
        EI                      ;EI ONLY AFTER ALL FLAGS BYTES CHANGED
;
MOVECUR:
        CALL    DOSPEC          ;KK ISSUE SPECIFY COMMAND
;
;       MOVE HEAD OF PDT
;
        LXI     D,CURPDT
        LHLD    CURPDTADRS
        MVI     B,LENPDTH       ;LENGTH OF PDT HEADER
MAKEC3  MOV     A,M
        STAX    D               ;MOVE (FIRST PART OF) PDT TO CURPDT
        INX     H
        INX     D
        DCR     B
        JNZ     MAKEC3          ;IF MORE PDT TO MOVE
        MVI     C,NFMT8
        LDA     CURPHYSDEV
        ANI     4
```

```
        JZ      MAKEC4          ;IF WORKING ON 8" FLOPPY
        MVI     C,NFMT5
MAKEC4  DAD     B               ;FORM ADDRESS OF SKEW LIST
        SHLD    CURPDTSKEW      ;ADDRESS OF SKEW LIST IN PDT ENTRY
MAKECX
        LDA     CURPHYSDEV      ;KK
        CALL    GETFPYTAB       ;KK <HL> RETURNED WITH ^FLOPPY_TABLE_ENTRY
        SHLD    CURFPYTAB       ;KK SAVE FOR ANYONE
        LXI     D,RAMDPBSKEW
        XRA     A
        STAX    D               ;KK NUL RAMDPB SKEW
        MOV     A,M             ;KK VALID RAMDPB?
        PUSH    PSW             ;KK SAVE FOR LATER
        RAL
        JNC     MAKECX1         ;KK NO, LEAVE RAMDPBSKEW INVALID
        LXI     B,6             ;KK OFFSET TO SKEW IN FPYTAB
        DAD     B
        MOV     A,M
        STAX    D               ;KK IS VALID, SAVE
MAKECX1:
        POP     PSW             ;KK FPYFLAGS, SEE IF NBC NEEDS TO BE DOUBLED
        MOV     E,A             ;KK COPY FOR ANDing
        XRI     10H             ;KK IF (NOT DOUBLESTEP)
        RLC
        ANA     E               ;KK AND (96 TPI)
        ANI     20H
        JZ      MAKECX2
        LHLD    CURPDTADRS      ;KK THEN (CYLINDERS = CYLINDERS * 2)
        LXI     D,PDTNBC        ;KK OFFSET TO NUMBER OF CYLINDERS
        DAD     D
        MOV     A,M
        ADD     A               ;KK *2
        STA     CURPDTNBC       ;KK SAVE IN CURRENT TABLE
MAKECX2:
        LDA     CURPHYSDEV
        ANI     4               ;SET PHYS DEV TYPE INTO PSW:
                                ;FLOPPY EXIT    ALT EXIT
                                ;-----------    --------
                ;JZ             8" FD           HARD
                ;JNZ            5" FD           MEM DSK
        RET

BADLU:  MVI     A,10H           ;EC=BAD LOGICAL UNIT
        JMP     BEGINGENSTAT    ;SET GENSTAT & RESETS LOCAL STACK
;
        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       20.  LOGICAL/PHYSICAL DEVICE SUBROUTINES
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

;       MLP - MAP LOGICAL TO PHYSICAL
;
;       ENTRY:  <A>  = LOGICAL DEVICE NUMBER
;
;       EXIT:   <A>  = FIRST PHYSICAL DEVICE NUMBER
;               <BC>   UNCHANGED
;               <DE> = LOGICAL DEVICE TABLE ADDRESS
;               <HL> = FIRST PHYSICAL DEVICE TABLE ADDRESS
;
MLP:
        CPI     10H             ;CARRY SET IF LOGICAL DEVICE = 00H..0FH
        CMC
        RC                      ;IF LOGICAL DEVICE NUMBER IS INVALID
        MOV     L,A
        ADD     A               ;*2
        ADD     A               ;*4
        ADD     A               ;*8
        SUB     L               ;*7
        MOV     L,A
        MVI     H,0
        LXI     D,LOGDEVTBL
        DAD     D               ;<HL> = ADDRESS OF GIVEN LOGICAL DEVICE TABLE
        MOV     A,M             ;FIRST PHYSICAL DEVICE NUMBER
        CPI     1               ;DO WE HAVE ANY PHYSICAL DEVICE?
        RC                      ;IF LOG DEV HAS NO PHYS DEVICE, EXIT WITH CARRY
        PUSH    H
        MOV     E,A
        MVI     D,0
        LXI     H,PDTFINDERS-20H
        DAD     D
        DAD     D               ;<HL> = ADDRESS OF PHYSICAL DEVICE FINDER
        MOV     E,M
        INX     H
        MOV     D,M
        XCHG                    ;<HL> = ADDRESS OF FIRST PHYS DEV TABLE
        POP     D               ;<DE> = ADDRESS OF GIVEN LOGICAL DEVICE TABLE
        RET                     ;NOTE: CARRY IS RESET BY "DAD"

;       DLDS - DEFINE LOGICAL DEVICE SET
;
;       ACCEPTS BYTES FROM HOST (VIA THE GETHOST RST)
;
DLDS:   RST     GETHOST
        MOV     C,A             ;SAVE LOGICAL DEVICE NUMBER
        CPI     10H
        RNC                     ;IF INVALID LOGICAL DEVICE NUMBER
        CALL    KLD             ;KILL (CURRENT) LOGICAL DEVICE
        MOV     A,C
        CALL    MLP             ;MAP LOGICAL TO PHYSICAL
                                ;NOTE: MAPPING FAILS, BUT <HL>=LDT ADDRESS
DLDS1   RST     GETHOST         ;GET PHYS DEVICE FROM HOST
        MOV     B,A
        ANI     7FH             ;DROP NOT-THE-LAST-ONE FLAG
        CPI     10H
        RC                      ;IF INVALID PHYSICAL DEVICE NUMBER
        CPI     1DH             ;1D = MAX PHYS DEV + 1
        RNC                     ;IF INVALID PHYSICAL DEVICE NUMBER
        MOV     M,A             ;1ST PHYS DEVICE TO LDT -OR- FORWARD THREAD PDT
        CALL    KPD             ;KILL PHYSICAL DEVICE

        MOV     A,C             ;LOGICAL DEVICE
        CALL    MLP             ;MAP LOGICAL DEVICE TO PHYSICAL
        RC                      ;IF MLP FAILS WE MUST HAVE
                                ;JUST KILLED OUR OWN LDT WITH KPD

        MOV     A,B             ;HOST SPECIFIED PHYSICAL DEVICE CODE
        ANI     7FH             ;DROP "TO-BE-CONTINUED" BIT
        MOV     E,A
        MVI     D,0
        LXI     H,PDTFINDERS-20H
        DAD     D
        DAD     D               ;<HL> = ADDRESS OF PHYSICAL DEVICE FINDER
        MOV     E,M
        INX     H
```

```
        MOV     D,M
        XCHG                    ;<HL> = ADDRESS OF PHYSICAL DEVICE TABLE
        MOV     M,C             ;STORE LOGICAL DEVICE NUMBER INTO PDT

        CMP     B
        RZ                      ;IF WE HAVE JUST DONE THE LAST DEVICE
        INX     H               ;FORM ADDRESS OF PLACE FOR PDT FORWARD THREAD
        JMP     DLDS1           ;GO AFTER THE NEXT PHYSICAL DEVICE

;       GLDS - GET LOGICAL DEVICE SET
;
GLDS:   RST     GETHOST
        CALL    MLP             ;MAP LOGICAL DEVICE TO PHYSICAL DEVICE
        JC      GLDS2           ;IF LOGICAL DEVICE IS ILL-FORMED OR EMPTY
        MOV     B,A             ;PHYSICAL DEVICE CODE
GLDS1   INX     H
        MOV     A,M             ;NEXT PHYSICAL DEVICE
        ORA     A
        JZ      GLDS3           ;IF LAST PHYSICAL DEVICE
        MOV     C,A             ;SAVE NEXT PHYSICAL DEVICE FOR NEXT TIME
        MOV     A,B
        ORI     80H
        RST     PUTHOST
        MOV     B,C
        MOV     E,C
        MVI     D,0
        LXI     H,PDTFINDERS-20H
        DAD     D
        DAD     D               ;<HL> = ADDRESS OF PHYSICAL DEVICE FINDER
        MOV     E,M
        INX     H
        MOV     D,M
        XCHG                    ;<HL> = ADDRESS OF (NEXT) PHYS DEVICE TABLE
        JMP     GLDS1

