;

; 17 JUL 80

;

; MODIFIED FOR HEAD LOAD DELAY 16 DEC 80

; MODIFIED FOR PROPER UNIT/STPRAT CORRELATION 9 DEC 80

; MODIFIED FOR SEEK ERROR TRAPPING 10 DEC 80

; MODIFIED FOR NOT READY TRAPPING 11 DEC 80

; MODIFIED FOR MINI/MAXI CONDITIONALS 11 DEC 80

; MODIFIED FOR UNALLOCATED WRITE CORRECTIONS 8 SEP 80

; MODIFIED FOR AUTO-BOOT OPERATION 7 SEP 80

; MODIFIED FOR COMPUTED SKEW FACTORS 7 SEP 80

; MODIFIED FOR WARM BOOTS FROM MINIS 14 AUG 80

; MODIFIED FOR PROPER DEFAULT DISK RESTORAL ON WARM BOOT 12 AUG 80

; MODIFIED FOR 2 MHZ 8080 OPERATION 12 AUG 80

;

TITLE 'CCS 2422 DEBLOCKED BIOS FOR CP/M 2.2'

PAGE 56

;

;

; THIS BIOS IS SET UP FOR AUTO SELECT OF DISK CHAR

;

;"BIAS" IS ADDRESS OFFSET FROM 2C00H FOR MEMORY SYSTEMS

;THAN 20K (REFERRED TO AS "B" THROUGHOUT THE TEXT).

;

VERS: EQU 22 ;CP/M VERSION NUMBER

MSIZE: EQU 20 ;CP/M VERSION MEMORY SIZE IN KILOBYTES

BIAS: EQU (MSIZE-20)\*1024

CCP: EQU 2C00H+BIAS ;BASE OF CCP

BDOS: EQU CCP+806H ;BASE OF BDOS

BIOS: EQU CCP+1600H ;BASE OF BIOS

WBOOTV: EQU 0

IOBYTE: EQU 3 ;INTEL IOBYTE LOCATION

CDISK: EQU 4

BDOSV: EQU 5

;

TRUE: EQU 0FFFFH

FALSE: EQU NOT TRUE

;

MINI: EQU TRUE

MAXI: EQU TRUE ;CONDITIONAL FLAG FOR 8" DRIVE SUPPORT

BOTH: EQU MINI AND MAXI

;

SDATA: EQU 20H ;SERIAL DATA PORT

SINTEN: EQU SDATA+1 ;SERIAL INTERRUPT ENABLE PORT

SIDENT: EQU SDATA+2 ;SERIAL INTERRUPT IDENTIFICATION PORT

SLCTRL: EQU SDATA+3 ;SERIAL LINE CONTROL PORT

SMDMCT: EQU SDATA+4 ;SERIAL MODEM CONTROL PORT

SLSTAT: EQU SDATA+5 ;SERIAL LINE STATUS PORT

SMDMST: EQU SDATA+6 ;SERIAL MODEM STATUS PORT

;

RXRDY: EQU 00000001B ;RECEIVE DATA AVAILABLE BIT

TXMTY: EQU 00100000B ;TRANSMIT BUFFER EMPTY BIT

;

; WHEN THE AUTO-BOOT JUMPER IS ENABLED, THE 2810

; SERIAL PORT WILL BE INITIALIZED TO 9600 BAUD.

; TO SELECT A DIFFERENT BAUD RATE, CHANGE SBAUD

; TO ONE OF THE FOLLOWING VALUES:

;

; BAUD RATE SBAUD

; 50 2304

; 75 1536

; 110 1047

; 134.5 857

; 150 768

; 300 384

; 600 192

; 1200 96

; 1800 64

; 2000 58

; 2400 48

; 3600 32

; 4800 24

; 7200 16

; 9600 12

; 19200 6

; 38400 3

; 56000 2

;

SBAUD: EQU 12 ;9600 BAUD DIVISOR FOR 2810

;

DSTAT EQU 30H ;DISK STATUS PORT

DCMMD EQU DSTAT ;DISK COMMAND PORT

DTRCK EQU DSTAT+1 ;DISK TRACK PORT

DSCTR EQU DSTAT+2 ;DISK SECTOR PORT

DDATA EQU DSTAT+3 ;DISK DATA PORT

DFLAG EQU DSTAT+4 ;DISK FLAF PORT

DCTRL EQU DSTAT+4 ;DISK CONTROL PORT

BCTRL: EQU 4 ;DISK STATUS 2 PORT

;

RSTR: EQU 8 ;BASIS OF RESTORE COMMAND

SEEKV: EQU 1CH ;BASIS OF SEEK COMMAND

STEPI: EQU 58H ;BASIS OF STEP IN COMMAND

RDSEC: EQU 88H ;BASIS OF READ SECTOR COMMAND

RDADD: EQU 0C4H ;READ ADDRESS COMMAND

;

; STEP RATES MAY BE TAILORED TO MEET INDIVIDUAL DRIVE REQUIREMENTS.

; TO DO SO, DETERMINE THE PROPER TRACK-TO-TRACK STEP RATES FROM

; YOUR DISK DRIVE'S TECHNICAL MANUAL, AND SET THE STEP5 (FOR MINI

; DRIVES) AND/OR STEP8 (FOR 8" DRIVES) TO THE VALUE SHOWN IN

; THE FOLLOWING TABLE:

;

; VALUE MINI DISKS 8" DISKS

; 0 6 MS 3 MS

; 1 12 MS 6 MS

; 2 20 MS 10 MS

; 3 30 MS 15 MS

;

STEP5: EQU 3 ;MINI DISK STEP RATE

STEP8: EQU 1 ;8" DISK STEP RATE

;

IF MINI

HLWAIT: EQU 3000H ;HEAD LOAD WAIT FOR MINI DRIVES

ENDIF

IF NOT MINI

HLWAIT: EQU 1600H ;HEAD LOAD WAIT FOR 8" DRIVES

ENDIF

;

TRIES: EQU 10 ;NUMBER OF ATTEMPTS

;

;DEBLOCK PARAMETERS

WRALL: EQU 0

WRDIR: EQU 1

WRUAL: EQU 2

;

;

CTRLC: EQU 3 ;ASCII ETX

BELL: EQU 7 ;ASCII BELL CHARACTER

CR: EQU 0DH ;ASCII CARRIAGE RETURN

LF: EQU 0AH ;ASCII LINE FEED

;

DISKNO: EQU 40H ;ACTIVE DISK NUMBER

TRACK: EQU DISKNO+1

SECTOR: EQU TRACK+1

SIDE: EQU SECTOR+1 ;SIDE SELECT HOLD AREA

SPT: EQU SIDE+1 ;SECTORS PER TRACK HOLD

TWOSID: EQU SPT+1 ;SINGLE/DOUBLE SIDED SWITCH HOLD

STPRAT: EQU 46H ;STEP RATE SAVE AREA

STATUS: EQU 47H

CMND: EQU STATUS+1

LUNIT: EQU 49H ;LAST USED DRIVE

CUNIT: EQU LUNIT+1 ;CURRENT DRIVE

RWFLG: EQU 4BH

HSTBUF: EQU 4CH ;HOST BUFFER ADDRESS

IDSV: EQU 4EH ;SECTOR ID SAVE AREA

TBUF: EQU 80H

TPA: EQU 100H

;

ORG BIOS ;ORIGIN OF THIS PROGRAM

;

;JUMP VECTOR FOR INDIVIDUAL SUBROUTINES

JMP BOOT ;COLD START

WBOOTE: JMP WBOOT ;WARM START

CSTAT: JMP CONST ;CONSOLE STATUS

CONIN: JMP CONI ;CONSOLE CHARACTER IN

CONO: JMP CONOUT ;CONSOLE CHARACTER OUT

JMP LIST ;LIST CHARACTER OUT

JMP PUNCH ;PUNCH CHARACTER OUT

JMP READER ;READER CHARACTER OUT

JMP HOME ;MOVE HEAD TO HOME POSITION

JMP SELDSK ;SELECT DISK

JMP SETTRK ;SET TRACK NUMBER

JMP SETSEC ;SET SECTOR NUMBER

JMP SETDMA ;SET DMA ADDRESS

JMP READ ;READ DISK

JMP WRITE ;WRITE DISK

JMP LISTST ;RETURN LIST STATUS

JMP SECTRAN ;SECTOR TRANSLATE

;

