```
*****************************************************************
*                                                               *
* CBIOS FOR CP/M VER 2.2 FOR DISK JOCKEY 2D CONTROLLER (ALL      *
* REVS, AND MODELS A & B). HANDLES DISKETTES WITH SECTOR SIZES   *
* OF 128 BYTES SINGLE DENSITY, 256, 512, 1024 BYTES DOUBLE       *
* DENSITY. THERE ARE CONDITIONAL ASSEMBLIES FOR DISKUS HARD      *
* DISK CONTROLLER.                                               *
*                                                               *
* WRITTEN BY BOBBY DALE GIFFORD.                                 *
* 12/8/80                                                        *
*                                                               *
* CUSTOMIZED BY JAY O'BRIEN                                      *
* 1/16/82                                                        *
*                                                               *
* DISK MAP OF SECTORS USED BY COLD BOOT, WARM BOOT, FIRMWARE,    *
* AND CP/M:                                                      *
*                                                               *
* TRK 0 SEC  1 = FIRST SECTOR OF COLD BOOT.         E700H        *
*       0        2 = COLD BOOT 256.                    80H        *
*       0        3 = COLD BOOT 512.                    80H        *
*       0        4 = COLD BOOT 1024.                   80H        *
*       0        5 = WARM BOOT 256.                    80H        *
*       0        6 = WARM BOOT 512.                    80H        *
*       0        7 = WARM BOOT 1024.                   80H        *
*       0        8 = COLD/WARM BOOT.                 2C00H        *
*       0        9 = FIRMWARE.                       E400H        *
*       0       10 = FIRMWARE+80H.                   E480H        *
*       0       11 = FIRMWARE+100H.                  E500H        *
*       0       12 = FIRMWARE+180H.                  E580H        *
*       0       13 = FIRMWARE+200H.                  E600H        *
*       0       14 = FIRMWARE+280H.                  E680H        *
*       0       15 = FIRMWARE+300H.                  E700H        *
*       0       16 = FIRMWARE+380H.                  E780H        *
*       0       17 = CCP.                            2700H        *
*       0       10 = CCP+80H.                        2780H        *
*       0       12 = CCP+100H.                       2800H        *
*       0       14 = CCP+180H.                       2880H        *
*       0       16 = CCP+200H.                       2900H        *
*       0       18 = CCP+280H.                       2980H        *
*       0       20 = CCP+300H.                       2A00H        *
*       0       22 = CCP+380H.                       2A80H        *
*       0       24 = CCP+400H.                       2B00H        *
*       0       26 = CCP+480H.                       2B80H        *
*       1          = REST OF CP/M.              2C00H-4FFFH       *
*                                                               *
*****************************************************************

        TITLE   '*** Cbios For CP/M Ver. 2.2 ***'

*****************************************************************
*                                                               *
* THE FOLLOWING REVISION NUMBER IS IN REFERENCE TO THE CP/M      *
* 2.2 CBIOS.                                                     *
*                                                               *
*****************************************************************
```

Handwritten margin notes:

1/16/82
CBIOS5.PRN

CBIOS 5AB = V10-X vice MSDV
changes in Ned NO STARTUP

CBIOS5B IS SAME BUT
H/D IS ~ not 9/8/82

CBIOS5B
~~First file~~

Working on
Bios short
Problem - see page a 10/11

```
001C =          REVNUM  EQU     28                      ;CBIOS REVISION NUMBER
0016 =          CPMREV  EQU     22                      ;CP/M REVISION NUMBER


                ********************************************************************
                *                                                                *
                * THE FOLLOWING EQUATES SET UP THE RELATIONSHIP BETWEEN THE       *
                * 2D FLOPPIES AND THE HARD DISK CONTROLLERS.                      *
                *                                                                *
                ********************************************************************

0000 =          FIRST   EQU     0                       ;0 = FLOPPIES ARE A,B,C,D DRIVES AND
                                                        ;        HARD DISK ARE E,F,G,H
                                                        ;1 = HARD DISKS ARE A,B,C,D DRIVES AND
                                                        ;        FLOPPIES ARE E,F,G,H
0001 =          MAXHD   EQU     1                       ;SET TO NUMBER OF HARD DISKS
0002 =          MAXFLOP EQU     2                       ;SET TO NUMBER OF FLOPPIES

0001 =          M26     EQU     1                       ;SET ONLY ONE OF THESE VARIABLES
0000 =          M20     EQU     0
0000 =          M10     EQU     0

                        IF      M10 OR M20
                SDELAY  EQU     0                       ;SOFTWARE HEAD SETTLE DELAY (0 = NO, 1 = YES)
                        ELSE
0001 =          SDELAY  EQU     1
                        ENDIF

001A =          MREV    EQU     26*M26+20*M20+10*M10     ;HARD DISK TYPE
0003 =          LOGDSK  EQU     3*M26+3*M20+2*M10        ;LOGICAL DISKS PER DRIVE
0020 =          HDSPT   EQU     32*M26+21*M20+21*M10     ;SECTORS PER TRACK


                ********************************************************************
                *                                                                *
                * THE FOLLOWING EQUATES RELATE THE THINKER TOYS 2D CONTROLLER.    *
                * IF THE CONTROLLER IS NON STANDARD (0E000H) ONLY THE ORIGIN      *
                * EQUATE NEED BE CHANGED. THIS VERSION OF THE CBIOS WILL WORK     *
                * WITH 2D CONTROLLER BOARDS REV 0, 1, 3, 3.1, 4, MODEL B.         *
                *                                                                *
                ********************************************************************


                        IF      MAXFLOP NE 0    ;INCLUDE DISCUS 2D ?
E000 =          ORIGIN  EQU     0E000H
E400 =          DJRAM   EQU     ORIGIN+400H     ;DISK JOCKEY 2D RAM ADDRESS
E400 =          DJBOOT  EQU     DJRAM           ;DISK JOCKEY 2D INITIALIZATION
E003 =          DJCIN   EQU     ORIGIN+3H       ;DISK JOCKEY 2D CHARACTER INPUT ROUTINE
E006 =          DJCOUT  EQU     ORIGIN+6H       ;DISK JOCKEY 2D CHARACTER OUTPUT ROUTINE
E409 =          DJHOME  EQU     DJRAM+9H        ;DISK JOCKEY 2D TRACK ZERO SEEK
E40C =          DJTRK   EQU     DJRAM+0CH       ;DISK JOCKEY 2D TRACK SEEK ROUTINE
E40F =          DJSEC   EQU     DJRAM+0FH       ;DISK JOCKEY 2D SET SECTOR ROUTINE
E412 =          DJDMA   EQU     DJRAM+012H      ;DISK JOCKEY 2D SET DMA ADDRESS
E415 =          DJREAD  EQU     DJRAM+15H       ;DISK JOCKEY 2D READ ROUTINE
E418 =          DJWRITE EQU     DJRAM+18H       ;DISK JOCKEY 2D WRITE ROUTINE
E41B =          DJSEL   EQU     DJRAM+1BH       ;DISK JOCKEY 2D SELECT DRIVE ROUTINE
E021 =          DJTSTAT EQU     ORIGIN+21H      ;DISK JOCKEY 2D TERMINAL STATUS ROUTINE
```

```
E427 =              DJSTAT    EQU    DJRAM+27H        ;DISK JOCKEY 2D STATUS ROUTINE
E42A =              DJERR     EQU    DJRAM+2AH        ;DISK JOCKEY 2D ERROR, FLASH LED
E42D =              DJDEN     EQU    DJRAM+2DH        ;DISK JOCKEY 2D SET DENSITY ROUTINE
E430 =              DJSIDE    EQU    DJRAM+30H        ;DISK JOCKEY 2D SET SIDE ROUTINE
0008 =              DBLSID    EQU    8                ;SIDE BIT FROM CONTROLLER
                              ENDIF


           *****************************************************************
           *                                                               *
           * THE FOLLOWING EQUATES ARE FOR THE DISKUS HARD DISK WANTED.     *
           *                                                               *
           *****************************************************************


                              IF     MAXHD NE 0       ;WANT HARD DISK INCLUDED ?
0050 =              HDORG     EQU    50H              ;HARD DISK CONTROLLER ORIGIN
0050 =              HDSTAT    EQU    HDORG            ;HARD DISK STATUS
0050 =              HDCNTL    EQU    HDORG            ;HARD DISK CONTROL
0053 =              HDDATA    EQU    HDORG+3          ;HARD DISK DATA
0052 =              HDFUNC    EQU    HDORG+2          ;HARD DISK FUNCTION
0051 =              HDCMND    EQU    HDORG+1          ;HARD DISK COMMAND
0051 =              HDRESLT   EQU    HDORG+1          ;HARD DISK RESULT
0002 =              RETRY     EQU    2                ;RETRY BIT OF RESULT
0001 =              TKZERO    EQU    1                ;TRACK ZERO BIT OF STATUS
0002 =              OPDONE    EQU    2                ;OPERACTION DONE BIT OF STATUS
0004 =              COMPLT    EQU    4                ;COMPLETE BIT OF STATUS
0008 =              TMOUT     EQU    8                ;TIME OUT BIT OF STATUS
0010 =              WFAULT    EQU    10H              ;WRITE FAULT BIT OF STATUS
0020 =              DRVRDY    EQU    20H              ;DRIVE READY BIT OF STATUS
0040 =              INDEX     EQU    40H              ;INDEX BIT OF STATUS
0004 =              PSTEP     EQU    4                ;STEP BIT OF FUNCTION
00FB =              NSTEP     EQU    0FBH             ;STEP BIT MASK OF FUNCTION
0004 =              HDRLEN    EQU    4                ;SECTOR HEADER LENGTH
0200 =              SECLEN    EQU    512              ;SECTOR DATA LENGTH
000F =              WENABL    EQU    0FH              ;WRITE ENABLE
000B =              WRESET    EQU    0BH              ;WRITE RESET OF FUNCTION
0005 =              SCENBL    EQU    5                ;CONTROLLER CONTROL
0007 =              DSKCLK    EQU    7                ;DISK CLOCK FOR CONTROL
00F7 =              MDIR      EQU    0F7H             ;DIRECTION MASK FOR FUNCTION
00FC =              NULL      EQU    0FCH             ;NULL COMMAND
0000 =              IDBUFF    EQU    0                ;INITIALIZE DATA COMMAND
0008 =              ISBUFF    EQU    8                ;INITIALIZE HEADER COMMAND
0001 =              RSECT     EQU    1                ;READ SECTOR COMMAND
0005 =              WSECT     EQU    5                ;WRITE SECTOR COMMAND
                              ENDIF


           *****************************************************************
           *                                                               *
           * CP/M SYSTEM EQUATES. IF RECONFIGURATION OF THE CP/M SYSTEM     *
           * IS BEING DONE, THE CHANGES CAN BE MADE TO THE FOLLOWING        *
           * EQUATES.                                                       *
           *                                                               *
           *****************************************************************


0038 =              MSIZE     EQU    56               ;MEMORY SIZE OF TARGET CP/M
9000 =              BIAS      EQU    (MSIZE-20)*1024   ;MEMORY OFFSET FROM 20K SYSTEM
B700 =              CCP       EQU    2700H+BIAS       ;CONSOLE COMMAND PROCESSOR
```

```
BF00 =          BDOS    EQU     CCP+800H        ;BDOS ADDRESS
CD00 =          BIOS    EQU     CCP+1600H       ;CBIOS ADDRESS
5A00 =          OFFSETC EQU     2700H-BIOS      ;OFFSET FOR SYSGEN
0004 =          CDISK   EQU     4               ;ADDRESS OF LAST LOGGED DISK
0080 =          BUFF    EQU     80H             ;DEFAULT BUFFER ADDRESS
0100 =          TPA     EQU     100H            ;TRANSIENT MEMORY
00C0 =          INTIOBY EQU     192             ;INITIAL IOBYTE
0003 =          IOBYTE  EQU     3               ;IOBYTE LOCATION
0000 =          WBOT    EQU     0               ;WARM BOOT JUMP ADDRESS
0005 =          ENTRY   EQU     5               ;BDOS ENTRY JUMP ADDRESS


        *****************************************************************
        *                                                               *
        * THE FOLLOWING ARE INTERNAL CBIOS EQUATES. MOST ARE MISC.       *
        * CONSTANTS.                                                     *
        *                                                               *
        *****************************************************************


000A =          RETRIES EQU     10              ;MAX RETRIES ON DISK I/O BEFORE ERROR
000D =          ACR     EQU     0DH             ;A CARRIAGE RETURN
000A =          ALF     EQU     0AH             ;A LINE FEED
0019 =          CLEAR   EQU     19H             ;CLEAR SCREEN FOR MSDV V10-X
0003 =          AETX    EQU     3               ;ETX CHARACTER
0006 =          AACK    EQU     6               ;ACK CHARACTER
E800 =          MSDV    EQU     0E800H          ;VIDEO DRIVER FOR MSDV-100 VIDEO BOARD


        *****************************************************************
        *                                                               *
        * THE JUMP TABLE BELOW MUST REMAIN IN THE SAME ORDER, THE        *
        * ROUTINES MAY BE CHANGED, BUT THE FUNCTION EXECUTED MUST BE      *
        * THE SAME.                                                      *
        *                                                               *
        *****************************************************************


CD00                    ORG     BIOS            ;CBIOS STARTING ADDRESS

CD00 C3D2D5             JMP     CBOOT           ;COLD BOOT ENTRY POINT
CD03 C3B4CE     WBOOTE  JMP     WBOOT           ;WARM BOOT ENTRY POINT
CD06 C336CD             JMP     CONST           ;CONSOLE STATUS ROUTINE
CD09 C342CD             JMP     CONIN           ;CONSOLE INPUT
CD0C C357CD     COUT    JMP     CONOUT          ;CONSOLE OUTPUT
CD0F C377CD             JMP     LIST            ;LIST DEVICE OUTPUT
CD12 C36CCD             JMP     PUNCH           ;PUNCH DEVICE OUTPUT
CD15 C362CD             JMP     READER          ;READER DEVICE INPUT
CD18 C349CF             JMP     HOME            ;HOME DRIVE
CD1B C38BCF             JMP     SETDRV          ;SELECT DISK
CD1E C34BCF             JMP     SETTRK          ;SET TRACK
CD21 C33DCF             JMP     SETSEC          ;SET SECTOR
CD24 C343CF             JMP     SETDMA          ;SET DMA ADDRESS
CD27 C391D0             JMP     READ            ;READ THE DISK
CD2A C38AD0             JMP     WRITE           ;WRITE THE DISK
CD2D C382CD             JMP     LISTST          ;LIST DEVICE STATUS
CD30 C350CF             JMP     SECTRAN         ;SECTOR TRANSLATION

                        IF      MAXFLOP NE 0
CD33 C31BE4     DJDRV   JMP     DJSEL           ;HOOK FOR SINGLE.COM PROGRAM
```

```
                    ELSE
                    JMP     DONOP
                    ENDIF

            ****************************************************************
            *                                                              *
            * TERMINAL DRIVER ROUTINES. IOBYTE IS INITIALIZED BY THE COLD  *
            * BOOT ROUTINE, TO MODIFY, CHANGE THE "INTIOBY" EQUATE. THE    *
            * I/O ROUTINES THAT FOLLOW ALL WORK EXACTLY THE SAME WAY. USING *
            * IOBYTE, THEY OBTAIN THE ADDRESS TO JUMP TO IN ORDER TO EXECUTE*
            * THE DESIRED FUNCTION. THERE IS A TABLE WITH FOUR ENTRIES FOR  *
            * EACH OF THE POSSIBLE ASSIGNMENTS FOR EACH DEVICE. TO MODIFY   *
            * THE I/O ROUTINES FOR A DIFFERENT I/O CONFIGURATION, JUST      *
            * CHANGE THE ENTRIES IN THE TABLES.                            *
            *                                                              *
            ****************************************************************

E003 =              CITTY   EQU     DJCIN           ;INPUT FROM THE DISK JOCKEY 2D
E006 =              COTTY   EQU     DJCOUT          ;OUTPUT TO THE DISK JOCKEY 2D

            ****************************************************************
            *                                                              *
            * CONST: GET THE STATUS FOR THE CURRENTLY ASSIGNED CONSOLE     *
            *        DEVICE. THE CONSOLE DEVICE CAN BE GOTTEN FROM IOBYTE, *
            *        THEN A JUMP TO THE CORRECT CONSOLE STATUS ROUTINE IS  *
            *        PERFORMED.                                            *
            *                                                              *
            ****************************************************************

CD36 21B0CD         CONST   LXI     H,CSTBLE        ;BEGINNING OF JUMP TABLE
CD39 C348CD                 JMP     CONIN1          ;SELECT CORRECT JUMP

            ****************************************************************
            *                                                              *
            * CSREADER: IF THE CONSOLE IS ASSIGNED TO THE READER THEN A    *
            *           JUMP WILL BE MADE HERE, WHERE ANOTHER JUMP WILL    *
            *           OCCUR TO THE CORRECT READER STATUS.                *
            *                                                              *
            ****************************************************************

CD3C 21B8CD         CSREADR LXI     H,CSRTBLE       ;BEGINNING OF READER STATUS TABLE
CD3F C365CD                 JMP     READERA

            ****************************************************************
            *                                                              *
            * CONIN: TAKE THE CORRECT JUMP FOR THE CONSOLE INPUT ROUTINE.  *
            *        THE JUMP IS BASED ON THE TWO LEAST SIGNIFICANT BITS OF *
            *        IOBYTE.                                               *
            *                                                              *
            ****************************************************************

CD42 CD04D1         CONIN   CALL    FLUSH           ;FLUSH THE DISK BUFFER
CD45 2188CD                 LXI     H,CITBLE        ;BEGINNING OF CHARACTER INPUT TABLE

            *
            * ENTRY AT CONIN1 WILL DECODE THE TWO LEAST SIGNIFICANT BITS
```

```
                    * OF IOBYTE. THIS IS USED BY CONIN,CONOUT, AND CONST.
                    *

CD48 3A0300        CONIN1  LDA     IOBYTE
CD4B 17                    RAL