GLDS2   MVI     B,0             ;RETURN ZERO AS "NO DEVICE" INDICATION
GLDS3   MOV     A,B             ;LAST DEVICE IN THREAD
        JMP     SENDCHAR


;       KPD - KILL PHYSICAL DEVICE
;
;       ENTRY:  <A> = PHYSICAL DEVICE NUMBER
;
;       EXIT:   ALL PHYSICAL DEVICES IN LOGICAL DEVICE SET
;               CONTAINING GIVEN PHYSICAL DEVICE ARE KILLED
;
KPD:    CPI     10H
        RC                      ;IF INVALID PHYSICAL DEVICE NUMBER
        CPI     1DH             ;1D = MAX PHYS DEV + 1
        RNC                     ;IF INVALID PHYSICAL DEVICE NUMBER
        MOV     E,A
        MVI     D,0
        LXI     H,PDTFINDERS-20H
        DAD     D
        DAD     D               ;<HL> = ADDRESS OF PHYSICAL DEVICE FINDER
        MOV     E,M
        INX     H
        MOV     D,M             ;<DE> = ADDRESS OF PHYSICAL DEVICE TABLE
        LDAX    D               ;FETCH LOGICAL DEVICE NUMBER
;       ...
;       ...     NOTE: WE NOW FALL INTO KLD
;       ...
```

```
;       KLD - KILL LOGICAL DEVICE
;
;       ENTRY:  <A> = LOGICAL DEVICE TO KILL
;
;       EXIT:   JOB DONE
;               LDT 1ST PHYSICAL DEVICE ZEROED
;               FOR EACH PHYSICAL DEVICE IN LOGICAL DEVICE THREAD
;                 PDT LOGICAL DEVICE NUMBER = 0FFH
;                 PDT "NEXT PHYS DEVICE" POINTER = 0

KLD:    CALL    MLP             ;MAP LOGICAL TO PHYSICAL
        RC                      ;IF NO PHYSICAL DEVICE
        XRA     A
        STAX    D               ;KILL LDT POINTER TO 1ST PHYSICAL DEVICE
KLD1    MVI     M,0FFH          ;KILL LOGICAL DEVICE NUMBER IN PDT
        INX     H               ;<HL> NOW POINT TO NEXT PHYS DEV IN PDT
        CMP     M
        RZ                      ;IF NO NEXT PHYS DEVICE, THEN EXIT
        MOV     E,M             ;ELSE...FORM NEXT PDT ADDRESS
        MOV     D,A             ; (ZERO)
        LXI     H,PDTFINDERS-20H
        DAD     D
        DAD     D               ;<HL> = ADDRESS OF PHYSICAL DEVICE FINDER
        MOV     E,M
        INX     H
        MOV     D,M
        XCHG                    ;<HL> = ADDRESS OF NEXT PHYS DEV TABLE
        JMP     KLD1
;
        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       21.  SET AND USE RETRY COUNTERS
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

SRC:
        LHLD    SEEKRETRY1      ;SETUP RETRY LOGIC COUNTERS
        SHLD    RETRYC1         ;SETUP RETRY LOGIC COUNTERS
        MVI     A,3             ;SETUP RETRY LOGIC COUNTERS
        STA     RETRYC3         ;SETUP RETRY LOGIC COUNTERS
        RET

;       RWRETRY - READ/WRITE RETRY
;
;       CALLED AFTER EACH READ OR WRITE OPERATION LIKE THIS:
;
;       CALL    RWRETRY
;       DW      XYZZY           ;RETRY ADDRESS
;       JZ      OK              ;IF NO ERROR
;       JNZ     FAILURE         ;IF ALL RETRY HAS FAILED
;
;       NOTE:   WHEN ERROR RETRY FAILS CONTROL GOES BACK TO IDLE LOOP
;
RWRETRY:
        XTHL
        MOV     E,M
        INX     H
        MOV     D,M             ;<DE> = RETRY ADDRESS
        INX     H
        XTHL                    ;<TOP OF STACK> = NO ERROR RETURN ADDRESS
```

```
        LDA     GENSTAT
        ORA     A
        RZ                      ;IF NO ERROR TO RECOVER
        POP     B               ;** CHANGE RETURN ADDRESS TO
        PUSH    D               ;** GO TO RETRY POINT
        LXI     H,RETRYC1
        DCR     M
        RP                      ;IF BASIC RETRY COUNTER NOT EXPIRED
        LDA     SEEKRETRY1      ;ELSE...RESTORE BASIC
        MOV     M,A             ;       RETRY COUNTER
        INX     H               ;AND THEN TRY MORE EXOTIC RETRY METHODS
        DCR     M               ;<HL>=ADRS OF RETRYC2
        JP      RWR2            ;IF TO DO SEEK RETRY...
        POP     D               ;$$ ...ELSE ALL RETRY HAS FAILED
        PUSH    B               ;$$ GO BACK TO MAIN RETURN
        RET                     ;$$ WITH PSW NON-ZERO CONDITION
;
RWR1:
        MVI     M,3             ;RESET SEEK STATE VALUE
RWR2:
        LDA     CURPDTNBC       ;NUMBER OF CYLINDERS
        MOV     C,A             ;...FOR USE LATER
        LXI     H,RETRYC3
        DCR     M
        JM      RWR1            ;IF TO RESET STATE VALUE
        MOV     A,M
;KK     LHLD    CURPDTADRS      ;KK NO,NO!
;KK     INX     H               ;KK THIS IS CALCULATED (IF 48TPI IN 96TPI
;KK     INX     H               ;KK UNIT THEN TRACK:= TRACK * 2)
;KK     INX     H               ;<HL> = ADRS OF CURRENT CYLINDER NUMBER
        DCR     A
        LXI     H,RTRY$TRACK    ;KK USE UNCALC CYLINDER FOR RETRY INSTEAD
;
;       AT THIS POINT:
;
;       IF...           THEN...
;
;       <A> = 1         SEEK UP ONE TRACK
;       <A> = 0         FORCE RECAL
;       <A> = -1        SEEK DOWN ONE TRACK
;
        JZ      SEEK1           ;IF TO DO RECAL
        ADD     M               ;1 OR -1 PLUS CUR CYL = WANTED CYL
        SUB     C               ;<C> = NUMBER OF CYL
        JNC     RWR2            ;IF NO SUCH CYLINDER
        ADD     C               ;RESTORE DESIRED CYLINDER
        JMP     SEEK1
;
        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       22.  SET SKEW (HOST CALL); BUILD FLOPPY SKEW TABLE, SET SECTOR OFFSETS
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