;INDIVIDUAL SUBROUTINES TO PERFORM EACH FUNCTION

;

;

RETRY: DCR C

JNZ LOAD1

JRNZ LOAD1

LXI H,BOTMSG

CALL PRTRD

CALL PRTWA

WBOOT: LXI SP,TBUF

MVI C,TRIES ;# OF RETRIES

LOAD1: XRA A

MOV H,A ;SET THE UNIT AND TRACK

MOV L,A

SHLD DISKNO

LXI H,0D002H ;SET THE SIDE AND SECTOR

SHLD SECTOR

MVI B,12 ;ZERO OUT PRMTBL

LXI H,PRMTBL

LOAD1A: MOV M,A

INX H

DCR B

JNZ LOAD1A

DJNZ LOAD1A

PUSH B

MOV C,A

MOV E,A ;SET NEW UNIT INDICATOR

CALL SELDSKA

POP B

LDA PRMTBL+1

STA CUNIT

LXI H,CCP

SHLD HSTBUF

IF MAXI

MVI B,26 ;HOLD THE SPT IN (B)

ENDIF

IF BOTH

IN BCTRL ;SET # OF SECTORS FOR TRK00

ANI 2 ;ISOLATE MINI/MAXI BIT

JZ LOAD2 ;JUMP IF 8"

JRZ LOAD2

ENDIF

IF MINI

MVI B,18 ;SET MINI SPT

ENDIF

LOAD2: PUSH B

CALL DREAD

POP B

ORA A

JNZ RETRY

JRNZ RETEY

SHLD HSTBUF

MOV D,H ;SAVE THE PAGE ADDRESS

LXI H,SECTOR ;POINT TO THE SECTOR HOLD

MOV A,M ;SEE IF READY FOR NEXT TRACK

SUB B

JC LOAD3

JRC LOAD3

MOV M,A ;RESET THE SECTOR COUNT

DCX H ;POINT TO TRACK

INR M ;ADVANCE IT

INX H ;POINT BACK TO SECTOR

IF BOTH

IN BCTRL ;SET THE NEW SPT VALUE

ANI 2 ;SEE IF MINI OR MAXI

JZ LOAD3 ;CURRENT VALUE GOOD FOR MAXI, JUMP

JRZ LOAD3

ENDIF

IF MINI

LDA CUNIT ;SEE IF DOUBLE DENSITY

ANI 40H

JNZ LOAD3 ;CURRENT VALUE OK IF DDEN SET

JRNZ LOAD3

LDA IDSV+3 ;GET SECTOR SIZE INDICATOR

DCR A ;SEE WHICH SIZE

JM LOAD3 ;JUMP IF 128 BYTE SECTORS

CANNOT USE JNC!

MVI B,10 ;SPT FOR 256 BYTE SECTORS

JZ LOAD3

JRZ LOAD3

MVI B,5 ;SPT FOR 512 BYTE SECTORS

ENDIF

LOAD3: INR M

LDA IDSV+3 ;SEE IF ENOUGH LOADED IN

ADD D

SUI BIOS/256

JC LOAD2 ;JUMP IF MORE NEEDED

JRC LOAD2

;

;END OF LOAD OPERATION, SET PARAMETERS AND GO TO CP/M

BOOT0: MVI A,JMP ;GET A JUMP OP CODE

STA WBOOTV ;RESET THE JUMP VECTORS

STA BDOSV

LXI H,WBOOTE

SHLD WBOOTV+1

LXI H,BDOS

SHLD BDOSV+1

LXI H,DBUF ;SET UP BUFFER ADDRESS

SHLD HSTBUF

LXI H,TBUF ;DEFAULT DMA ADDRESS IS 80H

SHLD DMAAD

SHLD HSTACT ;HOST NOT ACTIVE

LDA CDISK

MOV C,A ;SEND TO THE CCP

JMP CCP+3 ;GO TO CPM

;

;

;SELECT DISK GIVEN BY REGISTER C

SELDSK: LXI H,DISKNO ;CHECK TO SEE IF DRIVE CHANGED

XRA A ;E=0 WILL MEAN CHANGE

MOV E,A

MOV A,C

CMP M ;NEW DISK?

JRNZ ITSNEW

INR E ;E=1 MEANS NO CHANGE

ITSNEW:

STA SEKDSK

SELDSKA: LXI H,0 ;ERROR RETURN CODE

CPI 4 ;MUST BE BETWEEN 0 AND 3

RNC ;NO CARRY IF 4,5,...

;DISK NUMBER IS IN THE PROPER RANGE

;COMPUTE PROPER DISK PARAMETER HEADER ADDRESS

SELDSK1: MOV L,A ;L=DISK NUMBER 0,1,2,3

DAD H ;\*2

DAD H ;\*4

DAD H ;\*8

DAD H ;\*16 (SIZE OF EACH HEADER)

MOV A,E ;GET THE NEW UNIT INDICATOR BIT

THIS SURE LOOKS LIKE A BUG. Where does E get set?

LXI D,DPBASE

DAD D ;HL=.DPBASE(DISKNO\*16)

RAR ;TEST THE NEW UNIT BIT

PUSH PSW

PUSH H

MOV A,C

CALL FDSB0

STA SEKSEL

POP H

POP PSW

MOV A,C ;RETURN THE DISKNO

RC ;RETURN IF NOT NEW UNIT

;

CKSET: PUSH H ;SAVE (H,L)

CKSET0: LHLD LUNIT ;FIRST, SAVE CURRENT DISK ASSIGNMENTS

PUSH H

STA CUNIT ;FORCE THE READ ADDRESS

LHLD DISKNO

LDA SIDE

MOV H,A

PUSH H

MOV A,C ;REGET DESIRED UNIT

STA DISKNO ;SET NEW DISK NUMBER

MVI A,0D0H ;SELECT SIDE 0 OF NEW DISK

STA SIDE

PUSH B

CALL IDRD1 ;FIND OUT WHAT IS OUT THERE

JNZ SELERR

JRNZ SELERR

MOV C,A ;GET THE SECTOR SIZE

DCX H ;POINT TO THE SELBITS

MOV B,M ;GET THEM

LXI D,4 ;ADDRESS TABLE ENTRY OFFSET

IF MINI

LXI H,MSELTBL-4 ;MINI TABLE ADDRESS

ENDIF

IF BOTH

MOV A,B ;REGET THE SELBITS

ANI 10H ;ISOLATE THE MINI DRIVE BIT

JZ SETUP1 ;JUMP IF MINI

JRZ SETUP1

ENDIF

IF MAXI

LXI H,SELTBL-4 ; ELSE, SET THE 8" TABLE ADDRESS

IN BCTRL ;CHECK FOR DOUBLE SIDED DISK

ANI 40H ;ISOLATE TWO-SIDED BIT

JNZ SETUP1 ;JUMP IF SINGLE-SIDED

JRNZ SETUP1

LXI H,SELTBLA-4

ENDIF

SETUP1: MOV A,B ;CHECK FOR DOUBLE DENSITY

STA SEKSEL

ANI 40H ;ISOLATE THE BIT

JZ SETUP3 ;JUMP IF SINGLE-DENSITY

JRZ SETUP3

DAD D ;OFFSET TO DOUBLE DENSITY ENTRIES

DAD D

DAD D

DAD D

SETUP3: DAD D ;OFFSET TABLE ADDRESS

DCR C

JP SETUP3

CANNOT USE JNC!