                    *
                    * ENTRY AT SELDEV WILL FORM AN OFFSET INTO THE TABLE POINTED
                    * TO BY H&L AND THEN PICK UP THE ADDRESS AND JUMP THERE.
                    *

CD4C E606          SELDEV  ANI     6H              ;STRIP OFF UNWANTED BITS
CD4E 1600                  MVI     D,0             ;FORM OFFSET
CD50 5F                    MOV     E,A
CD51 19                    DAD     D               ;ADD OFFSET
CD52 7E                    MOV     A,M             ;PICK UP HIGH BYTE
CD53 23                    INX     H
CD54 66                    MOV     H,M             ;PICK UP LOW BYTE
CD55 6F                    MOV     L,A             ;FORM ADDRESS
CD56 E9                    PCHL                    ;GO THERE !

                    *******************************************************************
                    *                                                                 *
                    * CONOUT: TAKE THE PROPER BRANCH ADDRESS BASED ON THE TWO LEAST *
                    *         SIGNIFICANT BITS OF IOBYTE.                             *
                    *                                                                 *
                    *******************************************************************

CD57 C5            CONOUT  PUSH    B               ;SAVE THE CHARACTER
CD58 CD04D1                CALL    FLUSH           ;FLUSH THE DISK BUFFER
CD5B C1                    POP     B               ;RESTORE THE CHARACTER
CD5C 2190CD                LXI     H,COTBLE        ;BEGINNING OF THE CHARACTER OUT TABLE
CD5F C348CD                JMP     CONIN1          ;DO THE DECODE

                    *******************************************************************
                    *                                                                 *
                    * READER: SELECT THE CORRECT READER DEVICE FOR INPUT. THE         *
                    *         READER IS SELECTED FROM BITS 2 AND 3 OF IOBYTE.         *
                    *                                                                 *
                    *******************************************************************

CD62 21A8CD        READER  LXI     H,RTBLE         ;BEGINNING OF READER INPUT TABLE

                    *
                    * ENTRY AT READERA WILL DECODE BITS 2 & 3 OF IOBYTE, USED
                    * BY CSREADER.
                    *

CD65 3A0300        READERA LDA     IOBYTE

                    *
                    * ENTRY AT READER1 WILL SHIFT THE BITS INTO POSITION, USED
                    * BY LIST AND PUNCH.
                    *

CD68 1F            READR1  RAR
```

```
CD69 C34CCD              JMP      SELDEV

         ******************************************************************
         *                                                                *
         * PUNCH: SELECT THE CORRECT PUNCH DEVICE. THE SELECTION COMES     *
         *        FROM BITS 4&5 OF IOBYTE.                                 *
         *                                                                *
         ******************************************************************

CD6C 21A0CD     PUNCH    LXI      H,PTBLE         ;BEGINNING OF PUNCH TABLE
CD6F 3A0300              LDA      IOBYTE

         *
         * ENTRY AT PNCH1 ROTATES BITS A LITTLE MORE IN PREP FOR
         * SELDEV, USED BY LIST.
         *

CD72 1F         PNCH1    RAR
CD73 1F                  RAR
CD74 C368CD              JMP      READR1

         ******************************************************************
         *                                                                *
         * LIST: SELECT A LIST DEVICE BASED ON BITS 6&7 OF IOBYTE          *
         *                                                                *
         ******************************************************************

CD77 2198CD     LIST     LXI      H,LTBLE         ;BEGINNING OF THE LIST DEVICE ROUTINES
CD7A 3A0300     LIST1    LDA      IOBYTE
CD7D 1F                  RAR
CD7E 1F                  RAR
CD7F C372CD              JMP      PNCH1

         ******************************************************************
         *                                                                *
         * LISTST: GET THE STATUS OF THE CURRENTLY ASSIGNED LIST DEVICE    *
         *                                                                *
         ******************************************************************

CD82 21C0CD     LISTST   LXI      H,LSTBLE        ;BEGINNING OF THE LIST DEVICE STATUS
CD85 C37ACD              JMP      LIST1

         ******************************************************************
         *                                                                *
         * IF CUSTOMIZING I/O ROUTINES IS BEING PERFORMED, THE TABLE       *
         * BELOW SHOULD BE MODIFIED TO REFLECT THE CHANGES. ALL I/O        *
         * DEVICES ARE DECODED OUT OF IOBYTE AND THE JUMP IS TAKEN FROM    *
         * THE FOLLOWING TABLES.                                           *
         *                                                                *
         ******************************************************************

         *
         * CONSOLE INPUT TABLE
         *

CD88 F6CD       CITBLE   DW       CIUC1           ;INPUT FROM USER CONSOLE 1 (CURRENTLY
```

```
                                               ;              SWBD PARALLEL PORT 4)
         CD8A 0BCE              DW     CICRT     ;INPUT FROM CRT (CURRENTLY SWITCHBOARD
                                               ;              SERIAL PORT 1)
         CD8C 62CD              DW     READER    ;INPUT FROM READER (DEPENDS ON READER
                                               ;              SELECTION)
         CD8E 03E0              DW     CITTY     ;INPUT FROM TTY (CURRENTLY INPUT FROM
                                               ;              DISK JOCKEY 2D)


                *
                * CONSOLE OUTPUT TABLE
                *

         CD90 C8CD      COTBLE  DW     COCRT     ;OUTPUT TO CRT (MSDV)
                                               ;
         CD92 C8CD              DW     COCRT     ;OUTPUT TO CRT (MSDV)
                                               ;
         CD94 77CD              DW     LIST      ;OUTPUT TO LIST DEVICE (DEPENDS ON
                                               ;         BITS 6&7 OF IOBYTE)
         CD96 06E0              DW     COTTY     ;OUTPUT TO TTY (CURRENTLY OUTPUT TO
                                               ;              DISK JOCKEY 2D)

                *
                * LIST DEVICE TABLE
                *

         CD98 06E0      LTBLE   DW     COTTY     ;OUTPUT TO TTY (CURRENTLY ASSIGNED
                                               ;         BY INTIOBY,OUTPUT TO 2D)
         CD9A 3ECE              DW     COPTR     ;OUTPUT TO PRINTER
                                               ;
         CD9C CCCD              DW     COLPT     ;OUTPUT TO LINE PRINTER (CURRENTLY
                                               ;         SWITCHBOARD SERIAL PORT 1)
         CD9E D7CD              DW     COUL1     ;OUTPUT TO USER LINE PRINTER 1 (CURRENTLY
                                               ;         SWITCHBOARD SERIAL PORT 1)

                *
                * PUNCH DEVICE TABLE
                *

         CDA0 06E0      PTBLE   DW     COTTY     ;OUTPUT TO THE TTY (CURRENTLY ASSIGNED
                                               ;         BY INTIOBY,OUTPUT TO 2D)
         CDA2 3ECE              DW     COPTR     ;OUTPUT TO PRINTER
                                               ;
         CDA4 CCCD              DW     COUP1     ;OUTPUT TO USER PUNCH 1 (CURRENTLY
                                               ;         SWITCHBOARD SERIAL PORT 1)
         CDA6 CCCD              DW     COUP2     ;OUTPUT TO USER PUNCH 2 (CURRNTLLY
                                               ;         SWITCHBOARD SERIAL PORT 1)

                *
                * READER DEVICE INPUT TABLE
                *

         CDA8 03E0      RTBLE   DW     CITTY     ;INPUT FROM TTY (CURRENTLY ASSIGNED
                                               ;         BY INTIOBY, INPUT FROM 2D)
         CDAA 0BCE              DW     CIPTR     ;INPUT FROM PAPER TAPE READER (CURRENTLY
                                               ;         SWITCHBOARD SERIAL PORT 1)
         CDAC 0BCE              DW     CIUR1     ;INPUT FROM USER READER 1 (CURRENTLY
                                               ;         SWITCHBOARD SERIAL PORT 1)
```

```
CDAE  ØBCE                DW       CIUR2           ;INPUT FROM USER READER 2 (CURRENTLY
                                                  ;       SWITCHBOARD SERIAL PORT 1)


                *
                * CONSOLE STATUS TABLE
                *

CDBØ  Ø2CE       CSTBLE DW         CSUC1           ;STATUS FROM SWBD PARALLEL PORT 4, AS
                                                  ;       READ FROM ATTN BIT Ø)
CDB2  1FCE              DW         CSCRT           ;STATUS FROM CRT (CURRENTLY SWITCHBOARD
                                                  ;       SERIAL PORT 1)
CDB4  3CCD              DW         CSREADR         ;STATUS FROM READER (DEPENDS ON READER DEVICE )
                                                  ;
CDB6  17CE              DW         CSTTY           ;STATUS OF TTY (CURRENTLY STSTUS FROM
                                                  ;       DISK JOCKEY 2D)
                *
                * STATUS FROM READER DEVICE
                *

CDB8  17CE       CSRTBLE DW        CSTTY           ;STATUS FROM TTY (CURRENTLY ASSIGNED
                                                  ;       BY INTIOBY, STATUS OF 2D)
CDBA  1FCE              DW         CSPTR           ;STATUS FROM PAPER TAPE READER (CURRENTLY
                                                  ;       SWITCHBOARD SERIAL PORT 1)
CDBC  1FCE              DW         CSUR1           ;STATUS FROM USER READER 1 (CURRENTLY
                                                  ;       SWITCHBOARD SERIAL PORT 1)
CDBE  1FCE              DW         CSUR2           ;STATUS OF USER READER 2 (CURRENTLY
                                                  ;       SWITCHBOARD SERIAL PORT 1)


                *
                * STATUS FROM LIST DEVICE
                *

CDCØ  2DCE       LSTBLE DW         READY           ;CONSOLE ALWAYS READY
CDC2  2DCE              DW         READY           ;GET LIST STATUS
CDC4  28CE              DW         LSLPT
CDC6  28CE              DW         LSLPT

                *********************************************************************
                *                                                                 *
                * ROUTINES FOR MY SYSTEM.  J. J. O'BRIEN                           *
                *                                                                 *
                *********************************************************************

                *
                * MSDV VIDEO DRIVER
                *

CDC8  79         COCRT  MOV        A,C             ;MSDV WANTS DATA IN A
CDC9  C3ØØE8            JMP        MSDV            ;GO THERE


                *
                *********************************************************************
                *                                                                 *
                * THE FOLLOWING EQUATES SET OUTPUT DEVICE TO OUTPUT TO THE         *
                * SWITCHBOARD SERIAL PORT 1.                                       *
```

```
                         *                                                      *
                         ****************************************************************
CDCC =           COPTP   EQU     $                       ;OUTPUT FROM PAPER TAPE PUNCH
CDCC =           COUP1   EQU     $                       ;OUTPUT FROM USER PUNCH 1
CDCC =           COUP2   EQU     $                       ;OUTPUT FROM USER PUNCH 2
CDCC DB02        COLPT   IN      2                       ;OUTPUT FROM LINE PRINTER,GET STATUS
CDCE E680                ANI     80H                     ;WAIT UNTIL OK TO SEND
CDD0 CACCCD              JZ      COLPT
CDD3 79                  MOV     A,C                     ;OUTPUT THE CHARACTER
CDD4 D301                OUT     1
CDD6 C9                  RET

                         ****************************************************************
                         *                                                      *
                         * CUSTOM I/O PRINTER DRIVER FOR DIABLO PRINTER WITH 1200 BAUD  *
                         * ETX/ACK HANDSHAKE.                                           *
                         *                                                      *
                         ****************************************************************

CDD7 CDCCCD      COUL1   CALL    COLPT                   ;OUTPUT THE CHARACTER
CDDA 3AF5CD              LDA     COUNT
CDDD 3D                  DCR     A
CDDE 32F5CD              STA     COUNT
CDE1 C0                  RNZ
CDE2 3E4E                MVI     A,78
CDE4 32F5CD              STA     COUNT
CDE7 0E03                MVI     C,AETX
CDE9 CDCCCD              CALL    COLPT
CDEC CD0BCE      PWAIT   CALL    CIPTR
CDEF FE06                CPI     AACK
CDF1 C2ECCD              JNZ     PWAIT
CDF4 C9                  RET

CDF5 32          COUNT   DB      50

                         ****************************************************************
                         *                                                      *
                         * THE FOLLOWING EQUATES SET THE INPUT TO COME FROM THE SWBD    *
                         * PARALLEL PORT 4, WITH STATUS ON ATTENTION PORT BIT 0.        *
                         *                                                      *
                         ****************************************************************

CDF6 DB03        CIUC1   IN      3                       ;GET ATTENTION BYTE
CDF8 E601                ANI     1                       ;GET BIT 0 ONLY
CDFA CAF6CD              JZ      CIUC1                   ;WAIT FOR CHARACTER
CDFD DB04                IN      4                       ;GET CHARACTER
CDFF E67F                ANI     7FH                     ;STRIP OFF THE PARITY
CE01 C9                  RET

CE02 DB03        CSUC1   IN      3                       ;GET ATTENTION BYTE
CE04 E601                ANI     1                       ;GET BIT 0 ONLY
CE06 EE01                XRI     1                       ;CHANGE POLARITY
CE08 C31ACE              JMP     STAT                    ;RETURN PROPER INDICATION

                         ****************************************************************
```
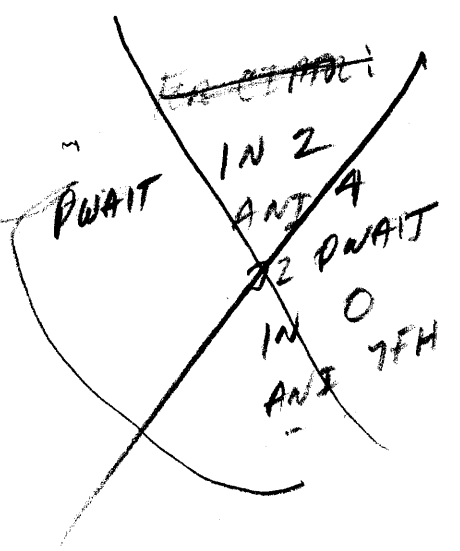
[Handwritten annotations:]