;       SETSKEW -- SET SKEW BYTE INTO DEVICE TABLE
;
;       HOST CALL:
;
;               08  OMNI COMMAND CODE
;               WW  LOGICAL (OR PHYSICAL) DEVICE
;               XX  SECTOR LENGTH CODE + DENSITY
;                       BYTES/SECTOR = 2**(SECTOR LENGTH CODE + 7)
;                       ie CODES 0,1,2,3 SPECIFY 128,256,512,1024 BYTES/SECTOR
;                       DENSITY: ADD 40H FOR DOUBLE DENSITY
;               YY  NUMBER OF SECTORS (OR ZERO TO ACCEPT FIRST XX MATCH)
;               ZZ  DESIRED SKEW FACTOR (OR ZERO TO RESTORE DEFAULT)
;
;               RESPONSE IS LEFT IN GENERAL STATUS --
;                 0: GOOD
;                 1: BAD DEVICE (NOT FLOPPY DISK)
;                 2: UNKNOWN FORMAT
;                 3: BAD SKEW FACTOR
;
SETSKEW:
        LXI     H,SSPU
        CALL    GC4             ;GET PARAMETERS
        MVI     C,1             ;ERROR CODE FOR BAD LOGICAL/PHYSICAL DEVICE
        LDA     SSPU
        CPI     10H
        CC      MLP             ;MAP LOGICAL TO PHYSICAL
        JC      SSEXIT          ;IF NOT A GOOD LOGICAL DEVICE
;
;       AT THIS POINT <A> = PHYSICAL DEVICE CODE
;
        CPI     18H
        JNC     SSEXIT          ;IF NOT ONE OF OUR EIGHT FLOPPY DEVICES
        MOV     E,A
        MVI     D,0             ;<DE> = PHYSICAL DEVICE CODE
        LXI     H,PDTFINDERS-20H
        DAD     D
        DAD     D               ;<HL> = ADRS OF PHYSICAL DEVICE TABLE ADDRESS
        MOV     E,M
        INX     H
        MOV     D,M             ;<DE> = PHYSICAL DEVICE TABLE ADDRESS
        LXI     H,LENPDTH
        DAD     D               ;<HL> = ADRS OF ZEROTH SKEW BYTE IN PDT
        XCHG                    ;ADDRESS OF THAT SKEW BYTE TO <DE>

;       8 OR 5 INCH?  FIND OUT AND SETUP FOR FDREL SCAN

        CPI     14H             ;(REMEMBER, <A> = PHYS DEVICE CODE)
        MVI     B,8             ;SET FOR 8" FLOPPY
        LXI     H,STD8
        JC      SS1             ;IF WE ARE WORKING ON 8"
        MVI     B,5
        LXI     H,STD5
SS1:
;
;       <HL> = FIRST RELATIONSHIP TABLE ADDRESS FOR DESIRED FD SIZE
;
;       FINISH SETUP FOR FDREL SCAN
;
        LDA     SSDL
        MOV     C,A
        ANI     40H             ;ISOLATE DENSITY
        ORA     B               ;MERGE WITH FD SIZE
        MOV     B,A             ;SAVE FOR SCAN
        MOV     A,C
        ANI     NOT 40H         ;ISOLATE SECTOR LENGTH
        MOV     C,A             ;SAVE FOR SCAN
;
;       SCAN FDREL LOOKING FOR GIVEN FLOPPY DISK FORMAT
;
SS2:
        MOV     A,M             ;FDREL DENS/SIZE
```

```
        INX     H
        CMP     B
        JNZ     SS4             ;IF WRONG DENSITY OR SIZE
        MOV     A,M             ;FDREL SECTOR LENGTH CODE, "N" LEN=(N+7)**2
        INX     H
        CMP     C
        JNZ     SS3             ;IF WRONG SECTOR LENGTH
        LDA     SSNS            ;SECTORS/TRACK FROM HOST
        ORA     A
        JZ      SS5             ;IF "DONT CARE" ON SECTORS/TRACK
        CMP     M
        JZ      SS5             ;IF NUMBER OF SECTORS ALSO MATCHES
SS3:
        DCX     H
;
;       AT THIS POINT <HL>=ADRS OF SECOND BYTE IN FDREL ENTRY
;
SS4:
        INX     D               ;BUMP TO NEXT SKEW BYTE ADDRESS
        PUSH    D
        LXI     D,LENFDREL-1
        DAD     D               ;<HL>=ADRS OF HEAD OF NEXT SLOT
        POP     D
        MOV     A,M
        ORA     A
        JP      SS2             ;IF NOT AT END OF FDRELTAB WE TRY NEXT SLOT
        MVI     C,2             ;ERROR CODE FOR "UNKNOWN DENSITY"
        JMP     SSEXIT
SS5:

;       AT THIS POINT--
;
;         <HL> = ADRS OF SECTORS/TRACK IN DESIRED FDREL SLOT
;         <DE> = ADDRESS OF DESIRED SKEW BYTE
;
        LDA     SSFACT          ;SKEW FACTOR FROM USER
        MVI     C,3             ;ERROR CODE USED IF BAD SKEW FACTOR IS GIVEN
        CMP     M
        JNC     SSEXIT          ;IF HOST GIVEN SKEW .GE. SECTORS/TRACK
        ORA     A
        JNZ     SS6             ;IF NOT TO USE DEFAULT SKEW
        INX     H
        INX     H
        INX     H               ;<HL> NOW POINTS TO DEFAULT SKEW BYTE W/I FDREL
        MOV     A,M             ;SSFACT=0 ==> USE DEFAULT SKEW
SS6:
        STAX    D               ;POST SKEW BYTE TO DEVICE TABLE
        MVI     C,0             ;"GOOD" RESPONSE
SSEXIT:
        MOV     A,C             ;ERROR CODE
        STA     GENSTAT
        RET


;       BSKEW - BUILD SKEW TABLE
;
;       ENTRY:  <D>  = NUMBER OF SECTORS
;               <E>  = SKEW FACTOR
;               <HL> = ADDRESS AT WHICH TO START BUILDING TABLE
;
;       EXIT:   <DE> UNCHANGED
;               <HL> ADVANCED PAST LAST ENTRY IN LIST
;
;       USES:   ALL
;
;       CALLS:  NONE
;
;       WORKS FOR ALL N (NUMBER OF SECTORS) 2..81H
;       WITH ALL S (SKEW FACTOR) 1..N-1
;
;       THIS CODE THEN FAILS BECAUSE OF 8-BIT ARITHMETIC
;
BSKEW:
        MOV     C,D             ;USED AS LOOP COUNTER
        XRA     A               ;1ST SECTOR # (1 ADDED AFTER STORE INTO LIST)
BSKEW1:
        MOV     B,A             ;ADJ AFTER WRAPAROUND IF ENTRY (WOULD) = <B>
BSKEW2:
        MOV     M,A             ;DROP ENTRY IN TABLE
        INR     M               ;(BECAUSE SKEW LIST IS 1..N NOT 0..N-1)
        INX     H
        DCR     C
        RZ                      ;IF ALL DONE
        ADD     E               ; +SKEW FACTOR
        CMP     D
        JM      BSKEW2          ;IF NO WRAPAROUND
        SUB     D               ; -NUMBER OF SECTORS
        CMP     B
        JNZ     BSKEW2          ;IF WRAPAROUND ADJUST IS NOT REQUIRED
;
; WRAPAROUND ADJUST SECTION IS ONLY USED IN NUMBER OF SECTORS
; AND SKEW FACTOR HAVE A COMMON DIVISOR.
;
        INR     A               ;NEXT ITEM FOR LIST...
        JMP     BSKEW1          ;...IS NEXT ADJUST POINT


;       KK HOST CALL: SETDPB
;
;               24H OMNI COMMAND CODE
;               P1  LOGICAL (OR PHYSICAL) DEVICE
;               P2  FLAGS:
;          7       6       5       4       3       2       1       0
;       +-------+-------+-------+-------:-------+-------+-------+-------+
;       |       |       |       |DOUBLE |       :       |       |       |
;       |RAM DPB|       | 96TPI | STEP  |       :       |       |       |
;       |  IS   |       | UNIT  |(48TPI :       |2 SIDED|       |       |
;       | VALID |       |(QUAD  |DISK IN:       |       |       |       |
;       |       |       | DENS) | 96TPI):       |       |       |       |
;       +-------+-------+-------+-------:-------+-------+-------+-------+
;
;               (BITS 4 AND 5 AFFECTED BY SETTPI ONLY)
;
;               P3  FDRELPOS
;               P4  SECTOR I.D. OFFSET FOR SIDE 0, SIGNED INTEGER
;               P5  SECTOR I.D. OFFSET FOR SIDE 1, SIGNED INTEGER
;               P6  SIDE I.D. OFFSET FOR SIDE 0, SIGNED INTEGER
;               P7  SIDE I.D. OFFSET FOR SIDE 1, SIGNED INTEGER
;               P8  SKEW (0= LEAVE ALONE)
;               P9  LENGTH OF DPB
;               P10..DPB
;
;       HOST CALL RESPONSE IS LEFT IN GENERAL STATUS --
;               0: GOOD
;               1: BAD DEVICE (NOT FLOPPY DISK)
;
SETRAMDPB:
        CALL    GETFPYHST       ;KK GET FLOPPY DEVICE FROM HOST
```