SET3: XCHG ;SAVE THE POINTER

CMP A ;ZERO OUT FLAGS

SELERR: POP B ;RESTORE REGISTERS

POP H ;RESTORE THE CURRENT DRIVE

MOV A,H ;RESTORE THE SIDE

STA SIDE

MOV A,L ; AND THE DISKNO

STA DISKNO

POP H

SHLD LUNIT

POP H

JNZ SELERRA ;JUMP IF A SELECT ERROR

JRNZ SELERRA

PUSH H ;GET, RESAVE DP BLOCK POINTER

LDAX D ;MOVE THE TABLE ENTRIES

MOV M,A

INX D

INX H

LDAX D

MOV M,A

INX D

PUSH D ;SAVE IT FOR A MOMENT

LXI D,9

DAD D ;OFFSET THE POINTER

POP D ;REGET TABLE ADDRESS

LDAX D ;MOVE THE TABLE ENTRIES

MOV M,A

INX D

INX H

LDAX D

MOV M,A

SET4: POP H

MOV A,C

RET

SELERRA: LXI H,0

MOV A,C

RET

;

;SET TRACK GIVEN BY REGISTER C

SETTRK: MOV A,C

STA SEKTRK

RET

;

;SET SECTOR GIVEN BY REGISTER C

SETSEC: MOV A,C

ANI 7FH ;STRIP OFF SIDE INDICATOR

DCR A

STA SEKSEC

MOV A,C ;REGET SECTOR NUMBER

RAL ;ISOLATE SIDE BIT

MVI A,0D0H ;SET UP SIDE SELECT BITS

JNC SETSEC1

JRNC SETSEC1

MVI A,90H ;SELECT SIDE 1

SETSEC1: STA SEKSID ;SET THE SIDE SELECT BITS

RET

;

;TRANSLATE THE SECTOR GIVEN BY BC USING THE

;TRANSLATE TABLE GIVEN BY DE

SECTRAN: PUSH D ;SAVE THE TABLE ADDRESS

LDA SEKDSK

CALL FDSB0 ;GET THE PARAMETER TABLE ADDRESS

INX H ;POINT TO THE SECTOR SIZE ENTRY

MOV D,M ;SECTOR SIZE NOW IN (D)

PUSH D ;SAVE FOR LATER USE

MOV A,C ;GET THE DESIRED SECTOR

STA CPMSEC

RAL

SECT0: ORA A ;CONVERT TO PHYSICAL SECTOR NUMBER

RAR

DCR D

JP SECT0

CANNOT USE JNC!

POP D

POP H ;REGET TABLE ADDRESS

CMP M ;SEE IF SIDE 1

MVI B,0 ;SET FOR SIDE 0

JC SECT1 ;JUMP IF SIDE 0

JRC SECT1

MVI B,80H ;FLAG BIT FOR SIDE 1

SUB M

SECT1: PUSH B

MOV B,D ;SAVE SECTOR SIZE IN (B)

MOV E,A ;SET UP TO BUILD SKEW

MOV D,M ;GET THE SKEW FACTOR

INX H

MVI C,0FFH ;GET A -1

SECT2: INR C ;BUILD SECTOR OFFSET IN (C)

SUB M

JNC SECT2 ;LOOP TIL OFFSET IS BUILT

JRNC SECT2

INX H ;POINT TO SKEW FACTOR

XRA A ; AND GET A ZERO

SECT3: ADD M ;BUILD THE SKEWED SECTOR NUMBER

DCR E

JP SECT3

CANNOT USE JNC!

SUB M

ADD C ;ADD ON THE OFFSET

SECT4: SUB D ;INSURE NUMBER IS IN RANGE

JNC SECT4

JRNC SECT4

ADD D

SECT5: MOV L,A ;MOVE SKEWED NUMBER OVER TO (L)

INR A ;SET PYHS SECTOR FOR ANTIC. LOGIC

STA NXTSCT

XRA A ;CONVERT PHYSICAL TO LOGICAL SECTOR

MOV H,A

SECT6: DCR B

JM SECT7 ;JUMP IF DONE

CANNOT USE JNC!

STC

ADC A ;BUILD THE MASK

DAD H ;OFFSET THE SECTOR NUMBER

JMP SECT6

JR SECT6

SECT7: POP B

ANA C ;STRIP OUT THE SUB-SECTOR

ORA L ;ADD IT TO THE SKEWED SECTOR

INR A

ORA B ;ADD ON THE SIDE SELECT BIT

MOV L,A

RET

;

;

;SET DMA ADDRESS GIVEN BY REGISTERS B AND C

SETDMA: MOV L,C ;LOW ORDER ADDRESS

MOV H,B ;HIGH ORDER ADDRESS

SHLD DMAAD ;SAVE THE ADDRESS

RET

;

;MOVE TO THE TRACK 00 POSITION OF CURRENT DRIVE

HOME: SUB A

STA SEKTRK

RET

;

;DEBLOCK ROUTINES

;

READ: MVI A,WRUAL ;TREAT READ AS UNALLOCATED

STA WRTYPE

STA READOP ;SET A READ OPERATION

JMP ALLOC ;GO DO THE READ

JR ALLOC

;

WRITE: XRA A ;FLAG AS A WRITE OPERATION

STA READOP

MOV A,C ;WRITE TYPE FROM CP/M

STA WRTYPE

CPI WRUAL ;SEE IF UNALLOCATED

JNZ CHKUNA ;JUMP IF NOT

JRNZ CHKUNA

;

; UNALLOCATED WRITE, SET PARAMETERS

CALL PHYSEC ;CONVERT TO PHYSICAL SECTOR

LHLD SEKHST ;SET PHYSICAL SECTOR AND SIDE

SHLD UNASEC

LHLD SEKDSK ;GET THE DISK NUMBER

SHLD UNADSK ;SAVE IT FOR UNALLOCATED WRITE

MOV A,L

CALL DPFND ;GET DP TABLE ADDRESS

INX D ;OFFSET TO THE BLOCK MASK

INX D

INX D

LDAX D ;GET THE BLOCK MASK

INR A ;(A) = LOGICAL SECTORS PER BLOCK

STA UNACNT ;SAVE THE SECTOR COUNT

;

CHKUNA: LXI H,UNACNT ;SEE IF ANY UNALLOCATED SPACE AVAILABLE

XRA A

CMP M

JZ ALLOC1 ;JUMP IF NOT

JRZ ALLOC1

DCR M ;ELSE, USE SOME OF IT

CALL PHYSEC ;SET THE PHYSICAL SECTOR NUMBER

LXI D,UNADSK

CALL COMP ;COMPARE THE UNITS

JZ NXTSEC ;GO DO THE WRITE

TOO FAR?

;

ALLOC: XRA A ;ALLOCATED WRITE REQUIRES PREREAD

STA UNACNT

ALLOC1: INR A

ALLOC2: STA RSFLAG

;

RWOPER: XRA A ;GET A ZERO

STA ERFLAG ;RESET THE ERROR FLAG

;

LXI H,HSTACT ;SEE IF HOST ACTIVE

ORA M

MVI M,1 ;MARK IT ACTIVE FOR NEXT TIME

JZ FILHST ;FILL THE HOST BUFFER IF EMPTY

JRZ FILHST

;

CALL PHYSEC ;SET THE PHYSICAL SECTOR NUMBER

LXI D,DISKNO

CALL COMP ;COMPARE THE UNITS

JZ MATCH

JRZ MATCH

;

NOMATCH: LDA HSTWRT ;SEE IF HOST WRITTEN

ORA A

CNZ DWRITA ;PURGE THE BUFFER IF NEED BE

FILHST: CALL PHYSEC

MOV A,D

STA IDSV+3

LHLD SEKDSK ;SET UP TO FILL THE BUFFER

SHLD DISKNO

LHLD SEKHST ;GET THE SECTOR, SIDE SELECT

SHLD SECTOR ;SET THEM

LDA SEKSEL

STA CUNIT

LDA RSFLAG

ORA A

CNZ DREAD ;FILL IT IF NEED BE

LXI H,ERFLAG

ORA M

MOV M,A

XRA A ;RESET PENDING WRITE FLAG

STA HSTWRT

;

MATCH: LHLD IDSV+2 ;GET SECTOR SIZE

XRA A ;GET A ZERO

MATCH1: ADC A ;BUILD SECTOR MASK

DCR H

STC

JP MATCH1 ;LOOP TIL MASK IS BUILT

CANNOT USE JNC!