PORT 2 80H BIT = PORT 1 XMT STATUS

PORT 1 TO Diablo

PWAIT IN 2 / ANI 4 / JZ PWAIT / IN 0 / ANI 7FH

← 7 NOP'S NG Here!

```
                    *                                                   *
                    * THE FOLLOWING EQUATES SET THE INPUT FROM THE DEVICES TO COME  *
                    * FROM THE SWITCHBOARD SERIAL PORT 1.                *
                    *                                                   *
                    *****************************************************************

CE0B =              CICRT   EQU     $               ;INPUT FROM CRT
CE0B =              CIUR1   EQU     $               ;INPUT FROM USER READER 1
CE0B =              CIUR2   EQU     $               ;INPUT FROM USER READER 2
CE0B DB02           CIPTR   IN      2               ;INPUT FROM PAPER TAPE READER, GET STATUS
CE0D E640                   ANI     40H             ;WAIT FOR CHARACTER
CE0F CA0BCE                 JZ      CIPTR
CE12 DB01                   IN      1
CE14 E67F                   ANI     7FH             ;STRIP OFF THE PARITY
CE16 C9                     RET

                    *****************************************************************
                    *                                                   *
                    * CONSOLE STATUS ROUTINES, TEST IF A CHARACTER HAS ARRIVED.   *
                    *                                                   *
                    *****************************************************************

CE17 CD21E0         CSTTY   CALL    DJTSTAT         ;STATUS FROM DISK JOCKEY 2D
CE1A 3E00           STAT    MVI     A,0             ;PREP FOR ZERO RETURN
CE1C C0                     RNZ                     ;NOTHING FOUND
CE1D 3D                     DCR     A               ;RETURN WITH 0FFH
CE1E C9                     RET

                    *****************************************************************
                    *                                                   *
                    * THE FOLLOWING EQUATES CAUSE THE DEVICES TO GET STATUS FROM   *
                    * THE SWITCHBOARD SERIAL PORT 1.                     *
                    *                                                   *
                    *****************************************************************

CE1F =              CSUR1   EQU     $               ;STATUS OF USER READER 1
CE1F =              CSUR2   EQU     $               ;STATUS OF USER READER 2
CE1F =              CSPTR   EQU     $               ;STATUS OF PAPER TAPE READER
CE1F DB02           CSCRT   IN      2               ;STATUS FROM CRT, GET STATUS
CE21 E640                   ANI     40H             ;STRIP OF DATA READY BIT
CE23 EE40                   XRI     40H             ;MAKE CORRECT POLARITY
CE25 C31ACE                 JMP     STAT            ;RETURN PROPER INDICATION

                    *****************************************************************
                    *                                                   *
                    * LIST DEVICE STATUS ROUTINES.                       *
                    *                                                   *
                    *****************************************************************

CE28 DB02           LSLPT   IN      2               ;ALL OTHER DEVICES WAIT
CE2A E680                   ANI     80H
CE2C C8                     RZ
CE2D 3EFF           READY   MVI     A,0FFH
CE2F C9                     RET

                    *****************************************************************
```

Handwritten annotations:

KYBD INPUT PORT 1

7OP's NG here!

7 NG here, TRY 1-NOP OK
4 NOP'S OK
OK here!   6 NOP'S OK!
7 NG

CE25 → WITH 6 NOP's

OK here! PORT 2 40H BIT = INPUT, SWBD BASE +1 (PORT 1)

7NOP's OK here!

REC STATUS

← 7 NOP's OK here

```
                *                                                    *
                * THIS INITIALLIZING ROUTINE SAMPLES BIT Ø OF SWBD PORT 7 TO   *
                * DETERMINE IF THE KEYBOARD IS PLUGGED IN.  IF THE KEYBOARD IS  *
                * PLUGGED IN, THE LSB RETURNS A Ø. OTHERWISE, IT IS A 1.        *
                * THIS 1 IS ADDED TO IOBYTE TO CHANGE THE CONSOLE INPUT FROM    *
                * THE SWBD PARALLEL PORT 4 (THE KEYBOARD) TO THE SWBD SERIAL    *
                * PORT THAT RECEIVES RS232 DATA FROM THE RS232 TERMINAL.        *
                *                                                    *
                ****************************************************************

CE30 ØE19      TINIT   MVI     C,CLEAR         ;INITIALIZE THE TERMINAL ROUTINE
CE32 DBØ7              IN      7               ;GET KEYBOARD INTERLOCK BYTE
CE34 E6Ø1             ANI     1               ;GET BIT 1 ONLY
CE36 C6CØ             ADI     INTIOBY         ;ADD INTIOBY TO KEYBOARD BIT
CE38 32Ø3ØØ           STA     IOBYTE          ;INITIALIZE IOBYTE
CE3B C3ØCCD           JMP     COUT


                ****************************************************************
                *                                                    *
                * ROUTINE FOR OKIDATA PRINTER                         *
                * PRINTER IS ON PORT Ø WITH PRINTER READY ON PORT 5 BIT 1       *
                *                                                    *
                ****************************************************************

CE3E DBØ2      COPTR   IN      2               ;INPUT FROM PORT 2
CE40 E6Ø8              ANI     8               ;WAIT UNTIL OK TO SEND
CE42 CA3ECE           JZ      COPTR
CE45 DBØ5      COPTR1  IN      5               ;BUFFER FULL?
CE47 E6Ø1             ANI     1
CE49 CA45CE           JZ      COPTR1          ;WAIT UNTIL PRINTER READY
CE4C 79              MOV     A,C             ;OUTPUT THE CHARACTER
CE4D D3ØØ            OUT     Ø
CE4F C9             RET


                ****************************************************************
                *                                                    *
                * GOCPM IS THE ENTRY POINT FROM COLD BOOTS, AND WARM BOOTS. IT  *
                * INITIALIZES SOME OF THE LOCATIONS IN PAGE Ø, AND SETS UP THE  *
                * INITIAL DMA ADDRESS (80H).                          *
                *                                                    *
                ****************************************************************
```