```
        PUSH    PSW             ;KK IF BAD DEVICE THEN CY=1, SAVE
        CALL    GETFPYTAB       ;KK <HL> RETURNED WITH ^FPYTAB(<A>)
        POP     PSW             ;KK WAS IT A BAD DEVICE?
        PUSH    PSW
        JNC     SETRAMDPB1      ;KK NO
        LXI     H,MISCBUF       ;KK YES, TRASH INPUT
SETRAMDPB1:
        RST     GETHOST         ;KK GET FLAGS
        ANI     NOT 30H         ;KK DON'T DESTROY TPI STATUS
        MOV     B,A
        MOV     A,M             ;KK OLD FLAGS
        ANI     30H
        ORA     B
        MOV     M,A
        INX     H
        MVI     B,6             ;KK FDRELPOS,SECTOFF(2),SIDEOFF(2),SKEW
        CALL    GC
        RST     GETHOST         ;KK DPB SIZE
        CPI     21              ;KK MAX DPB SIZE +1
        JNC     SETDPBEX0       ;KK THIS COULD HANG BUT BETTER THAN CRASH
        ORA     A
        JZ      SETDPBEX0       ;KK IF ZERO LENGTH DPB
        MOV     M,A             ;KK SAVE LENGTH IN FPY TABLE
        INX     H
        MOV     B,A             ;KK COUNT FOR GC
        CALL    GC              ;KK GET DPB
SETDPBEX0:
        POP     PSW             ;KK ERROR = (CY = 1)
SETDPBEX:
        MVI     A,0
        RAL                     ;KK IF CY THEN <A>:= 1
        STA     GENSTAT
        RET

;
;KK     PERFORM OPPOSITE OF ABOVE
;
GETRAMDPB:
        CALL    GETFPYHST       ;KK GET FLOPPY DEVICE FROM HOST
        JC      SETDPBEX        ;KK IF BAD DEVICE
        CALL    GETFPYTAB       ;KK <HL> RETURNED WITH ^FPYTAB(<A>)
        LXI     D,107H          ;KK FLAGS,FDRELPOS,SECTOFF(2),SIDEOFF(2),SKEW
        CALL    INBUFREAD3      ;KK SEND (E) BYTES TO HOST
        MOV     A,M             ;KK DPB SIZE
        RST     PUTHOST
        ORA     A               ;KK IF ZERO LENGTH
        JZ      SETDPBEX        ;KK IF ZERO LENGTH DPB
        MOV     E,A             ;KK USE FOR COUNTER
        INX     H               ;KK PAST DPB LENGTH
        MVI     D,1
        CALL    INBUFREAD3      ;KK SEND DPB
        ORA     A               ;KK CY:= NO ERROR
        JMP     SETDPBEX

;KK
;KK SET TPI STATUS FROM HOST (48 OR 96 TPI, 48TPI DISK IN 96TPI UNIT)
;KK STATUS IS IN FPYTAB FLAGS BYTE
;KK
SETTPI:
        CALL    GETFPYHST       ;KK GET FLOPPY DEVICE FROM HOST
        PUSH    PSW             ;KK IF BAD DEVICE THEN CY=1, SAVE
        CALL    GETFPYTAB       ;KK <HL> RETURNED WITH ^FPYTAB(<A>)
        MOV     A,M             ;KK MASK TPI STATUS BITS
        ANI     NOT 30H
        MOV     B,A             ;KK SAVE
        RST     GETHOST         ;KK GET NEW STATUS
        ANI     30H             ;KK WE WANT ONLY THESE
        ORA     B               ;KK COMBINE
        MOV     B,A
        POP     PSW             ;KK WAS IT A BAD DEVICE?
        JC      SETDPBEX        ;KK EXIT WITHOUT UPDATING
        MOV     M,B
        JMP     SETDPBEX

;KK
;KK GET TPI STATUS FROM HOST (48 OR 96 TPI, 48TPI DISK IN 96TPI UNIT)
;KK STATUS IS IN FPYTAB FLAGS BYTE (REVERSE OF ABOVE)
;KK
GETTPI:
        CALL    GETFPYHST       ;KK GET FLOPPY DEVICE FROM HOST
        PUSH    PSW             ;KK IF BAD DEVICE THEN CY=1, SAVE
        CALL    GETFPYTAB       ;KK <HL> RETURNED WITH ^FPYTAB(<A>)
        MOV     A,M
        RST     PUTHOST         ;KK GIVE STATUS TO HOST
        POP     PSW
        JMP     SETDPBEX

;KK
;KK GET FLOPPY UNIT NUMBER FROM HOST, MAP TO PHYSICAL, CY= BAD DEVICE
;KK
GETFPYHST:
        RST     GETHOST         ;KK LOGICAL OR PHYSICAL DEVICE
        CPI     10H
        CC      MLP             ;MAP LOGICAL TO PHYSICAL
        RC                      ;KK IF NOT A GOOD LOGICAL DEVICE
        CPI     18H
        CMC
;       RC                      ;IF NOT ONE OF OUR EIGHT FLOPPY DEVICES
        RET                     ;AT THIS POINT <A> = PHYSICAL DEVICE CODE

;KK
;KK GET ADDRESS OF <A> DISK IN FPYTAB TO <HL>
;KK
GETFPYTAB:
        LXI     H,FPYTAB        ;POINT TO SELECTED TABLE ENTRY
        LXI     D,FPYTABSIZ     ;SIZE OF ENTRIES
GFT1:
        ANI     7               ;CONVERT TO OFFSET TO FPYTAB, SET FLAGS IN LOOP
        RZ
        DAD     D
        DCR     A
        JMP     GFT1
;
; <HL> = OUR TABLE ENTRY
;
        RET

;KK
;KK      ADD SECTOR BIAS FROM CURFPYTAB THAT WAS SET BY MAKECUR
;KK      ENTRY:  <H> = HEAD NO (0 OR 1)
;KK              <A> = SECTOR TO ADD BIAS TO
;KK
SECBIAS:
        PUSH    H
        CALL    GETBIAS
        ADD     M
        POP     H
        RET
```

```
;KK
;KK     REVERSE OF ABOVE
;KK
SECUNBIAS:
        PUSH    H
        CALL    GETBIAS
        SUB     M
        POP     H
        RET

;KK
;KK     ADD SIDE BIAS SAME WAY AS ABOVE
;KK
SIDEBIAS:
        PUSH    H
        CALL    GETBIAS
        INX     H               ;KK SIDEBIAS 2 BYTES PAST SECTOR BIAS
        INX     H               ;KK ..IN FPYTAB
        ADD     M
        POP     H
        RET

GETBIAS:
        PUSH    D
        PUSH    PSW
        MOV     A,H             ;KK ADD SIDE (0 OR 1)
        ANI     1               ;KK JUST MAKE SURE
        MOV     E,A
        MVI     D,0
        LHLD    CURFPYTAB
        INX     H
        INX     H               ;KK THIS IS OFFSET FOR SIDE 0
        DAD     D               ;KK ADD SIDE NO. (0 OR 1)
        POP     PSW
        POP     D               ;KK <HL> POINTS TO BIAS
        RET
;KK
;KK     RETURN OMNI FIRMWARE VERSION NUMBER, HOST CALL
;KK
GETVERS:
        MVI     A,VERSION
        RST     PUTHOST
        RET

        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       23.  MACROS TO HELP BUILD FDREL; FDREL; HARD DISK DPB
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$


;       EQUATES USED BY FOLLOWING MACROS
;
;       (DEFINED WITH "SET" TO KEEP THEM OUT OF ".SYM" FILE --
;        THE VALUES ARE NOT CHANGED)

POW0    SET     1
POW1    SET     2
POW2    SET     4
POW3    SET     8
POW4    SET     10H
POW5    SET     20H
POW6    SET     40H
POW7    SET     80H
POW8    SET     100H