LHLD SEKSEC

ANA L ;FIND THE RELATIVE SECTOR

LXI H,DBUF-80H ;BUILD ADDRESS FOR CPM SECTOR

LXI B,80H ;# BYTES IN LOGICAL SECTOR

MATCH2: DAD B

DCR A

JP MATCH2

CANNOT USE JNC!

XCHG ;BUFFER ADDRESS TO (D,E)

LHLD DMAAD ;GET DMA ADDRESS

LDA READOP ;SEE IF READ OR WRITE

ORA A

JNZ RWMOVE ;POINTERS OK IF READ

JRNZ RWMOVE

;

XCHG ;ELSE, SWAP THEM

INR A ; AND MARK WRITE OPERATION

STA HSTWRT

;

RWMOVE: LDAX D ;GET A BYTE

MOV M,A ;PUT IT

INX D

INX H

DCR C ;LOOP CONTROL

JNZ RWMOVE

JRNZ RWMOVE

;

LDA WRTYPE ;GET WRITE TYPE

DCR A ;SEE IF DIRECTORY ENTRY

LDA ERFLAG ;GET THE ERROR FLAG

RNZ ;DONE IF NOT DIRECTORY ENTRY

;

ORA A ;SEE IF ANY ERRORS

RNZ ;RETURN IF SO

STA HSTWRT ;RESET HOST WRITTEN

DWRITA: CALL DWRITE ;UPDATE THE DIRECTORY

LXI H,ERFLAG

ORA M ;GET THE ERROR INDICATIONS

MOV M,A ;STORE THE ERROR FLAGS

RET

;

; UNALLOCATED WRITE NEXT SECTOR ANTICIPATION LOGIC

;

NXTSEC: LDA SEKDSK ;FIND ADDRESS OF DP TABLE

CALL DPFND

LDA CPMSEC ;GET LAST LOGICAL SECTOR NUMBER

INR A

XCHG ;DP TABLE ADDRESS TO (H,L)

CMP M ;SEE IF OVERFLOW

JNC NXTSC2 ;JUMP IF SO

JRNC NXTSC2

LXI H,-11 ;SAME TRACK, NOW SEE WHICH SIDE/SECTOR

DAD D

MOV E,M ;GET ADDRESS OF SKEW TABLE

INX H

MOV D,M

MOV C,A ;SET UP TO CALL SECTRAN

MVI B,0

CALL SECTRAN ;TRANSLATE THE SECTOR

MVI H,0D0H ;SIDE 0 SELECT

JP NXTSC1 ;JUMP IF SIDE 0

WHAT IS THIS TESTING?

MVI H,090H ;ELSE, SET SIDE 1

NXTSC1: LDA NXTSCT ;GET THE NEXT PHYSICAL SECTOR

MOV L,A ;SET THE SECTOR

JMP NXTSC3

JR NXTSC3

;

NXTSC2: LXI H,UNATRK ;SET FOR NEXT TRACK

INR M

LXI H,0D001H ;SIDE 0 SECTOR 1

NXTSC3: SHLD UNASEC ;SET THE NEXT SIDE, SECTOR

XRA A ;GET A ZERO

JMP ALLOC2 ;GO BACK TO MAINSTREAM

TOO FAR?

;

; FIND ADDRESS OF DISK PARAMETER TABLE

;

DPFND: LXI H,DPBASE-6 ;DEVELOP ADDRESS OF DPTABLE

LXI D,16

DPFND1: DAD D

DCR A

JP DPFND1

CANNOT USE JNC!

MOV E,M ;PULL UP THE ADDRESS

INX H

MOV D,M

RET

;

; LOGICAL TO PHYSICAL SECTOR TRANSLATION ROUTINE

PHYSEC: LDA SEKDSK ;GET THE DESIRED UNIT NUMBER

CALL FDSB0 ;GET PRMTBL POINTER

INX H ;POINT TO SECTOR SIZE

MOV D,M ;GET THE SECTOR SIZE

PUSH D

LDA SEKSEC ;GET THE LOGICAL SECTOR NUMBER

RAL

PHYSC1: ORA A ;RESET THE CARRY BIT

RAR ;CONVERT TO PHYSICAL SECTOR

DCR D

JP PHYSC1

INR A ;(A) NOW HAS PHYSICAL SECTOR #

STA SEKHST ;SET IT

POP D

RET

;

;COMPARE THE UNITS

COMP: LXI H,SEKDSK

MVI B,4

COMP1: LDAX D

SUB M

RNZ ;DONE IF NO COMPARE

INX H ;POINT TO NEXT ENTRY

INX D

DCR B

JNZ COMP1

DJNZ COMP1

RET

;

;

; THE FOLLOWING ROUTINES DO THE PRIMITIVE DISK ACCESSES.

; IN ALL CASES, ONE SECTOR OF DATA IS TRANSFERRED.

; IF THE DISK HAS NOT BEEN PREVIOUSLY ACCESSED,

; THESE ROUTINES WILL AUTOMATICALLY DETERMINE THE

; DISK TYPE (8" OR 5"), SINGLE OR DOUBLE DENSITY,

; AND SECTOR SIZE.

;

; BEFORE THE DESIRED DATA IS TRANSFERRED, THE DESIRED

; TRACK IS SEEKED OUT, THE DESIRED SECTOR AND SIDE IS

; SET, THEN THE ACTUAL DATA TRANSFER.

;

; UP TO TEN TRIES WILL BE ATTEMPTED BEFORE THE DATA

; TRANSFER IS ABORTED. ON RETURN TO THE CALLING

; ROUTINE, THE A REGISTER WILL CONTAIN A ZERO IF THE

; OPERATION WAS SUCCESSFUL, OR NON-ZERO IF NOT

; SUCCESSFUL. THE FLAG REGISTER WILL NOT NECESSARILY

; CORRESPOND WITH THE A REGISTER CONTENT.

;

;

DREAD: DB 3EH ;SIM. MVI A INSTR

DWRITE: XRA A ;SET WRITE FLAG XRA A OPCODE = 0AFH

STA RWFLG ;SAVE IT FOR LATER USE

MOV D,A

MVI C,TRIES ;NUMBER OF RETRIES

AGN: PUSH B

CALL SEEK

CZ RDWR

READ3: POP B

RZ

DCR C ;SEE IF ALL TRIES DONE

RZ ;YES, ERROR RETURN

ANI 10H ;SEE IF RNF OR SEEK ERROR

CNZ EOJB ;RESTORE DRIVE IF SO

JMP AGN

JR AGN

;

RDWR: MOV E,A ;SAVE COMMAND

LDA RWFLG

ORA A

MOV A,E ;REGET THE COMMAND

BIT 0,D

DI

JZ WRDAT ;WRITE IF ZERO

JRZ WRDAT

RDAT: STA CMND

OUT DCMMD ;DISK COMMAND PORT

READ1: IN DDATA ;READ THE DATA

MOV M,A ;PUT INTO BUFFER

INX H ;INCREMENT MEMORY POINTER

IN DDATA ;READ THE DATA

MOV M,A ;PUT INTO BUFFER

INX H ;INCREMENT MEMORY POINTER

IN DDATA ;READ THE DATA

MOV M,A ;PUT INTO BUFFER

INX H ;INCREMENT MEMORY POINTER

IN DDATA ;READ THE DATA

MOV M,A ;PUT INTO BUFFER

INX H ;INCREMENT MEMORY POINTER

DCR B

JNZ READ1

DJNZ READ1

CALL EOJ

ANI 9CH ;ISOLATE READ ERROR BITS

RET

;