CE3E

```
CE50 218ØØØ    GOCPM   LXI     H,BUFF          ;SET UP INITIAL DMA ADDRESS
CE53 CD43CF           CALL    SETDMA
CE56 3EC3            MVI     A,(JMP)         ;INITIALIZE JUMP TO WARM BOOT
CE58 32ØØØØ          STA     WBOT            ;INITIALIZE JUMP TO BDOS
CE5B 32Ø5ØØ          STA     ENTRY           ;INITIALIZE JUMP TO BDOS
CE5E 21Ø3CD          LXI     H,WBOOTE        ;ADDRESS IN WARM BOOT JUMP
CE61 22Ø1ØØ          SHLD    WBOT+1
CE64 21Ø6BF          LXI     H,BDOS+6        ;ADDRESS IN BDOS JUMP
CE67 22Ø6ØØ          SHLD    ENTRY+1
CE6A AF             XRA     A               ;A <- Ø
CE6B 32FDD4          STA     BUFSEC          ;DISK JOCKEY BUFFER EMPTY
CE6E 32Ø5D1          STA     BUFWRTN         ;SET BUFFER NOT DIRTY FLAG
CE71 3AØ4ØØ          LDA     CDISK           ;JUMP TO CP/M WITH CURRENTLY SELECTED DISK IN C
CE74 4F             MOV     C,A
```

```
CE75 3AA2CE            LDA     CWFLG
CE78 A7                ANA     A
CE79 11A4CE            LXI     D,COLDBEG          ;BEGINNING OF INITIAL COMMAND
CE7C 3E0F              MVI     A,COLDEND-COLDBEG+1 ;LENGTH OF COMMAND
CE7E CA86CE            JZ      CLDCMND
CE81 11B3CE            LXI     D,WARMBEG
CE84 3E01              MVI     A,WARMEND-WARMBEG+1
CE86 2108B7   CLDCMND LXI     H,CCP+8            ;COMMAND BUFFER
CE89 3207B7            STA     CCP+7
CE8C 47                MOV     B,A
CE8D CDCCD1            CALL    MOVLOP
CE90 3AA2CE            LDA     CWFLG
CE93 A7                ANA     A
CE94 3AA3CE            LDA     AUTOFLG
CE97 CA9BCE            JZ      CLDBOT
CE9A 1F                RAR
CE9B 1F       CLDBOT  RAR
CE9C DA00B7            JC      CCP
CE9F C303B7            JMP     CCP+3              ;ENTER CP/M

CEA2 00       CWFLG   DB      0                  ;COLD/WARM BOOT FLAG

              ******************************************************************
              *                                                                *
              * THE FOLLOWING BYTE DETERMINES IF AN INITIAL COMMAND IS TO BE   *
              * GIVEN TO CP/M ON WARM OR COLD BOOTS. THE VALUE OF THE BYTE IS  *
              * USED TO GIVE THE COMMAND TO CP/M:                              *
              *                                                                *
              * 0 = NEVER GIVE COMMAND.                                        *
              * 1 = GIVE COMMAND ON COLD BOOTS ONLY.                           *
              * 2 = GIVE THE COMMAND ON WARM BOOTS ONLY.                       *
              * 3 = GIVE THE COMMAND ON WARM AND COLD BOOTS.                   *
              *                                                                *
              ******************************************************************

CEA3 01       AUTOFLG DB      1                  ;AUTO COMMAND FEATURE

              ******************************************************************
              *                                                                *
              * IF THERE IS A COMMAND INSERTED HERE, IT WILL BE GIVEN IF THE   *
              * AUTO FEATURE IS ENABLED.                                       *
              *       FOR EXAMPLE:                                             *
              *                                                                *
              *       COLDBEG DB      'MBASIC MYPROG'                          *
              *       COLDEND DB      0                                        *
              *                                                                *
              * WILL EXECUTE MICROSOFT BASIC, AND MBASIC WILL EXECUTE THE      *
              * "MYPROG" BASIC PROGRAM.                                        *
              *                                                                *
              ******************************************************************

CEA4 5355424D49COLDBEG DB      'SUBMIT STARTUP';COLD BOOT COMMAND
CEB2 00       COLDEND DB      0
              WARMBEG DB      ' '                ;WARM BOOT COMMAND GOES HERE
CEB3 00       WARMEND DB      0
```

```
                    *****************************************************************
                    *                                                               *
                    * WBOOT LOADS IN ALL OF CP/M EXCEPT THE CBIOS, THEN INITIALIZES *
                    * SYSTEM PARAMETERS AS IN COLD BOOT. SEE THE COLD BOOT LOADER   *
                    * LISTING FOR EXACTLY WHAT HAPPENS DURING WARM AND COLD BOOTS.  *
                    *                                                               *
                    *****************************************************************

CEB4 310001   WBOOT    LXI    SP,TPA              ;SET UP STACK POINTER
CEB7 3E01              MVI    A,1
CEB8 =        WFLG     EQU    $-1                 ;TEST IF BEGINNING OR
CEB9 A7                ANA    A                   ;          ENDING A WARM BOOT
CEBA 3E01              MVI    A,1
CEBC 32B8CE            STA    WFLG
CEBF 32A2CE            STA    CWFLG               ;SET COLD/WARM BOOT FLAG
CEC2 CA50CE            JZ     GOCPM
CEC5 AF                XRA    A
CEC6 32B8CE            STA    WFLG
CEC9 4F                MOV    C,A

                       IF     (MAXHD NE 0) AND FIRST  ;SUPPLY WARM BOOT FROM HARD DISK ?
                       LXI    H,CCP-200H          ;INITIAL DMA ADDRESS
                       PUSH   H
                       STA    HEAD
                       MVI    A,4
                       PUSH   PSW                 ;SAVE FIRST SECTOR
                       CALL   HDDRV               ;SELECT DRIVE A
                       MVI    C,0
                       CALL   HDTRK               ;HOME THE DRIVE
              WARMLOD  POP    PSW                 ;RESTORE SECTOR
                       POP    H                   ;RESTORE DMA ADDRESS
                       INR    A
                       STA    HDSECTR
                       CPI    16                  ;PAST BDOS ?
                       JZ     WBOOT               ;YES, ALL DONE
                       INR    H                   ;UPDATE DMA ADDRESS
                       INR    H
                       SHLD   HDADD
                       PUSH   H
                       PUSH   PSW
              WARMRD   LXI    B,RETRIES*100H+0  ;RETRY COUNTER
              WRMREAD  PUSH   B                   ;SAVE THE RETRY COUNT
                       CALL   HDREAD              ;READ THE SECTOR
                       POP    B
                       JNC    WARMLOD             ;TEST FOR ERROR
                       DCR    B                   ;UPDATE THE ERROR COUNT
                       JNZ    WRMREAD             ;KEEP TRYING IF NOT TO MANY ERRORS
                       HLT                        ;CAN'T WARM BOOT
                       ENDIF

                       IF     (MAXFLOP NE 0) AND NOT FIRST ;SUPPLY WARM BOOT FROM 2D ?
CECA CD33CD            CALL   DJDRV               ;SELECT DRIVE A
CECD 0E00              MVI    C,0                 ;SELECT SINGLE DENSITY
CECF CD2DE4            CALL   DJDEN
CED2 0E00              MVI    C,0                 ;SELECT SIDE 0
CED4 CD30E4            CALL   DJSIDE
```

```
CED7 3E0F              MVI     A,15                ;INITIALIZE THE SECTOR TO READ
CED9 32F7CE            STA     NEWSEC
CEDC 2100B6            LXI     H,CCP-100H          ;AND THE DMA ADDRESS
CEDF 2216CF            SHLD    NEWDMA
CEE2 CDF6CE            CALL    WARMLOD             ;READ IN CP/M
CEE5 0100BC            LXI     B,CCP+500H          ;LOAD ADDRESS FOR REST OF WARM BOOT
CEE8 CD12E4            CALL    DJDMA
CEEB 0E08             MVI     C,8
CEED CD0FE4            CALL    DJSEC
CEF0 CD2ACF            CALL    WARMRD
CEF3 C303BC            JMP     CCP+503H


CEF6 3E0F     WARMLOD  MVI     A,15                ;PREVIOUS SECTOR
CEF7 =        NEWSEC   EQU     $-1
CEF8 3C                INR     A                   ;UPDATE THE PREVIOUS SECTOR
CEF9 3C                INR     A
CEFA FE1B              CPI     27                  ;WAS IT THE LAST ?
CEFC DA0ECF            JC      NOWRAP
CEFF D609              SUI     9                   ;YES
CF01 FE13              CPI     19
CF03 C8               RZ
CF04 2A16CF            LHLD    NEWDMA
CF07 1180FB            LXI     D,-480H
CF0A 19                DAD     D
CF0B 2216CF            SHLD    NEWDMA
CF0E 32F7CE   NOWRAP   STA     NEWSEC              ;SAVE THE NEW SECTOR TO READ
CF11 4F                MOV     C,A
CF12 CD0FE4            CALL    DJSEC
CF15 2100B6            LXI     H,CCP-100H          ;GET THE PREVIOUS DMA ADDRESS
CF16 =        NEWDMA   EQU     $-2
CF18 110001            LXI     D,100H              ;UPDATE THE DMA ADDRESS
CF1B 19                DAD     D
CF1C 2216CF            SHLD    NEWDMA              ;SAVE THE DMA ADDRESS
CF1F 44                MOV     B,H
CF20 4D                MOV     C,L
CF21 CD12E4            CALL    DJDMA               ;SET THE DMA ADDRESS
CF24 CD2ACF            CALL    WARMRD
CF27 C3F6CE            JMP     WARMLOD


CF2A 01000A   WARMRD   LXI     B,RETRIES*100H+0;MAXIMUM # OF ERRORS
CF2D C5       WRMREAD  PUSH    B
CF2E CD0CE4            CALL    DJTRK               ;SET THE TRACK
CF31 CD15E4            CALL    DJREAD              ;READ THE SECTOR
CF34 C1                POP     B
CF35 D0                RNC                         ;CONTINUE IF SUCCESSFUL
CF36 05                DCR     B
CF37 C22DCF            JNZ     WRMREAD             ;KEEP TRYING
CF3A C32AE4            JMP     DJERR
                      ENDIF
```

```
        *********************************************************************
        *                                                                   *
        * SETSEC JUST SAVES THE DESIRED SECTOR TO SEEK TO UNTIL AN           *
        * ACTUAL READ OR WRITE IS ATTEMPTED.                                *
        *                                                                   *
        *********************************************************************
```

```