ALSET1  SET     080H
ALSET2  SET     0C0H
ALSET3  SET     0E0H
ALSET4  SET     0F0H
ALSET5  SET     0F8H
ALSET6  SET     0FCH
ALSET7  SET     0FEH
ALSET8  SET     0FFH
ALSET9  SET     080FFH
ALSET10 SET     0C0FFH
ALSET11 SET     0E0FFH
ALSET12 SET     0F0FFH
ALSET13 SET     0F8FFH
ALSET14 SET     0FCFFH
ALSET15 SET     0FEFFH
ALSET16 SET     0FFFFH

SD      SET     0               ;SINGLE DENS (FM) CODE
DD      SET     40H             ;DOUBLE DENS (MFM) CODE
TRACKS8 SET     77
TRACKS5 SET     40


FDREL   MACRO   INCHES,DENS,SLEN,SPT,GAPLRW,GAPLFM,SKEW,SINGSIDE,DOUBSIDE
        DB      INCHES+DENS,SLEN,SPT,GAPLRW,GAPLFM,SKEW
SECTPT  SET     SPT
;; SET TRKLEN TO # OF 128-BLOCKS / TRACK
        SETPOW  LSPERPS,%SLEN   ;LOGICAL SECTORS PER PHYSICAL SECTOR
TRKLEN  SET     LSPERPS*SPT
TRKS    SET     TRACKS&INCHES   ;KK ADDED TRKS TO DOUBLE CAPACITY ON 2 SIDED
        FDREL1  INCHES,SINGSIDE
;;SECTPT         SET     SECTPT+SECTPT ;;KK DON'T DOUBLE
;;TRKLEN         SET     TRKLEN+TRKLEN ;;KK TRKLEN IS NOT CYLINDER LENGTH
TRKS    SET     TRKS+TRKS               ;;KK DOUBLE THIS INSTEAD
        FDREL1  INCHES,DOUBSIDE
        ENDM


FDREL1  MACRO INCHES,TOFFSET,BSHIFTF,DIRBKS,DIRENTS,MSDSAL,MSDRDE,MSDMBY
        SETPOW  BSHIFTM,%BSHIFTF
;;DSM    SET     (TRKLEN*(TRACKS&INCHES-TOFFSET)/BSHIFTM)-1 ;KK MODIFIED AS
DSM     SET     (TRKLEN*(TRKS-TOFFSET)/BSHIFTM)-1          ;KK ..THIS LINE
;; MAKE NUMBER OF DIRECTORY ENTRIES FROM DIRBKS OR DIRENTS
DIRE    SET     0
        IF NOT NUL DIRBKS
DIRE    SET     (DIRBKS)*4*BSHIFTM
        ENDIF
        IF NOT NUL DIRENTS
DIRE    SET     DIRENTS
        ENDIF
;; NOW REVERSE ENGINEER TO GET DIRECTORY BLOCKS
DIRB    SET     ((DIRE+3)/4+BSHIFTM-1)/BSHIFTM
        IF NOT NUL DIRBKS
        IF DIRBKS > DIRB
DIRB    SET     DIRBKS
        ENDIF
        ENDIF
        IF DSM < 256
        SETPOW  EXM,%BSHIFTF-3
```

```
        ELSE
        SETPOW  EXM,%BSHIFTF-4
        ENDIF
        SETAL01 %DIRB
        DB      TRKLEN,0,BSHIFTF,BSHIFTM-1,EXM-1
;       DW      DSM,DIRE-1,AL01,(DIRE+3)/4,TOFFSET,0    ;KK MAKES TOO BIG
                                                        ;KK ..CKS
CKSIZE  SET     (DIRE+3)/4                              ;KK
        IF INCHES EQ 5                                  ;KK
CKMAX   SET     30H                                     ;KK
        ELSE                                            ;KK
CKMAX   SET     20H                                     ;KK
        ENDIF                                           ;KK
        IF CKSIZE GT CKMAX                              ;KK
CKSIZE  SET     CKMAX                                   ;KK
        ENDIF                                           ;KK
        DW      DSM,DIRE-1,AL01,CKSIZE,TOFFSET,0        ;KK
; NOTE: DUMMY CP/M 3 PSH AND PHM FOLLOWS TRACK OFFSET
; BPB FOLLOWS...
TOTSECT SET     SECTPT*TRKS     ;TOTAL NUMBER OF SECTORS IN MEDIA
TOTALOC SET     TOTSECT/MSDSAL  ;TOTAL ALLOCATION UNITS IN MEDIA
SLEN    SET     LSPERPS*128     ;SECTOR LENGTH IN BYTES
CFS     SET     (TOTALOC+TOTALOC+TOTALOC+SLEN+SLEN-1)/(SLEN+SLEN)
        DW      SLEN            ;BYTES/SECTOR
        DB      MSDSAL,1,0,2    ;SECT/ALLOC, RESVD SECTS, NUMBER FATS
        DW      MSDRDE,TOTSECT  ;ROOT DIR ENTS AND TOTAL SECTORS IN MEDIA
        DB      MSDMBY          ;MEDIA BYTE
        DW      CFS,SECTPT      ;CALCULATED SECTORS PER FAT, SECTORS PER TRACK
        ENDM

SETPOW  MACRO   RESULT,ARG
RESULT  SET     POW&ARG
        ENDM

SETAL01 MACRO   ARG
AL01    SET     ALSET&ARG
        ENDM
;
;       RELATIONSHIP BETWEEN --
;
;       DISK TYPE               GAP LENGTH 1
;       SECTOR LENGTH           GAP LENGTH 2
;       NUMBER OF SECTORS       SKEW FACTOR
;
;       PLUS FOR CP/M ONE AND TWO SIDED DPB'S
;
;       DPB PARAMETERS ARE <ONE SIDE>,<TWO SIDES>
;       WITHIN THE POINTY BRACKETS ARE  ---
;
;   P1  TRACK OFFSET (IS THIS EVER NOT = 2?)
;   P2  CP/M ALLOCATION UNIT SIZE (3=1K, 4=2K, 5=4K, 6=8K, 7=16K)
;   P3  ALLOCATION UNITS RESERVED FOR DIRECTORY
;   P4  MAXIMUM NUMBER OF FILES ALLOWED (IF NULL THEN COMPUTE FROM P3)
;
;       AND THEN FOR MS-DOS...
;
;   P5  (PHYSICAL) SECTORS/ALLOCATION UNIT
;   P6  NUMBER OF ROOT DIRECTORY ENTRIES
;   P7  MEDIA BYTE
;       SECTORS REQUIRED TO HOLD FAT IS GENERATED IN MACRO
;
;       ALL PARAMETERS ARE REQUIRED EXCEPT ONLY ONE OF P3/P4
;       NEED BE PRESENT.  IF BOTH P3 AND P4 ARE PRESENT, P3 IS
;       ONLY USED IF IT IS LARGER THAN NEEDED.
;
;       TYPE:
;               05H  5" SINGLE DENSITY (FM)
;               08H  8" SINGLE DENSITY (FM)
;               45H  5" DOUBLE DENSITY (MFM)
;               48H  8" DOUBLE DENSITY (MFM)
;
;       SECTOR LENGTH:
;               0   128
;               1   256
;               2   512
;               3  1024
;
;       NUMBER OF SECTORS PER TRACK:
;               SECTORS PER SIDE IF DOUBLE SIDED
;
;       GAP LEN1:
;               GAP LENGTH TO BE USED WITH NEC765 (INTEL 8272) READ/WRITE CMDS
;
;       GAP LEN2:
;               GAP LENGTH TO BE USED WITH NEC765 FORMAT TRACK COMMAND
;
;       SKEW FACTOR:
;               LOGICALY CONTIGEOUS SECTORS ARE THIS FAR APART
;
;KK CP/M TOFFSET DOUBLED ON DOUBLE SIDED FLOPPIES

FDRELTAB:
STD8:
 FDREL 8,SD,0,26,007H,01BH,6,<2,3,2,,4,068,002H>,<4,4,2,,4,068,003H> ;SD 128
LENFDREL EQU    $-FDRELTAB
 FDREL 8,SD,1,15,00EH,02AH,1,<2,4,2,,2,112,004H>,<4,4,2,,2,112,005H> ;8" SD 256
 FDREL 8,SD,2,08,01BH,03AH,1,<2,4,2,,2,112,006H>,<4,4,2,,2,112,007H> ;8" SD 512
 FDREL 8,SD,3,04,047H,08AH,1,<2,4,2,,2,112,008H>,<4,4,2,,2,112,009H> ;8" SD 1024 ???