WRDAT: ORI 20H ;MAKE INTO WRITE COMMAND

STA CMND

OUT DCMMD ;DISK COMMAND PORT

WRT1: MOV A,M ;GET DATA FROM BUFFER

OUT DDATA ;OUTPUT IT

INX H ;ADVANCE MEMORY POINTER

MOV A,M ;GET DATA FROM BUFFER

OUT DDATA ;OUTPUT IT

INX H ;ADVANCE MEMORY POINTER

MOV A,M ;GET DATA FROM BUFFER

OUT DDATA ;OUTPUT IT

INX H ;ADVANCE MEMORY POINTER

MOV A,M ;GET DATA FROM BUFFER

OUT DDATA ;OUTPUT IT

INX H ;ADVANCE MEMORY POINTER

DCR B

JNZ WRT1

DJNZ WRT1

JMP EOJ

JR EOJ

;

EOJB: MVI B,RSTR ;BASIS OF RESTORE COMMAND

EOJA: PUSH H ;SAVE (H,L)

PUSH B

CALL FDSB ;GET THE PRMTBL POINTER

POP B

DCX H ;POINT TO THE STEP RATE ENTRY

MOV A,M ;GET THE STEP RATE

POP H ;RESTORE (H,L)

ORA B ;ADD ON THE COMMAND

EOJC: STA CMND

OUT DCMMD ;DO THE COMMAND

EOJ: EI

IN DFLAG ;DISK FLAG PORT

RAR

JNC EOJ

JRNC EOJ

EOJ1: IN DSTAT ;GET THE DISK STATUS

STA STATUS

ANI 0FCH

IF MAXI

RP ;DONE IF DRIVE IS READY

LDA DISKNO ;GET DRIVE NUMBER

ADI 'A' ;CONVERT TO ASCII

STA DNRMSGA ;PUT IT INTO MESSAGE

PUSH H ;SAVE (H,L)

LXI H,DNRMSG

CALL PRTRD ;PRINT THE MESSAGE

POP H ;RESTORE (H,L)

JMP EOJ1 ; AND TRY AGAIN

JR EOJ1

ENDIF

IF NOT MAXI

RET

ENDIF

;

SEEK: CALL IDRD ;INSURE HEADER HAS BEEN READ

RNZ ;ERROR RETURN

SEEK1: LDA SECTOR ;SET THE SECTOR

OUT DSCTR ;DISK SECTOR PORT

IN DTRCK ;READ THE CURRENT TRACK SETTING

MOV C,A ;SAVE FOR A MOMENT

LDA TRACK ;GET DESIRED TRACK

CMP C ;SEE IF SEEK NEEDED

JZ RDWRT ;NO, PRESS ON

JRZ RDWRT

OUT DDATA ;SET THE SEEK TRACK

ORA A ;SEE IF TRACK 0 DESIRED

MVI C,0 ;NO AUTOWAIT WANTED

LDA CUNIT

CALL SU2 ;RESET THE DENSITY BIT IF SO

MVI B,SEEKV ;BUILD THE SEEK COMMAND

CALL EOJA ;DO THE SEEK

ANI 98H ;SEEK ERROR MASK

RNZ ;DONE IF SEEK ERROR

RDWRT: MVI C,80H ;AUTO-WAIT BIT

CALL SETUP

IN DFLAG ;DISK FLAG PORT

ANI 20H ;SEE IF HEAD IS LOADED

MVI A,4

JZ RDWRT1 ;JUMP IF NOT

JRZ RDWRT1

XRA A ;ELSE, RESET THE HEAD LOAD FLAG

RDWRT1: ADI RDSEC ;BUILD A READ SECTOR COMMAND

MOV C,A ;SAVE THE COMMAND IN C

IN DTRCK ;SEE IF ON TRACK 0

ORA A

JZ RDWRT3 ;SET SECTOR SIZE = 0

JRZ RDWRT3

INX H ;GET THE SECTOR SIZE

MOV A,M

RDWRT3: MOV B,A ;PUT IN B FOR LOOP CONTROL

MVI A,10H ;SECTOR BASE LENGTH

RDWRT0: ADD A

DCR B

JP RDWRT0

CANNOT USE JNC!

MOV B,A

MOV A,C

LHLD HSTBUF ;GET THE DMA ADDRESS

RDWRT2: CMP A ;CLEAR THE FLAGS

RET

;

IDRD5: MVI B,STEPI ;BUILD A STEP-IN COMMAND

CALL EOJA

IDRD: LHLD LUNIT

MOV A,H ;GET THE CUNIT VALUE

CMP L ;SEE IF SAME AS LUNIT

RZ ;RETURN IF SO

IDRD1: MVI C,80H ;SET THE AUTO-WAIT BIT

CALL SETUP

PUSH H ;SAVE POINTER

LXI H,HLWAIT ;WAIT FOR HEADS TO SETTLE

IDRD3: DCX H

MOV A,H

ORA A

JNZ IDRD3

JRNZ IDRD3

LXI H,IDSV ;SET UP TO READ ADDRESS

MVI B,2 ;SET UP TO READ 6(8) BYTES OF DATA

MVI A,RDADD ;READ ADDRESS COMMAND

DI

CALL RDAT

POP H ;RESTORE POINTER

JZ IDRD2 ;JUMP IF GOOD READ

JRZ IDRD2

MVI A,40H ;SEE IF DDEN IS SET

CMP M

RC ;TAKE THE ERROR IF SO

ORA M ;ELSE, TRY DDEN

MOV M,A

JMP IDRD

JR IDRD

;

IDRD2: IN DSCTR ;GET THE TRACK NUMBER

OUT DTRCK ;SET THE TRACK REGISTER

CPI 2 ;INSURE NOT ON TRACK 0 OR 1

JC IDRD5 ;JUMP IF SO

JRC IDRD5

MOV A,M ;REGET SELBITS

STA LUNIT ;UPDATE LAST USED UNIT

STA CUNIT

INX H ;SET THE SECTOR SIZE

LDA IDSV+3

MOV M,A

CMP A ;RESET ERROR FLAGS

RET

;

;SET UP DRIVE NUMBER

SETUP: CALL FDSB ;GET THE DISK SELECT BITS

JNZ SU0 ;YES, SKIP INIT CODE

JRNZ SU0

;

SETIT: STC ;DRIVE SELECT BIT

SET1: RAL ;SHIFT BIT INTO POSITION

DCR B

JNZ SET1 ;LOOP TIL BIT IS IN POSITION

DJNZ SET1

IF MINI

ORI 20H ;ADD ON MOTOR ON BIT

MOV M,A ;SAVE IT

OUT DCTRL ;SELECT THE DRIVE

ENDIF

IF BOTH

MVI A,RSTR OR 3 ;GET WORST CASE RESTORE COMMAND

CALL EOJC ;RESTORE THE DRIVE

IN BCTRL ;READ THE MINI TRK00 BIT

RAR ;ISOLATE IT

ENDIF

IF MINI

MVI A,STEP5 ;MINI STEP RATE BITS

ENDIF

IF BOTH

JNC SET2 ;JUMP IF MINI DRIVE

JRNC SET2

MOV A,M ;REGET THE SELBITS

ENDIF

IF MAXI

ORI 30H ;ADD ON THE 8", MOTOR ON BITS

OUT DCTRL

MOV M,A

IN BCTRL ;SEE IF DOUBLE-SIDED DRIVE

ANI 40H ;ISOLATE THE BIT

JZ SET2 ;SET 3 MS STEP RATE FOR DOUBLE-SIDED DRIVES

JRZ SET2

MVI A,STEP8 ;SET MAXI STEP RATE

ENDIF

SET2: DCX H ;POINT TO STEP RATE HOLD

MOV M,A ;SET THE STEP RATE

INX H ;RESET POINTER

SU0: IN DTRCK ;ELSE, SEE IF TRACK ZERO

ORA A

IN DDATA ;CLEAR OUT ANY JUNK

MOV A,M ;REGET THE SELBITS

SU2: JNZ SU1

JRNZ SU1

ANI 0BFH ;INSURE DDEN IS RESET

SU1: ORA C ;ADD ON AUTOWAIT BIT

OUT DCTRL ;OUTPUT THE SELBITS

LDA SIDE ;SET THE SIDE SELECT

OUT BCTRL

JMP EOJ1 ;GO INSURE A UNIT IS THERE

TOO FAR!