CF3D 60        SETSEC MOV     H,B
CF3E 69               MOV     L,C
CF3F 22F5D4           SHLD    CPMSEC
CF42 C9        DONOP  RET

               **********************************************************************
               *                                                                    *
               * SETDMA SAVES THE DMA ADDRESS FOR THE DATA TRANSFER.                 *
               *                                                                    *
               **********************************************************************

CF43 60        SETDMA MOV     H,B              ;HL <- BC
CF44 69               MOV     L,C
CF45 22E5D0           SHLD    CPMDMA           ;CP/M DMA ADDRESS
CF48 C9               RET

               **********************************************************************
               *                                                                    *
               * HOME IS TRANSLATED INTO A SEEK TO TRACK ZERO.                       *
               *                                                                    *
               **********************************************************************

CF49 0E00      HOME   MVI     C,0              ;TRACK TO SEEK TO

               **********************************************************************
               *                                                                    *
               * SETTRK SAVES THE TRACK # TO SEEK TO. NOTHING IS DONE AT THIS        *
               * POINT, EVERYTHING IS DEFFERED UNTIL A READ OR WRITE.                *
               *                                                                    *
               **********************************************************************

CF4B 79        SETTRK MOV     A,C              ;A <- TRACK #
CF4C 32F8D4           STA     CPMTRK           ;CP/M TRACK #
CF4F C9               RET

               **********************************************************************
               *                                                                    *
               * SECTRAN TRANSLATES A LOGICAL SECTOR # INTO A PHYSICAL SECTOR        *
               * #.                                                                  *
               *                                                                    *
               **********************************************************************

                      IF      (MAXHD NE 0) AND (MAXFLOP NE 0) ;BOTH TYPES ?
CF50 3AF7D4    SECTRAN LDA     CPMDRV                  ;GET THE DRIVE NUMBER

                      IF      FIRST
                      CPI     MAXHD*LOGDSK     ;OVER THE # OF HARD DISKS ?
                      JC      TRANHD
                      ELSE
CF53 FE02             CPI     MAXFLOP          ;OVER THE # OF FLOPPIES ?
CF55 D287CF           JNC     TRANHD
                      ENDIF
                      ENDIF

                      IF      (MAXHD EQ 0) OR (MAXFLOP EQ 0) ;JUST ONE TYPE ?
```

```
                    SECTRAN EQU     $
                            ENDIF

                            IF      MAXFLOP NE Ø      ;FLOPPY TRANSLATION
     CF58 Ø3        TRANFP  INX     B
     CF59 D5                PUSH    D                 ;SAVE TABLE ADDRESS
     CF5A C5                PUSH    B                 ;SAVE SECTOR #
     CF5B CD69DØ            CALL    GETDPB            ;GET DPB ADDRESS INTO HL
     CF5E 7E                MOV     A,M               ;GET # OF CP/M SECTORS/TRACK
     CF5F B7                ORA     A                 ;CLEAR CARY
     CF6Ø 1F                RAR                       ;DIVIDE BY TWO
     CF61 91                SUB     C
     CF62 F5                PUSH    PSW               ;SAVE ADJUSTED SECTOR
     CF63 FA6FCF            JM      SIDETWO
     CF66 F1        SIDEA   POP     PSW               ;DISCARD ADJUSTED SECTOR
     CF67 C1                POP     B                 ;RESTORE SECTOR REQUESTED
     CF68 D1                POP     D                 ;RESTOR ADDRESS OF XLT TABLE
     CF69 EB        SIDEONE XCHG                      ;HL <- &(TRANSLATION TABLE)
     CF6A Ø9                DAD     B                 ;BC = OFFSET INTO TABLE
     CF6B 6E                MOV     L,M               ;HL <- PHYSICAL SECTOR
     CF6C 2600              MVI     H,Ø
     CF6E C9                RET

     CF6F Ø1ØFØØ    SIDETWO LXI     B,15              ;OFFSET TO SIDE BIT
     CF72 Ø9                DAD     B
     CF73 7E                MOV     A,M
     CF74 E6Ø8              ANI     8                 ;TEST FOR DOUBLE SIDED
     CF76 CA66CF            JZ      SIDEA             ;MEDIA IS ONLY SINGLE SIDED
     CF79 F1                POP     PSW               ;RETRIEVE ADJUSTED SECTOR
     CF7A C1                POP     B
     CF7B 2F                CMA                       ;MAKE SECTOR REQUEST POSITIVE
     CF7C 3C                INR     A
     CF7D 4F                MOV     C,A               ;MAKE NEW SECTOR THE REQUESTED SECTOR
     CF7E D1                POP     D
     CF7F CD69CF            CALL    SIDEONE
     CF82 3E8Ø              MVI     A,8ØH             ;SIDE TWO BIT
     CF84 B4                ORA     H                 ;        AND SECTOR
     CF85 67                MOV     H,A
     CF86 C9                RET
                            ENDIF

                            IF      MAXHD NE Ø        ;HARD DISK TRANSLATION ROUTINE
     CF87 6Ø        TRANHD  MOV     H,B
     CF88 69                MOV     L,C
     CF89 23                INX     H
     CF8A C9                RET
                            ENDIF

             **********************************************************************
             *                                                                    *
             * SETDRV SELECTS THE NEXT DRIVE TO BE USED IN READ/WRITE             *
             * OPERATIONS. IF THE DRIVE HAS NEVER BEEN SELECTED BEFORE, A         *
             * PARAMETER TABLE IS CREATED WHICH CORRECTLY DESCRIBES THE           *
             * DISKETTE CURRENTLY IN THE DRIVE. DISKETTES CAN BE OF FOUR          *
             * DIFFERENT SECTOR SIZES:                                            *
```

```
                    *      1) 128 BYTES SINGLE DENSITY.                        *
                    *      2) 256 BYTES DOUBLE DENSITY.                        *
                    *      3) 512 BYTES DOUBLE DENSITY.                        *
                    *      4) 1024 BYTES DOUBLE DENSITY.                       *
                    *                                                         *
                    ************************************************************

CF8B 79        SETDRV  MOV     A,C                ;SAVE THE DRIVE #
CF8C 32F7D4            STA     CPMDRV
CF8F FE05             CPI     MAXFLOP+(MAXHD*LOGDSK) ;CHECK FOR A VALID DRIVE #
CF91 D25AD0           JNC     ZRET               ;ILLEGAL DRIVE #
CF94 7B               MOV     A,E                ;TEST IF DRIVE EVER LOGGED IN BEFORE
CF95 E601             ANI     1
CF97 C241D0           JNZ     SETDRV1            ;BIT 0 OF E = 0 -> NEVER SELECTED BEFORE

                      IF      (MAXHD NE 0) AND (MAXFLOP NE 0) ;BOTH TYPES ?
CF9A 3AF7D4           LDA     CPMDRV             ;GET THE DRIVE NUMBER

                      IF      FIRST
                      CPI     MAXHD*LOGDSK       ;OVER THE # OF HARD DISKS ?
                      JC      DRVHD
                      SUI     MAXHD*LOGDSK
                      ELSE
CF9D FE02             CPI     MAXFLOP            ;OVER THE # OF FLOPPIES ?
CF9F D2F7CF           JNC     SUBFP
                      ENDIF
                      ENDIF

                      IF      (MAXFLOP NE 0) AND FIRST
                      MOV     C,A                ;SAVE DRIVE #
                      MVI     A,0                ;HAVE THE FLOPPIES BEEN ACCESSED YET ?
                FLOPFLG EQU    $-1
                      ANA     A
                      JNZ     FLOPOK
                      MVI     B,17               ;FLOPPIES HAVN'T BEEN ACCESSED
                      LXI     H,DJBOOT           ;CHECK IF 2D CONTROLLER IS INSTALLED
                      MVI     A,(JMP)
                CLOPP MOV     CMP     M
                      JNZ     ZRET
                      DCR     B
                      JNZ     CLOPP
                      CALL    DJBOOT             ;INITIALIZE THE CONTROLLER
                      MVI     A,1                ;SAVE 2D INITIALIZED FLAG
                      STA     FLOPFLG
                      ENDIF
                      IF      MAXFLOP NE 0
CFA2 210100     FLOPOK LXI     H,1                ;SELECT SECTOR 1 OF TRACK 1
CFA5 22F9D4           SHLD    TRUESEC
CFA8 3E01             MVI     A,1
CFAA 32F8D4           STA     CPMTRK
CFAD CD96D1           CALL    FILL               ;FLUSH BUFFER AND REFILL
CFB0 DA5AD0           JC      ZRET               ;TEST FOR ERROR RETURN
CFB3 CD27E4           CALL    DJSTAT             ;GET STATUS ON CURRENT DRIVE
CFB6 E60C             ANI     0CH                ;STRIP OFF UNWANTED BITS
CFB8 F5              PUSH    PSW                ;USED TO SELECT A DPB
CFB9 1F              RAR
```

```
CFBA 2182D0              LXI     H,XLTS              ;TABLE OF XLT ADDRESSES
CFBD 5F                  MOV     E,A
CFBE 1600                MVI     D,0
CFC0 19                  DAD     D
CFC1 E5                  PUSH    H                   ;SAVE POINTER TO PROPER XLT
CFC2 CD69D0              CALL    GETDPB              ;GET DPH POINTER INTO DE
CFC5 EB                  XCHG                        ;
CFC6 D1                  POP     D
CFC7 0602                MVI     B,2                 ;NUMBER OF BYTES TO MOVE
CFC9 CDCCD1              CALL    MOVLOP              ;MOVE THE ADDRESS OF XLT
CFCC 110800              LXI     D,8                 ;OFFSET TO DPB POINTER
CFCF 19                  DAD     D                   ;HL <- &DPH.DPB
CFD0 E5                  PUSH    H
CFD1 2A07E0              LHLD    ORIGIN+7            ;GET ADDRESS OF DJ TERMINAL OUT ROUTINE