 FDREL 8,DD,1,26,00EH,036H,3,<2,4,2,,2,112,00AH>,<4,4,2,,2,112,00BH> ;8" DD 256
 FDREL 8,DD,2,15,01BH,054H,1,<2,4,2,,2,112,00CH>,<4,4,2,,2,112,00DH> ;8" DD 512
 FDREL 8,DD,3,08,035H,074H,2,<2,4,2,,1,112,00EH>,<4,4,2,,1,192,00FH> ;8" DD 1024 W/W SKEW
NFMT8   EQU     ($-STD8)/LENFDREL

STD5:
 FDREL 5,SD,0,16,010H,019H,1,<2,3,2,,1,112,018H>,<4,3,2,,2,112,019H> ;5" SD 128 16 SECT
 FDREL 5,SD,0,17,007H,009H,1,<2,3,2,,1,112,01AH>,<4,3,2,,2,112,01BH> ;5" SD 128 **NEW**
 FDREL 5,SD,0,18,007H,009H,5,<3,3,,32,1,112,01CH>,<4,3,2,,2,112,01DH> ;5" SD 128 xerox
 FDREL 5,SD,1,08,018H,030H,1,<2,3,2,,1,112,01EH>,<4,3,2,,2,112,01FH> ;5" SD 256
 FDREL 5,SD,1,10,007H,010H,2,<3,4,,64,1,112,020H>,<4,3,2,,2,112,021H> ;5" SD 256 osborne
 FDREL 5,SD,2,04,046H,087H,1,<2,3,2,,1,112,022H>,<4,3,2,,2,112,023H> ;5" SD 512

 FDREL 5,DD,1,16,020H,032H,1,<2,3,,64,1,112,024H>,<2,4,,256,2,112,025H> ;5" DD 256 16 SEC
 FDREL 5,DD,1,17,00AH,00CH,1,<3,3,,64,1,112,026H>,<6,4,,128,2,112,027H> ;5" DD 256 **NEW*
 FDREL 5,DD,1,18,00AH,00CH,1,<2,3,2,,1,112,028H>,<4,4,,64,2,112,029H> ;5" DD 256 18 SECT
 FDREL 5,DD,2,08,02AH,050H,1,<1,3,,64,1,064,0FEH>,<2,4,,64,2,112,0FFH> ;5" DD 512 PC 8 SE
 FDREL 5,DD,2,09,0030,0060,1,<2,3,,64,1,064,0FCH>,<4,4,2,,2,112,0FDH> ;5" DD 512 PC 9 SEC
 FDREL 5,DD,2,10,0017,0034,4,<2,3,,64,1,064,0FAH>,<4,4,,128,2,112,0FBH> ;5" DD 512 10 SEC
 FDREL 5,DD,3,04,080H,0F0H,1,<2,3,2,,1,112,030H>,<4,4,2,,2,112,031H> ;5" DD 1024 4 SECT
 FDREL 5,DD,3,05,0028,0050,3,<2,4,,128,1,112,032H>,<4,4,,192,2,112,033H> ;5" DD 1024 **NE

NFMT5   EQU     ($-STD5)/LENFDREL

        DB      0FFH                    ;END OF LIST


PDTFINDERS:
        DW      FD800
```

```
        DW      FD801
        DW      FD810
        DW      FD811
        DW      FD500
        DW      FD501
        DW      FD510
        DW      FD511
        DW      HD00
        DW      HD01
        DW      HD02
        DW      HD03
        DW      MEMDISK
;
        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       24.  DEBUG SUBROUTINE
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

        IF FALSE
DEBUG:
        PUSH    H
        PUSH    D
        PUSH    B
        PUSH    PSW
        LDA     DEBUGB
        ORA     A
        JZ      DEBUG2
        LXI     H,9+6+16-2
        DAD     SP
        PUSH    H               ;SO WE CAN ALSO SEE STACK DEPTH
        XCHG
        LHLD    DEBUGA
        MVI     C,10+6+16
DEBUG1  LDAX    D
        MOV     M,A
        DCX     D
        INX     H
        DCR     C
        JNZ     DEBUG1
        SHLD    DEBUGA
        POP     H               ;PULL OFF SP+WHATEVER
DEBUG2  POP     PSW
        POP     B
        POP     D
        POP     H
        RET

        ENDIF
;
        PAGE
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;
;       25.  RAM ORGANIZATION
;
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

        ORG     RAMORG
```

```
BUFFER:         DS      2000H   ;FLOPPY DISK TRACK BUFFER
WDBUFF          DS      200H    ;512-BYTE HARD DISK BUFFER
;BUFTAG MUST BE ON 100H BOUNDRY
BUFTAG:         DS      40H     ;TRACK BUFFER FLAGS (64 128-BYTE SECTORS)
READTAG         EQU     80H     ;DATA IS VALID IN BUFFER
WRITETAG        EQU     20H     ;DATA NEEDS TO BE WRITTEN BACK TO DISK
;
;       FOLLOWING AREA SETUP BY "INITALL"
;
;       ANY CHANGES MUST ALSO BE REFLECTED IN THE "PUCONST" LIST
;
PUDEF:
WRITE$DELAY     DS      2       ;# 250MS TICKS BEFORE MODIFIED BUFFERS WRITTEN
UNLOAD$DELAY    EQU     WRITE$DELAY+1   ;TICKS BEFORE FD HEADS UNLOAD
FD$STEP$RATE    DS      2+6     ;KK ALL STEP RATES (IN NEC "SPECIFY" FORMAT)
FD$LOAD$TIME    DS      2       ;8/5 HEAD LOAD TIME IN MS
FD$SETTLE       DS      2       ;8/5 HEAD SETTLE (AFTER SEEK) IN MS
SEEKRETRY1:     DS      1       ;R/W RETRY BEFORE FANCY SEEK STUFF
SEEKRETRY2:     DS      1       ;R/W RETRY WITH SEEK STUFF (TOT RETRY = R1*R2)
H1234           DS      2       ;SET=1234H AS SIGNAL THAT POWER IS UP
HDHEADS         DS      1       ;HARD DISK PARAMETER: NUMBER OF HEADS
HDPRECOMP       DS      1       ;PRE COMP CYLINDER
HDSRATE         DS      1       ;HARD DISK STEP RATE
HDCTYPE         DS      1       ;HARD DISK CONTROLLER TYPE
JMPBIO          DS      1       ;JMP INSTRUCTION***MUST BE LAST IN INIT LIST***
;
;       END OF AREA SETUP BY "INITALL"
;
;       NOTE: BOOTSUB MUST FOLLOW LAST ENTRY IN "INITALL" LIST
;
BOOTSUB:        DS      2       ;READ OR WRITE ADDRESS FOR BOOTIO SUBROUTINE
TRHLOAD         DS      1       ;TIME REMAINING FOR HEAD LOAD (TICKS AT 1 MS)
TRHSETL         DS      1       ;-DITTO- FOR HEAD SETTLE (MUST BE TRHLOAD+1)
MMDEST:         DS      3       ;MOVE MEMORY DESTINATION ADDRESS (LO, HI, EX)
MMFROM:         DS      3       ;MOVE MEMORY SOURCE ADDRESS (LO, HI, EX)
MD$ADRS         EQU     MMFROM  ;3-BYTE MEMORY DISK ADRS FROM SEEKTRK/SEEKSEC
BUFTRKS:        DS      2       ;FOR "LHLD" OF FOLLOWING
BUFTRACK EQU    BUFTRKS         ;CURRENT TRACK NUMBER IN BUFFER
BUFSIDE  EQU    BUFTRKS+1       ;CURRENT SIDE NUMBER IN BUFFER