;

FDSB: LDA DISKNO ;GET THE SELECT UNIT

FDSB0: LXI H,PRMTBL-2 ;SEE IF DRIVE HAS BEEN ACTIVE

INR A ;OFFSET FOR LOOP CONTROL

MOV B,A ;SAVE FOR LATER USE

FDSB1: INX H ;OFFSET TABLE POINTER

INX H

INX H

DCR A

JNZ FDSB1 ;LOOP CONTROL

JRNZ FDSB1

MOV A,M ;READ THE SELECT BITS

ORA A

RET

;

;

;

;FIXED DATA TABLES FOR FOUR-DRIVE STANDARD

;IBM-COMPATIBLE 8" DISKS

;DISK PARAMETER HEADER FOR DISK 00

DPBASE: DW 0,0

DW 0,0

DW DIRBF,0

DW CHK00,ALL00

;DISK PARAMETER HEADER FOR DISK 01

DW 0,0

DW 0,0

DW DIRBF,0

DW CHK01,ALL01

;DISK PARAMETER HEADER FOR DISK 02

DW 0,0

DW 0,0

DW DIRBF,0

DW CHK02,ALL02

;DISK PARAMETER HEADER FOR DISK 03

DW 0,0

DW 0,0

DW DIRBF,0

DW CHK03,ALL03

;

;PARAMETER TABLE FOR DRIVE-UNIQUE CONSTANTS

PRMTBL: DB 0,0,0 ;DRIV 0 STP RAT, SLCT BYTS, SCTR SZ

DB 0,0,0 ; 1

DB 0,0,0 ; 2

DB 0,0,0 ; 3

;

;SECTOR TRANSLATE VECTOR

;

IF MAXI

SELTBL: DW T826,DP8S0

DW T815,DP8S1

DW T88,DP8S2

DW T84,DP8S2

DW T848,DP8D0

DW T826,DP8D1

DW T815,DP8D2

DW T88,DP8D3

;

DP8S0: DW 26 ;SECTORS PER TRACK

DB 3 ;BLOCK SHIFT FACTOR

DB 7 ;BLOCK MASK

DB 0 ;EXTENT MASK

DW 242 ;BLOCKS PER DISKETTE

DW 63 ;# DIRCTORY ENTRIES

DB 192 ;ALLOC 0

DB 0 ;ALLOC 1

DW 16 ;DIR CHECK VECTOR SIZE

DW 2 ;SYSTEM TRACK OFFSET

DP8S1: DW 30 ;SECTORS PER TRACK

DB 4 ;BLOCK SHIFT FACTOR

DB 15 ;BLOCK MASK

DB 1 ;EXTENT MASK

DW 139 ;BLOCKS PER DISKETTE

DW 63 ;# DIRCTORY ENTRIES

DB 128 ;ALLOC 0

DB 0 ;ALLOC 1

DW 16 ;DIR CHECK VECTOR SIZE

DW 2 ;SYSTEM TRACK OFFSET

DP8S2: DW 32 ;SECTORS PER TRACK

DB 4 ;BLOCK SHIFT FACTOR

DB 15 ;BLOCK MASK

DB 1 ;EXTENT MASK

DW 149 ;BLOCKS PER DISKETTE

DW 63 ;# DIRCTORY ENTRIES

DB 128 ;ALLOC 0

DB 0 ;ALLOC 1

DW 16 ;DIR CHECK VECTOR SIZE

DW 2 ;SYSTEM TRACK OFFSET

DP8D0: DW 48 ;SECTORS PER TRACK

DB 4 ;BLOCK SHIFT FACTOR

DB 15 ;BLOCK MASK

DB 1 ;EXTENT MASK

DW 195 ;BLOCKS PER DISKETTE

DW 63 ;# DIRECTORY ENTRIES

DB 128 ;ALLOC 0

DB 0 ;ALLOC 1

DW 16 ;DIR CHECK VECTOR SIZE

DW 2 ;SYSTEM TRACK OFFSET

DP8D1: DW 52 ;SECTORS PER TRACK

DB 4 ;BLOCK SHIFT FACTOR

DB 15 ;BLOCK MASK

DB 1 ;EXTENT MASK

DW 242 ;BLOCKS PER DISKETTE

DW 63 ;# DIRECTORY ENTRIES

DB 128 ;ALLOC 0

DB 0 ;ALLOC 1

DW 16 ;DIR CHECK VECTOR SIZE

DW 2 ;SYSTEM TRACK OFFSET

DP8D2: DW 60 ;SECTORS PER TRACK

DB 5 ;BLOCK SHIFT FACTOR

DB 31 ;BLOCK MASK

DB 3 ;EXTENT MASK

DW 139 ;BLOCKS PER DISKETTE

DW 127 ;# DIRECTORY ENTRIES

DB 128 ;ALLOC 0

DB 0 ;ALLOC 1

DW 32 ;DIR CHECK VECTOR SIZE

DW 2 ;SYSTEM TRACK OFFSET

DP8D3: DW 64 ;SECTORS PER TRACK

DB 4 ;BLOCK SHIFT FACTOR

DB 15 ;BLOCK MASK

DB 0 ;EXTENT MASK

DW 299 ;BLOCKS PER DISKETTE

DW 127 ;# DIRECTORY ENTRIES

DB 192 ;ALLOC 0

DB 0 ;ALLOC 1

DW 32 ;DIR CHECK VECTOR SIZE

DW 2 ;SYSTEM TRACK OFFSET

;

SELTBLA: DW T826,DP8D1

DW T815,DP8D2

DW T88,DP8D3

DW T84,DP8D3

DW T848,DP8D0A

DW T826,DP8D1A

DW T815,DP8D2A

DW T88,DP8D3A

;

DP8D0A: DW 96 ;SECTORS PER TRACK

DB 5 ;BLOCK SHIFT FACTOR

DB 31 ;BLOCK MASK

DB 3 ;EXTENT MASK

DW 225 ;BLOCKS PER DISKETTE

DW 127 ;# DIRECTORY ENTRIES

DB 128 ;ALLOC 0

DB 0 ;ALLOC 1

DW 32 ;DIR CHECK VECTOR SIZE

DW 2 ;SYSTEM TRACK OFFSET

DP8D1A: DW 104 ;SECTORS PER TRACK

DB 5 ;BLOCK SHIFT FACTOR

DB 31 ;BLOCK MASK

DB 3 ;EXTENT MASK

DW 242 ;BLOCKS PER DISKETTE

DW 127 ;# DIRECTORY ENTRIES

DB 128 ;ALLOC 0

DB 0 ;ALLOC 1

DW 32 ;DIR CHECK VECTOR SIZE

DW 2 ;SYSTEM TRACK OFFSET

DP8D2A: DW 120 ;SECTORS PER TRACK

DB 6 ;BLOCK SHIFT FACTOR

DB 63 ;BLOCK MASK

DB 7 ;EXTENT MASK

DW 139 ;BLOCKS PER DISKETTE

DW 127 ;# DIRECTORY ENTRIES

DB 128 ;ALLOC 0

DB 0 ;ALLOC 1

DW 32 ;DIR CHECK VECTOR SIZE

DW 2 ;SYSTEM TRACK OFFSET

DP8D3A: DW 128 ;SECTORS PER TRACK

DB 6 ;BLOCK SHIFT FACTOR

DB 63 ;BLOCK MASK

DB 7 ;EXTENT MASK

DW 149 ;BLOCKS PER DISKETTE

DW 127 ;# DIRECTORY ENTRIES

DB 128 ;ALLOC 0

DB 0 ;ALLOC 1

DW 32 ;DIR CHECK VECTOR SIZE

DW 2 ;SYSTEM TRACK OFFSET

;