CFD4 23                  INX     H                   ;BUMP TO LOOK AT ADDRESS OF
                                                     ;       UART STATUS LOCATION
CFD5 7E                  MOV     A,M
CFD6 EE03                XRI     3                   ;ADJUST FOR PROPER REV DJ
CFD8 6F                  MOV     L,A
CFD9 26E3                MVI     H,(ORIGIN+300H)/100H
CFDB 7E                  MOV     A,M
CFDC E608                ANI     DBLSID              ;CHECK DOUBLE SIDED BIT
CFDE 11F5D3              LXI     D,DPB128S           ;BASE FOR SINGLE SIDED DPB'S
CFE1 C2E7CF              JNZ     SIDEOK
CFE4 1135D4              LXI     D,DPB128D           ;BASE OF DOUBLE SIDED DPB'S
CFE7 EB       SIDEOK    XCHG                        ;HL <- DBP BASE, DE <- &DPH.DPB
CFE8 D1                  POP     D                   ;RESTORE DE (POINTER INTO DPH)
CFE9 F1                  POP     PSW                 ;OFFSET TO CORRECT DPB
CFEA 17                  RAL
CFEB 17                  RAL
CFEC 4F                  MOV     C,A
CFED 0600                MVI     B,0
CFEF 09                  DAD     B
CFF0 EB                  XCHG                        ;PUT DPB ADDRESS IN DPH
CFF1 73                  MOV     M,E
CFF2 23                  INX     H
CFF3 72                  MOV     M,D
                         ENDIF

                         IF      (MAXHD NE 0) AND (MAXFLOP NE 0)
CFF4 C341D0              JMP     SETDRV1             ;SKIP OVER THE HARD DISK SELECT
                         IF      NOT FIRST
CFF7 D602     SUBFP     SUI     MAXFLOP             ;ADJUST THE DRIVE #
                         ENDIF
                         ENDIF

                         IF      MAXHD NE 0
CFF9 CD60D0   DRVHD     CALL    DIVLOG             ;DIVIDE BY LOGICAL DISKS PER DRIVE
CFFC 79                  MOV     A,C
CFFD 3222D3              STA     HDDISK
D000 CD10D3              CALL    DRVPTR
D003 7E                  MOV     A,M
D004 3C                  INR     A
D005 C241D0              JNZ     SETDRV1
D008 F6FC                ORI     NULL               ;SELECT DRIVE
D00A D352                OUT     HDFUNC
```

```
     D00C 3E05            MVI     A,SCENBL         ;ENABLE THE CONTROLLER
     D00E D350            OUT     HDCNTL
     D010 0EEF            MVI     C,239            ;WAIT APPROX 2 MINUTES FOR DISK TO READY
     D012 210000          LXI     H,0
     D015 2B      TDELAY  DCX     H
     D016 7C              MOV     A,H
     D017 B5              ORA     L
     D018 CC5ED0          CZ      DCRC
     D01B C8              RZ
     D01C DB50            IN      HDSTAT           ;TEST IF READY YET
     D01E E620            ANI     DRVRDY
     D020 C215D0          JNZ     TDELAY

                          IF      SDELAY
     D023 210000          LXI     H,0              ;TIME ONE REVOLUTION OF THE DRIVE
     D026 0E40            MVI     C,INDEX
     D028 DB50            IN      HDSTAT
     D02A A1              ANA     C
     D02B 47              MOV     B,A              ;SAVE CURRENT INDEX LEVEL IN B
     D02C DB50    INDX1   IN      HDSTAT
     D02E A1              ANA     C
     D02F B8              CMP     B                ;LOOP UTIL INDEX LEVEL CHANGES
     D030 CA2CD0          JZ      INDX1
     D033 23      INDX2   INX     H
     D034 DB50            IN      HDSTAT           ;START COUNTING UNTIL INDEX RETURNS TO
     D036 A1              ANA     C                ;        PREVIOUS STATE
     D037 B8              CMP     B
     D038 C233D0          JNZ     INDX2
                          IF      M10
                          DAD     H
                          ENDIF
     D03B 2208D2          SHLD    SETTLE           ;SAVE THE COUNT FOR TIMEOUT DELAY
                          ENDIF
     D03E CDF2D1          CALL    HDHOME
                          ENDIF

     D041 CD69D0  SETDRV1 CALL    GETDPB           ;GET ADDRESS OF DPB IN HL
     D044 010F00          LXI     B,15             ;OFFSET TO SECTOR SIZE
     D047 09              DAD     B
     D048 7E              MOV     A,M              ;GET SECTOR SIZE
     D049 E607            ANI     7H
     D04B 3296D0          STA     SECSIZ
     D04E 7E              MOV     A,M
     D04F 1F              RAR
     D050 1F              RAR
     D051 1F              RAR
     D052 1F              RAR
     D053 E60F            ANI     0FH
     D055 32D4D0          STA     SECPSEC
     D058 EB              XCHG                     ;HL <- DPH
     D059 C9              RET

     D05A 210000  ZRET    LXI     H,0              ;SELDRV ERROR EXIT
     D05D C9              RET

                          IF      MAXHD NE 0
```

```
D05E ØD        DCRC     DCR     C                    ;CONDITIONAL DECREMENT C ROUTINE
D05F C9                 RET

D060 ØEØØ       DIVLOG   MVI     C,Ø
D062 D6Ø3       DIVLOGX  SUI     LOGDSK
D064 D8                  RC
D065 ØC                  INR     C
D066 C362DØ              JMP     DIVLOGX
                         ENDIF


        ***********************************************************************
        *                                                                     *
        * GETDPB RETURNS HL POINTING TO THE DPB OF THE CURRENTLY              *
        * SELECTED DRIVE, DE POINTING TO DPH.                                  *
        *                                                                     *
        ***********************************************************************

D069 3AF7D4     GETDPB   LDA     CPMDRV
D06C 6F                  MOV     L,A                  ;FORM OFFSET
D06D 2600                MVI     H,Ø
D06F 29                  DAD     H
D070 29                  DAD     H
D071 29                  DAD     H
D072 29                  DAD     H
D073 11A5D4              LXI     D,DPBASE             ;BASE OF DPH'S
D076 19                  DAD     D
D077 E5                  PUSH    H                    ;SAVE ADDRESS OF DPH
D078 110A00              LXI     D,1Ø                 ;OFFSET TO DPB
D07B 19                  DAD     D
D07C 7E                  MOV     A,M                  ;GET LOW BYTE OF DPB ADDRESS
D07D 23                  INX     H
D07E 66                  MOV     H,M                  ;GET LOW BYTE OF DPB
D07F 6F                  MOV     L,A
D080 D1                  POP     D
D081 C9                  RET


        ***********************************************************************
        *                                                                     *
        * XLTS IS A TABLE OF ADDRESS THAT POINT TO EACH OF THE XLT            *
        * TABLES FOR EACH SECTOR SIZE.                                         *
        *                                                                     *
        ***********************************************************************

                         IF      MAXFLOP NE Ø
D082 27D3       XLTS     DW      XLT128               ;XLT FOR 128 BYTE SECTORS
D084 42D3                DW      XLT256               ;XLT FOR 256 BYTE SECTORS
D086 77D3                DW      XLT512               ;XLT FOR 512 BYTE SECTORS
D088 B4D3                DW      XLT124               ;XLT FOR 1Ø24 BYTE SECTORS
                         ENDIF


        ***********************************************************************
        *                                                                     *
        * WRITE ROUTINE MOVES DATA FROM MEMORY INTO THE BUFFER. IF THE        *
        * DESIRED CP/M SECTOR IS NOT CONTAINED IN THE DISK BUFFER, THE        *
        * BUFFER IS FIRST FLUSHED TO THE DISK IF IT HAS EVER BEEN             *
        * WRITTEN INTO, THEN A READ IS PERFORMED INTO THE BUFFER TO GET       *
```

```
                     * THE DESIRED SECTOR. ONCE THE CORRECT SECTOR IS IN MEMORY, THE *
                     * BUFFER WRITTEN INDICATOR IS SET, SO THE BUFFER WILL BE         *
                     * FLUSHED, THEN THE DATA IS TRANSFERRED INTO THE BUFFER.         *
                     *                                                               *
                     ****************************************************************

DØ8A 79      WRITE   MOV    A,C               ;SAVE WRITE COMMAND TYPE
DØ8B 32FCDØ          STA    WRITTYP
DØ8E 3EØ1            MVI    A,1               ;SET WRITE COMMAND
DØ9Ø Ø6              DB     (MVI) OR (B*8)    ;THIS "MVI B" INSTRUCTION  CAUSES
                                             ;          THE FOLLOWING "XRA A" TO
                                             ;          BE SKIPPED OVER.


                     ****************************************************************
                     *                                                               *
                     * READ ROUTINE TO BUFFER DATA FROM THE DISK. IF THE SECTOR      *
                     * REQUESTED FROM CP/M IS IN THE BUFFER, THEN THE DATA IS SIMPLY *
                     * TRANSFERRED FROM THE BUFFER TO THE DESIRED DMA ADDRESS. IF    *
                     * THE BUFFER DOES NOT CONTAIN THE DESIRED SECTOR, THE BUFFER IS *
                     * FLUSHED TO THE DISK IF IT HAS EVER BEEN WRITTEN INTO, THEN    *
                     * FILLED WITH THE SECTOR FROM THE DISK THAT CONTAINS THE        *
                     * DESIRED CP/M SECTOR.                                          *
                     *                                                               *
                     ****************************************************************

DØ91 AF      READ    XRA    A                 ;SET THE COMMAND TYPE TO READ
DØ92 32E8DØ          STA    RDWR              ;SAVE COMMAND TYPE


                     ****************************************************************
                     *                                                               *
                     * REDWRT CALCULATES THE PHYSICAL SECTOR ON THE DISK THAT        *
                     * CONTAINS THE DESIRED CP/M SECTOR, THEN CHECKS IF IT IS THE    *