TIME$WRITE:     DS      2        ;TIMER FOR UNWRITTEN WRITE TAGS
TIME$HLOAD      EQU TIME$WRITE+1 ;HEAD UNLOAD TIMER (TIME HEADS STAY LOADED)
NECINTADRS:     DS      2       ;NEC READ/WRITE "CONTINUE" ADDRESS
BOOTLEN:        DS      1       ;(REMAINING) # BLOCKS FOR BOOT/SYSWRT
SPSAVE:         DS      2       ;USED TO ASSURE SP RESTORED AFTER I/O DONE INT
PIOFIRSTR:      DS      1       ;FIRST RECORD WE TRIED TO READ/WRITE
PIOLASTR:       DS      1       ;LAST RECORD WE TRIED TO (OR DID) READ/WRITE
OPTYPE:         DS      1       ;R/W OP TYPE FOR GENSTAT 80H=READ, 40H=WRITE
RWTYPE:         DS      1       ;0=NORMAL, 1=DIRECTORY, 2=SEQ WRITE
RLINE5:         DS      1       ;NON-ZERO IF 5" DRIVES HAVE READY LINE

SEEKDEV:        DS      1 ; KEEP IN     LOGICAL DEVICE FROM HOST
SEEKTRK:        DS      2 ; ORDER       LOGICAL TRACK FROM HOST
SEEKHD:         DS      1 ; FOR WD      HEAD NUMBER FROM HOST (NOT FOR R/W)
SEEKSEC:        DS      1 ; TABLE       LOGICAL SECTOR FROM HOST
LWDP    SET     $-SEEKDEV
SEEKDMALO:      DS      1       ;S100 DMA ADDRESS FORM HOST, BITS 7-0
SEEKDMAHI:      DS      1       ;S100 DMA ADDRESS FROM HOST, BITS 15-8
SEEKDMAEX:      DS      1       ;S100 DMA ADDRESS FROM HOST, BITS 23-16
MODE:           DS      2       ;DMA MODE + DMA PRIORITY (ONE BYTE)
CURLCONT:       EQU     MODE+1  ;RAM COPY OF WHAT WAS LAST SENT TO LOCCONT
FWTA:           DS      2       ;ADRS OF BUFTAG FOR WRITE FAILURE (OR =0 IF OK)

WDDEV   DS      LWDP+1          ;WD COPY OF SEEK... STUFF
```

```
WDCYL   EQU     WDDEV+1
WDHEAD  EQU     WDCYL+2
WDSEC   EQU     WDHEAD+1
WDBMOD  EQU     WDSEC+1         ;WD BUFFER MODIFIED FLAG

;       READ/WRITE SECTOR IN ERROR PLUS
;       7-BYTE RESULT PHASE VECTOR

GENSTAT DS      1               ;GENERAL STATUS BYTE
PIOERSECT DS    1               ;PHYS SECTOR THAT HAD ERROR (=0 IF NO ERROR)
RES$ST0 DS      1               ; **
RES$ST1 DS      1               ; ** SEE BOOK FOR MEANING OF VARIOUS BITS
RES$ST2 DS      1               ; **
RES$C   DS      1               ;>
RES$H   DS      1               ;>>
RES$R   DS      1               ;>>>  SEE BOOK FOR MEANING OF
RES$N   DS      1               ;>>>> CHRN AT RESULT PHASE
LENESL  EQU     $-GENSTAT       ;LENGTH OF EXTENDED STATUS LIST

CPS$SECTOR: DS  1               ;PHYSICAL SECTOR FROM CPS
CPS$TRKS: DS    2               ;FOR "LHLD" OF FOLLOWING
CPS$TRACK EQU   CPS$TRKS        ;TRACK NUMBER AS COMPUTED BY CPS
CPS$SIDE EQU    CPS$TRKS+1      ;SIDE NUMBER (0 OR 1) AS COMPUTED BY CPS
CPS$BUFADR: DS  2               ;ADDRESS OF LOGICAL SECTOR WITHIN BUF
CPS$PSECT: DS   2               ;NUMBER OF BYTES IN PHYSICAL SECTOR
PFIOFLAG: DS    1               ;PHYSICAL FMT I/O FLAG

; WORKING AREA FOR R/W RETRY -- SEE ALSO SEEKRETRY1 & 2

RETRYC1: DS     3               ;TIMES LEFT TO RETRY BEFORE SEEK
RETRYC2 EQU     RETRYC1+1       ;TIMES LEFT TO RETRY USING SEEK
RETRYC3 EQU     RETRYC1+2       ;SEEK STATE: 3=NEXT SEEK UP, 2=RECAL, 1=DOWN

CURLDTADRS: DS  2               ;CURRENT LDT ADDRESS
CURLOGDEV: DS   1               ;CURRENT LOGICAL DEVICE

CURPDTADRS: DS  2
CURPHYSDEV: DS  1               ;CURRENT PHYSICAL DEVICE
CURPDT:
CURPDTLDN: DS   1
CURPDTNPD: DS   1
CURPDTFLG: DS   1
CURPDTTRK: DS   1
CURPDTPSS: DS   1
CURPDTNST: DS   1
CURPDTGPL: DS   1
CURPDTLSS: DS   1
CURPDTNBH: DS   1
CURPDTNBC: DS   1
CURPDTSKEW: DS  2       ;ADDRESS OF SKEW LIST

;*****************************************************************************
;                                                                            *
;       LOGICAL DEVICE TABLE                                                 *
;                                                                            *
;*****************************************************************************

LOGDEVTBL:
        DS      1       ;FIRST PHYSICAL DEVICE (OR 00 IF NONE)
        DS      1       ;FLAGS
        DS      1       ;LOGICAL SECTOR SIZE (0=128, 1=256,,,6=8196)
        DS      1       ;NUMBER OF LOGICAL SECTORS/TRACK
        DS      1       ;NUMBER OF LOGICAL HEADS
        DS      2       ;NUMBER OF LOGICAL CYLINDERS
        DS      7*15    ;SPACE FOR LOGICAL DEVICES 01-0F  (CP/M B: TO P:)


;*****************************************************************************
;                                                                            *
;       PHYSICAL DEVICE TABLE                                                *
;                                                                            *
;*****************************************************************************