T848: DB 48,48,1

T826: DB 26,13,6

T815: DB 15,15,4

T88: DB 8,8,3

T84: DB 4,2,2

ENDIF

;

IF MINI

MSELTBL: DW T518,DP5S0

DW T510,DP5S1

DW T55,DP5S1

DW T52,DP5S3

DW T529,DP5D0

DW T518,DP5D1

DW T510,DP5D2

DW T55,DP5D2

DP5S0: DW 18 ;SECTORS PER TRACK

DB 3 ;BLOCK SHIFT FACTOR

DB 7 ;BLOCK MASK

DB 0 ;EXTENT MASK

DW 71 ;BLOCKS PER DISKETTE

DW 63 ;# DIRECTORY ENTRIES

DB 192 ;ALLOC 0

DB 0 ;ALLOC 1

DW 16 ;DIR CHECK VECTOR SIZE

DW 3 ;SYSTEM TRACK OFFSET

DP5S1: DW 20 ;SECTORS PER TRACK

DB 3 ;BLOCK SHIFT FACTOR

DB 7 ;BLOCK MASK

DB 0 ;EXTENT MASK

DW 79 ;BLOCKS PER DISKETTE

DW 63 ;# DIRECTORY ENTRIES

DB 192 ;ALLOC 0

DB 0 ;ALLOC 1

DW 16 ;DIR CHECK VECTOR SIZE

DW 3 ;SYSTEM TRACK OFFSET

DP5S3: DW 16 ;SECTORS PER TRACK

DB 3 ;BLOCK SHIFT FACTOR

DB 7 ;BLOCK MASK

DB 0 ;EXTENT MASK

DW 63 ;BLOCKS PER DISKETTE

DW 63 ;# DIRECTORY ENTRIES

DB 192 ;ALLOC 0

DB 0 ;ALLOC 1

DW 16 ;DIR CHECK VECTOR SIZE

DW 3 ;SYSTEM TRACK OFFSET

DP5D0: DW 29 ;SECTORS PER TRACK

DB 3 ;BLOCK SHIFT FACTOR

DB 7 ;BLOCK MASK

DB 0 ;EXTENT MASK

DW 115 ;BLOCKS PER DISKETTE

DW 63 ;# DIRECTORY ENTRIES

DB 192 ;ALLOC 0

DB 0 ;ALLOC 1

DW 16 ;DIR CHECK VECTOR SIZE

DW 3 ;SYSTEM TRACK OFFSET

DP5D1: DW 36 ;SECTORS PER TRACK

DB 4 ;BLOCK SHIFT FACTOR

DB 15 ;BLOCK MASK

DB 1 ;EXTENT MASK

DW 71 ;BLOCKS PER DISKETTE

DW 63 ;# DIRECTORY ENTRIES

DB 128 ;ALLOC 0

DB 0 ;ALLOC 1

DW 16 ;DIR CHECK VECTOR SIZE

DW 3 ;SYSTEM TRACK OFFSET

DP5D2: DW 40 ;SECTORS PER TRACK

DB 4 ;BLOCK SHIFT FACTOR

DB 15 ;BLOCK MASK

DB 1 ;EXTENT MASK

DW 79 ;BLOCKS PER DISKETTE

DW 63 ;# DIRECTORY ENTRIES

DB 128 ;ALLOC 0

DB 0 ;ALLOC 1

DW 16 ;DIR CHECK VECTOR SIZE

DW 3 ;SYSTEM TRACK OFFSET

T529: DB 29,29,7

T518: DB 18,9,4

T510: DB 10,10,3

T55: DB 5,5,2

T52: DB 2,2,1

ENDIF

;

;END OF FIXED TABLES

;

;

CO: LDA IOBYTE

ANI 3 ;ISOLATE CONSOLE ASGT

JZ TTYOUT ;TTY ACTIVE

CPI 2

JM CRTOUT ;CRT ACTIVE

JNZ CUSO1 ;USER CONSOLE 1 ACTIVE

;

LO: LDA IOBYTE

ANI 0C0H ;ISOLATE LIST ASGT

JZ TTYOUT ;TTY ACTIVE

CPI 80H

JM CRTOUT ;CRT ACTIVE

JZ LPRT ;LINE PRINTER ACTIVE

JMP LUSE1 ;USER PRINTER 1 ACTIVE

;

CSTS: LDA IOBYTE

ANI 3 ;ISOLATE CONSOLE ASGT

JZ TTST ;TTY ACTIVE

CPI 2

JM CRTST ;CRT ACTIVE

JNZ CUST1 ;USER CONSOLE 1 ACTIVE

;

BATST: LDA IOBYTE

ANI 0CH ;ISOLATE BATCH ASGT

JZ TTST ;TTY ACTIVE

CPI 8

JM PTRST ;PAPER TAPE READER ACTIVE

JZ RUST1 ;USER READER 1 ACTIVE

JMP RUST2 ;USER READER 2 ACTIVE

;

CI: LDA IOBYTE

ANI 3 ;ISOLATE CONSOLE ASGT

JZ TTYIN ;KBD ACTIVE

CPI 2

JM CRTIN ;CRT ACTIVE

JNZ CUSI1 ;USER CONSOLE 1 ACTIVE

;

RI: LDA IOBYTE

ANI 0CH ;ISOLATE BATCH ASGT

JZ TTYRDR ;TTY ACTIVE

CPI 8

JM PTRIN ;PAPER TAPE READER ACTIVE

JZ RUSI1 ;USER READER 1 ACTIVE

JMP RUSI2 ;USER READER 2 ACTIVE

;

LSTAT: LDA IOBYTE

ANI 0C0H ;ISOLATE THE LIST DEVICE ASSIGNMENT

JZ TTOST

CPI 80H

JM HSPST

JZ LPRTS

JMP LUST1

;

PO: LDA IOBYTE

ANI 30H ;ISOLATE PUNCH ASGT

JZ TTPNCH ;TTY ACTIVE

CPI 20H

JM HSP ;HIGH SPEED PUNCH ACTIVE

JZ PUSO1 ;USER PUNCH 1 ACTIVE

JMP PUSO2 ;USER PUNCH 2 ACTIVE

;

; ROUTINE CONI READS THE CONSOLE AND STRIPS OFF THE ASCII

; PARITY BIT.

;

PRTRD: CALL PMSG

CONI: CALL CI ;GET THE NEXT CHARACTER

ANI 7FH ;STRIP OFF THE PARITY BIT

RTS: RET

;

; ROUTINE PRTWD PRINTS AN ASCII STRING ONTO THE CONSOLE.

; THE STRING MUST BE TERMINATED BY BIT 7 SET IN THE

; LAST CHARACTER OF THE STRING. THE STRING WILL START

; A NEW LINE (EP = PRTWD) OR CONTINUE ON THE SAME

; LINE (EP = PRTWA)

;

PRTWD: CALL CRLF ;START A NEW LINE

PRTWA: PUSH B ;SAVE (B,C)

PRTA: MOV C,M ;GET NEXT CHARACTER FROM MEMORY

CALL CO ;OUTPUT IT

INX H ;INCREMENT MEMORY POINTER

MOV A,C

RLC ;TEST FOR BIT 7 DELIMITER

JNC PRTA ;NO DELIMITER, GO DO NEXT CHARACTER

PRTB: POP B ;RESTORE (B,C)

RET

;

; ROUTINE CRLF GENERATES A CARRIAGE RETURN, LINE FEED

; SEQUENCE ON THE CURRENT CONSOLE TO START A NEW LINE

; IT INCLUDES TWO NULL CHARACTERS FOR TTY TYPE

; DEVICES FOR THE HEAD MOVEMENT TIME.

;

PMSG: CALL PRTWD

CRLF: PUSH H ;SAVE THE CONTENTS OF (H,L)

CRLFA: LXI H,CRMSG ;ADDRESS OF CR,LF MESSAGE

CALL PRTWA ; OUTPUT IT

POP H ;RESTORE (H,L)

RET

;

IOER: XRA A ;RESET IOBYTE

STA IOBYTE

LXI H,IOMSG ;ADDRESS OF IO ERROR MESSAGE

JMP COMERR

;

RSTER: LXI H,RSTMSG ;GET ADDRESS OF RESTART ERROR MSG

COMERR: CALL PMSG ;PRINT IT ON NEW LINE

JMP WBOOTV ;GO TO WARM BOOT

;

IOMSG: DB BELL,'I/O ASGT ERRO','R'+80H

RSTMSG: DB BELL,'RST ER','R'+80H

DNRMSG: DB BELL,'DRIVE '

DNRMSGA: DB 0,' NOT READ','Y'+80H

BOTMSG: DB BELL,'CANNOT BOO','T'+80H

CRMSG: DB CR,LF,0,80H

;

; I/O DRIVERS FOR THE 8250 ASYNC COMM ELEMENT

;

TTST: IN SLSTAT ;GET 8250 LINE STATUS

ANI RXRDY ;SEE IF RECEIVE DATA AVAILABLE

RZ ;RETURN IF NOT

ADI 0FFH AND NOT RXRDY ;FLAG THAT DATA IS AVAILABLE

RET

;

TTYIN: CALL TTST ;GET 8250 LINE STATUS

JZ TTYIN ;LOOP UNTIL DATA IS IN

IN SDATA ;READ THE DATA

RET

;

TTOST: IN SLSTAT ;GET 8250 LINE STATUS

ANI TXMTY ;ISOLATE TX BUFFER EMPTY BIT

RZ ;RETURN IF NOT EMPTY

ADI 0FFH AND NOT TXMTY ;FLAG THE EMPTY STATE

RET

;

TTYOUT: CALL TTOST ;GET 8250 LINE STATUS

JZ TTYOUT ;WAIT UNTIL ONE OF THE REGISTERS EMPTIES

MOV A,C ;MOVE THE DATA OVER

OUT SDATA ;OUTPUT THE DATA

RET

;

; EQUATES FOR ADDITIONAL CONSOLE DEVICES

;

CRTIN: EQU IOER

CRTOUT: EQU IOER

CRTST: EQU IOER

CUSI1: EQU IOER ;UNASSIGNED USER CONSOLE (INPTUT)

CUSO1: EQU IOER ;UNASSIGNED USER CONSOLE (OUPTUT)

CUST1: EQU IOER

;

; EQUATES FOR ADDITIONAL PAPER TAPE PUNCH DEVICES

;

TTPNCH: EQU TTYOUT ;UNASSIGNED TELETYPE PUNCH

HSP: EQU IOER ;UNASSIGNED HIGH SPEED PUNCH

HSPST: EQU IOER ;UNASSIGNED HIGH SPEED PUNCH STATUS

PUSO1: EQU IOER ;UNASSIGNED USER PUNCH 1

PUSO2: EQU IOER ;UNASSIGNED USER PUNCH 2

;

; EQUATES FOR ADDITIONAL LIST DEVICES

;

LPRT: EQU IOER ;UNASSIGNED LINE PRINTER

LPRTS: EQU IOER ;UNASSIGNED LINE PRINTER STATUS

LUSE1: EQU IOER ;UNASSIGNED LIST DEVICE 1

LUST1: EQU IOER ;UNASSIGNED LIST DEVICE 1 STATUS

;

; EQUATES FOR ADDITIONAL PAPER TAPE READER DEVICES

;

TTYRDR: EQU TTYIN ;UNASSIGNED TELETYPE PAPER TAPE READER

PTRIN: EQU IOER ;UNASSIGNED HIGH SPEED PAPER TAPE READER

PTRST: EQU IOER ;UNASSIGNED HS PTR STATUS

RUSI1: EQU IOER ;UNASSIGNED PAPER TAPE READER 1

RUST1: EQU IOER ;UNASSIGNED PAPER TAPE READER 1 (STATUS)

RUSI2: EQU IOER ;UNASSIGNED PAPER TAPE READER 2

RUST2: EQU IOER ;UNASSIGNED PAPER TAPE READER 2 (STATUS)

;

; ONE-TIME CODE (USED ONLY DURING COLD BOOT)

;

BOOT: IN DCTRL ;SEE IF AUTO-BOOT IN PROGRESS

ANI 40H ;ISOLATE THE BOOT BIT

JNZ BOOTA ;JUMP IF NOT AUTO-BOOT

MVI A,0FH ;SET MODEM CONTROL REGISTER

OUT SMDMCT

MVI A,83H ;SET BAUD RATE DIVISOR ACCESS

OUT SLCTRL

MVI A,SBAUD/256 ;SET DIVISOR HIGH BYTE

OUT SINTEN

MVI A,SBAUD MOD 256

OUT SDATA ;SET DIVISOR LOW BYTE

MVI A,3 ;SET 8250 LINE CONTROL

OUT SLCTRL

XRA A ;SET HANDSHAKE LINES ACTIVE

OUT SINTEN

OUT SLSTAT

BOOTA: LXI H,LOGMSG ;SIGN ON TO THE SYSTEM

CALL PMSG

LXI H,0

SHLD IOBYTE ;SET IOBYTE, CDISK

JMP BOOT0 ;GO DO THE BOOT COMMON CODE

;

LOGMSG: DB MSIZE/10+'0',MSIZE MOD 10 + '0'

DB 'k CP/M Vers '

DB VERS/10+'0','.',VERS MOD 10+'0',80H

;

ORG BOOT

;

;THE REMAINDER OF THE CCBIOS IS RESERVED UNINITIALIZED

;DATA AREA, AND DOES NOT NEED TO BE A PART OF THE

;SYSTEM MEMORY IMAGE (THE SPACE MUST BE AVAILABLE,

;HOWEVER, BETWEEN "BEGDAT" AND "ENDDAT").

;

BEGDAT EQU $ ;BEGINNING OF DATA AREA

;

SEKDSK: DS 1

SEKTRK: DS 1

SEKHST: DS 1

SEKSID: DS 1

SEKSEC: DS 1

SEKSEL: DS 1

;

DMAAD: DS 2 ;DIRECT MEMORY ADDRESS

;

HSTACT: DS 1

HSTWRT: DS 1

;

UNADSK: DS 1

UNATRK: DS 1

UNASEC: DS 1

UNASID: DS 1

UNACNT: DS 1

CPMSEC: DS 1

NXTSCT: DS 1

;

ERFLAG: DS 1

RSFLAG: DS 1

READOP: DS 1

WRTYPE: DS 1

;

;SCRATCH RAM AREA FOR BDOS USE

;

DIRBF: DS 128 ;SCRATCH DIRECTORY AREA

ALL00: DS 38 ;ALLOCATION VECTOR 0

ALL01: DS 38 ;ALLOCATION VECTOR 1

ALL02: DS 38 ;ALLOCATION VECTOR 2

ALL03: DS 38 ;ALLOCATION VECTOR 3

CHK00: DS 32 ;CHECK VECTOR 0

CHK01: DS 32 ;CHECK VECTOR 1

CHK02: DS 32 ;CHECK VECTOR 2

CHK03: DS 32 ;CHECK VECTOR 3

;

DBUF: DS 1024 ;DISK BUFFER

;

ENDDAT EQU $ ;END OF DATA AREA

DATSIZ EQU $-BEGDAT;SIZE OF DATA AREA

;

CONST: EQU CSTS

CONOUT: EQU CO

LIST: EQU LO

PUNCH: EQU PO

READER: EQU RI

LISTST: EQU LSTAT

END