REORG   SET     $
        ORG     0
PDTLDN  DS      1       ;LOGICAL DEVICE NUMBER
PDTNPD  DS      1       ;NEXT PHYSICAL DEV IN SAME LOGICAL DEV (OR 00 IF LAST)
PDTFLG  DS      1       ;FLAGS
                        ;80 RECAL REQUIRED BEFORE FIRST SEEK
                        ;40 MFM, 00 FM
                        ;20 WRITE PROTECT (PHYSICAL)
                        ;10
                        ;08
                        ;04 DOUBLE SIDED
                        ;02 ** BOTH OF THESE BITS ARE USED AS MEDIA CHANGE
                        ;01 ** STATE TABLE VALUES -- SEE TABLE BELOW
;
;               ----------MEDIA CHANGE STATUS STATE TABLES------------
;                                                                 (MEDIA
; CODE & NAME   COME    DROP    GOOD    FAILED  GET M CH 8<==>5  CHANGE
; ------------  READY   READY   GETDPB  GETDPB  STAT     SWAP    STATUS)
; 00 IPL STATE  00      00      10      00      00       01      -1
; 01 DONT KNOW  01      01      10      00      01       01       0
; 10 LIVE UNIT  10      01      10      00      10       01      +1
;
;       IE DROP READY DOES "ANI NOT 02"
;          COME READY AND GET MEDIA CHANGE STATUS DO NOT CHANGE FLAGS
;          GOOD GETDPB DOES "ANI NOT 03, ORI 02"
;          FAILED GETDPB DOES "ANI NOT 03"
;          8<==>5 SWAP DOES "ANI NOT 03, ORI 01"
;
;       THE ABOVE IS FOR FLOPPY DISK, REMOVEABLE MEDIA.
;
;       FOR HARD DISK AND MEMORY DISK, FIXED MEDIA, THE
;       FOLLOWING APPLYS:
;               DEVICE TABLE FLAG BITS ARE NOT USED
;               MEDIA CHANGE STATUS = +1 ALLWAYS AND FORVER
;               (I SURE HOPE THAT THIS WORKS)
;
;       SPECIAL NOTE FOR 5" FLOPPY DRIVES:
;               MANY 5" DRIVES DO NOT HAVE A "READY" LINE.
;               FOR THESE DRIVES, A FAKED "READY" APPEARS ON ALL 4 5" UNITS.
;               FOR THSE DRIVES, HEAD UNLOAD SETS FLAGS SAME AS DROP READY.
;
;               SOME NEWER 5" DRIVES HAVE A "READY" LINE.
;               FOR THESE DRIVES, "READY" COMES FROM THE DRIVE ITSELF.
;
;               FOR MIXED 5" TYPES THE BOARD MUST BE WIRED TO FAKE "READY".
;
;
PDTTRK  DS      1       ;"C" CURRENT TRACK (IF SEEKING = DESTINATION TRACK)
                        ;"H" & "R" ARE NOT IN THIS TABLE
PDTPSS  DS      1       ;"N" PHYSICAL SECTOR SIZE (0=128, 1=256,,,6=8196)
PDTNST  DS      1       ;"EOT" SECTORS/TRACK
PDTGPL  DS      1       ;"GPL" GAP LENGTH FOR READ/WRITE
PDTLSS  DS      1       ;LOGICAL SECTORS/SIDE (128-BYTE SIZE)
PDTNBH  DS      1       ;PHYSICAL NUMBER OF HEADS (1 OR 2 FOR FLOPPY)
PDTNBC  DS      1       ;PHYSICAL NUMBER OF CYL
```

```
        DS      0       ;SKEW LIST BEGINS HERE
LENPDTH EQU     $       ;LENGTH OF PDT HEADER (FOR FLOPPYS)
;       SKEW FACTOR LIST FOLLOWS
;       FOLLOWED BY A SKEW LIST

        ORG     REORG

LST8    EQU     52      ;LENGTH OF SKEW TABLE FOR 8" DRIVES
LST5    EQU     18      ;LENGTH OF SKEW TABLE FOR 5" DRIVES

PHYSDEVTBL:
FD800   DS      LENPDTH+NFMT8+LST8      ;8" 00
FD801   DS      LENPDTH+NFMT8+LST8      ;8" 01
FD810   DS      LENPDTH+NFMT8+LST8      ;8" 10
FD811   DS      LENPDTH+NFMT8+LST8      ;8" 11
FD500   DS      LENPDTH+NFMT5+LST5      ;5" 00
FD501   DS      LENPDTH+NFMT5+LST5      ;5" 01
FD510   DS      LENPDTH+NFMT5+LST5      ;5" 10
FD511   DS      LENPDTH+NFMT5+LST5      ;5" 11

REORG   SET     $
        ORG     0               ;HARD DISK PHYSICAL DEVICE TABLE STRUCTURE
        DS      1               ;LOGICAL DEV #
        DS      1               ;NEXT DEVICE IN SET (NO REAL SUPPORT FOR THIS)
        DS      1               ;FLAGS (NO USE ...YET)
PDTHHEADS DS    1               ;NUMBER OF HEADS ON THIS DEVICE
PDTHPCMP DS     1               ;PRECOMP CYLINDER FOR THIS DEVICE
PDTHSRATE DS    1               ;STEP RATE FOR THIS DEVICE
PDTHDEV DS      1               ;DEVICE SELECT CODE
PDTHOFFSET DS   2               ;"HARDWARE" OFFSET
PDTHOSLEN DS    1               ;LENGTH OF FOLLOWING OS TABLE
        DS      25              ;RESERVE SPACE FOR OS TABLE
LENHDDT EQU     $               ;LENGTH OF HARD DISK DEVICE TABLE
        ORG     REORG

HD00    DS      LENHDDT         ;HARD DISK 0 NOTE -- FORMAT NOT THE SAME AS FD
HD01    DS      LENHDDT         ;HARD DISK 1
HD02    DS      LENHDDT         ;HARD DISK 2
HD03    DS      LENHDDT         ;HARD DISK 3


MEMDISK:
        DS      1               ;LOGICAL DEVICE NUMBER
        DS      1               ;NEXT PHYSICAL DEVICE IN SET (OR 00 IF LAST)
        DS      1               ;FLAGS
MD$INT$FLG DS   1               ;INIT FLAG FOR MEMORY DISK
MD$LEP  DS      8               ;(4 SETS OF) LENGTH/(END ADDRESS + 1) PARAMETER
MD$LOSB DS      1               ;LENGTH OF OS BLOCK FOR MEMORY DISK
MD$OSB  DS      25              ;OS BLOCK (DPB OR EQUIV) AS SENT FROM HOST

;
; FPYTAB DEFINED
;
;FLAGS: DS      1               ;XXXX XXXX
;                               ;^     ^
;                               ;|     \--- DOUBLE SIDED
;                               ;\--------- ACTIVE
;FDRELPOS: DS   1               ;RELITIVE POSITION IN FDRELPOS
;SECTOFF:  DS   2               ;SECTOR OFFSET FOR SIDE 0, SIDE 1
;SIDEOFF:  DS   2               ;SIDE I.D. OFFSET FOR SIDE 0, SIDE 1
;SKEW:     DS   1               ;SKEW FACTOR, 0 = NO CHANGE
;LENDPB:   DS   1               ;LENGTH OF DPB TO FOLLOW
;DPB:      DS   20              ;ROOM FOR MAX SIZE DPB AND THEN SOME
FPYTABSIZ: EQU  27
;
FPYTAB: DS      FPYTABSIZ*8     ;8 FLOPPY UNITS
CURFPYTAB: DS   2               ;CURRENT FPYTAB ADDRESS
CURRAMDPB: DS   2               ;POINTER TO CURRENT RAM DPB, 0= NONE
RAMDPBSKEW: DS  1               ;SKEW FACTOR IF RAMDPB VALID

RID$SIDE:   DS  1               ;KK USED BY RID
RTRY$TRACK: DS  2               ;KK CALCULATED CYLINDER FOR RWRETRY, FROM SMS
VFYSEC:     DS  1               ;KK USED BY VERIFY FORMAT AS OPTION (VRS 1.1)
BOOTTRK:    DS  2               ;KK TRACK COUNTER FOR BOOT

DEBUGA: DS      2
DEBUGB: DS      1
DEBUGC  EQU     ($+15) AND 0FFF0H

STACK   EQU     RAMEND-80H      ;STACK BASE
MISCBUF EQU     RAMEND-80H      ;BUFFER USED FOR MEMDISK AND DMA MOVE

        ORG     MISCBUF
; AND 0FFF0H

STACK   EQU     RAMEND-80H      ;STACK BASE
MISCBUF EQU     RAMEND-80H      ;BUFFER USED FOR MEMDISK AND DMA MOVE

        ORG     MISCBUF
;
;       ORG FOR VARIOUS SCRAP AREAS THAT ARE NOT NEEDED OUTSIDE THE
;       ROUTINE THAT SETS THEM.  BE CAREFUL AS THIS IS ALSO THE
;       DMA SCRAP BUFFER.
;

PRWA    DS      2               ;TEMP STORAGE FOR PRW

FMT$NUM DS      1               ;FORMAT NUMBER
FMT$CNTL DS     1               ;FORMAT CONTROL
FMT$RELADRS DS  2               ;TEMP AREA FOR FMTTRK

SSPU:   DS      1               ;1 SET SKEW (LOGICAL OR) PHYSICAL UNIT
SSDL:   DS      1               ; 2 SET SKEW DENSITY AND SECTOR LEN
SSNS:   DS      1               ;  3 SET SKEW NUMBER OF SECTORS (0=DONT CARE)
SSFACT: DS      1               ;   4 SET SKEW FACTOR (0 ==> DEFAULT)

        END

A>
```