                     * SECTOR CURRENTLY IN THE BUFFER. IF NO MATCH IS MADE, THE      *
                     * BUFFER IS FLUSHED IF NECESSARY AND THE CORRECT SECTOR READ    *
                     * FROM THE DISK.                                                *
                     *                                                               *
                     ****************************************************************

DØ95 Ø6ØØ    REDWRT  MVI    B,Ø               ;THE Ø IS MODIFIED TO CONTAIN THE LOG2
DØ96 =       SECSIZ  EQU    $-1               ;        OF THE PHYSICAL SECTOR SIZE/128
                                             ;        ON THE CURRENTLY SELECTED DISK.
DØ97 2AF5D4          LHLD   CPMSEC            ;GET THE DESIRED CP/M SECTOR #
DØ9A 7C              MOV    A,H
DØ9B E68Ø            ANI    8ØH               ;SAVE ONLY THE SIDE BIT
DØ9D 4F              MOV    C,A               ;REMEMBER THE SIDE
DØ9E 7C              MOV    A,H
DØ9F E67F            ANI    7FH               ;FORGET THE SIDE BIT
DØA1 67              MOV    H,A
DØA2 2B              DCX    H                 ;TEMPORARY ADJUSTMENT
DØA3 Ø5      DIVLOOP DCR    B                 ;UPDATE REPEAT COUNT
DØA4 CAB1DØ          JZ     DIVDONE
DØA7 B7              ORA    A
DØA8 7C              MOV    A,H
DØA9 1F              RAR
DØAA 67              MOV    H,A
DØAB 7D              MOV    A,L
```

```
       DØAC 1F                  RAR                          ;DIVIDE THE CP/M SECTOR # BY THE SIZE
                                                             ;         OF THE PHYSICAL SECTORS
       DØAD 6F                  MOV      L,A
       DØAE C3A3DØ              JMP      DIVLOOP             ;
       DØB1 23       DIVDONE    INX      H
       DØB2 7C                  MOV      A,H
       DØB3 B1                  ORA      C                   ;RESTORE THE SIDE BIT
       DØB4 67                  MOV      H,A
       DØB5 22F9D4              SHLD     TRUESEC             ;SAVE THE PHYSICAL SECTOR NUMBER
       DØB8 21F7D4              LXI      H,CPMDRV            ;POINTER TO DESIRED DRIVE,TRACK, AND SECTOR
       DØBB 11FBD4              LXI      D,BUFDRV            ;POINTER TO BUFFER DRIVE,TRACK, AND SECTOR
       DØBE 0605                MVI      B,5                 ;COUNT LOOP
       DØCØ 05       DTSLOP     DCR      B                   ;TEST IF DONE WITH COMPARE
       DØC1 CACFDØ              JZ       MOVE                ;YES, MATCH. GO MOVE THE DATA
       DØC4 1A                  LDAX     D                   ;GET A BYTE TO COMPARE
       DØC5 BE                  CMP      M                   ;TEST FOR MATCH
       DØC6 23                  INX      H                   ;BUMP POINTERS TO NEXT DATA ITEM
       DØC7 13                  INX      D
       DØC8 CACØDØ              JZ       DTSLOP              ;MATCH, CONTINUE TESTING

              *****************************************************************
              *                                                             *
              * DRIVE, TRACK, AND SECTOR DON'T MATCH, FLUSH THE BUFFER IF    *
              * NECESSARY AND THEN REFILL.                                   *
              *                                                             *
              *****************************************************************

       DØCB CD96D1              CALL     FILL                ;FILL THE BUFFER WITH CORRECT PHYSICAL SECTOR
       DØCE D8                  RC                           ;NO GOOD, RETURN WITH ERROR INDICATION

              *****************************************************************
              *                                                             *
              * MOVE HAS BEEN MODIFIED TO CAUSE EITHER A TRANSFER INTO OR OUT *
              * THE BUFFER.                                                  *
              *                                                             *
              *****************************************************************

       DØCF 3AF5D4   MOVE       LDA      CPMSEC              ;GET THE CP/M SECTOR TO TRANSFER
       DØD2 3D                  DCR      A                   ;ADJUST TO PROPER SECTOR IN BUFFER
       DØD3 E600                ANI      Ø                   ;STRIP OFF HIGH ORDERED BITS
       DØD4 =        SECPSEC    EQU      $-1                 ;THE Ø IS MODIFIED TO REPRESENT THE # OF
                                                             ;         CP/M SECTORS PER PHYSICAL SECTORS
       DØD5 6F                  MOV      L,A                 ;PUT INTO HL
       DØD6 2600                MVI      H,Ø
       DØD8 29                  DAD      H                   ;FORM OFFSET INTO BUFFER
       DØD9 29                  DAD      H
       DØDA 29                  DAD      H
       DØDB 29                  DAD      H
       DØDC 29                  DAD      H
       DØDD 29                  DAD      H
       DØDE 29                  DAD      H
       DØDF 11FFD4              LXI      D,BUFFER            ;BEGINNING ADDRESS OF BUFFER
       DØE2 19                  DAD      D                   ;FORM BEGINNING ADDRESS OF SECTOR TO TRANSFER
       DØE3 EB                  XCHG                         ;DE = ADDRESS IN BUFFER
       DØE4 210000              LXI      H,Ø                 ;GET DMA ADDRESS, THE Ø IS MODIFIED TO
                                                             ;         CONTAIN THE DMA ADDRESS
```

```
D0E5 =          CPMDMA  EQU     $-2
D0E7 3E00               MVI     A,0                     ;THE ZERO GETS MODIFIED TO CONTAIN
                                                        ;       A ZERO IF A READ, OR A 1 IF WRITE
D0E8 =          RDWR    EQU     $-1
D0E9 A7                 ANA     A                       ;TEST WHICH KIND OF OPERATION
D0EA C2F2D0             JNZ     INTO                    ;TRANSFER DATA INTO THE BUFFER
D0ED CDCAD1     OUTOF   CALL    MOVER
D0F0 AF                 XRA     A
D0F1 C9                 RET

D0F2 EB         INTO    XCHG                            ;
D0F3 CDCAD1             CALL    MOVER                   ;MOVE THE DATA, HL = DESTINATION
                                                        ;       DE = SOURCE
D0F6 3E01               MVI     A,1
D0F8 3205D1             STA     BUFWRTN                 ;SET BUFFER WRITTEN INTO FLAG
D0FB 3E00               MVI     A,0                     ;CHECK FOR DIRECTORY WRITE
D0FC =          WRITTYP EQU     $-1
D0FD 3D                 DCR     A
D0FE 3E00               MVI     A,0
D100 32FCD0             STA     WRITTYP                 ;SET NO DIRECTORY WRITE
D103 C0                 RNZ                             ;NO ERROR EXIT
```

```
        ********************************************************************
        *                                                                  *
        * FLUSH WRITES THE CONTENTS OF THE BUFFER OUT TO THE DISK IF        *
        * IT HAS EVER BEEN WRITTEN INTO.                                    *
        *                                                                  *
        ********************************************************************
```

```
D104 3E00       FLUSH   MVI     A,0                     ;THE 0 IS MODIFIED TO REFLECT IF
                                                        ;       THE BUFFER HAS BEEN WRITTEN INTO
D105 =          BUFWRTN EQU     $-1
D106 A7                 ANA     A                       ;TEST IF WRITTEN INTO
D107 C8                 RZ                              ;NOT WRITTEN, ALL DONE

                IF      (MAXHD NE 0) AND (MAXFLOP NE 0)
D108 2118E4             LXI     H,DJWRITE               ;WRITE OPERATION FOR DISK JOCKEY
D10B 1192D2             LXI     D,HDWRITE               ;WRITE OPERATION FOR HARD DISK
D10E CDD9D1             CALL    DECIDE
                ELSE
                IF      MAXHD NE 0
                LXI     H,HDWRITE
                ENDIF
                IF      MAXFLOP NE 0
                LXI     H,DJWRITE
                ENDIF
                ENDIF
```

```
        ********************************************************************
        *                                                                  *
        * PREP PREPARES TO READ/WRITE THE DISK. RETRIES ARE ATTEMPTED.     *
        * UPON ENTRY, H&L MUST CONTAIN THE READ OR WRITE OPERATION         *
        * ADDRESS.                                                         *
        *                                                                  *
        ********************************************************************
```

```
        D111 AF        PREP     XRA     A                    ;RESET BUFFER WRITTEN FLAG
        D112 3205D1             STA     BUFWRTN
        D115 2277D1             SHLD    RETRYOP              ;SET UP THE READ/WRITE OPERATION
        D118 060A              MVI     B,RETRIES            ;MAXIMUM NUMBER OF RETRIES TO ATTEMPT
        D11A C5        RETRYLP PUSH     B                    ;SAVE THE RETRY COUNT
        D11B 3AFBD4            LDA     BUFDRV               ;GET DRIVE NUMBER INVOLVED IN THE OPERATION

                               IF      (MAXHD NE 0) AND (MAXFLOP NE 0)
                               IF      FIRST
                               CPI     MAXHD*LOGDSK
                               JC      NOADJST
                               SUI     MAXHD*LOGDSK
                               ELSE
        D11E FE02              CPI     MAXFLOP
        D120 DA25D1            JC      NOADJST
        D123 D602              SUI     MAXFLOP
                               ENDIF

        D125 4F        NOADJST MOV     C,A
        D126 2133CD            LXI     H,DJDRV              ;SELECT DRIVE
        D129 11E1D1            LXI     D,HDDRV
        D12C CDD5D1            CALL    DECIDGO
                               ELSE
                               MOV     C,A
                               IF      MAXHD NE 0
                               CALL    HDDRV
                               ENDIF
                               IF      MAXFLOP NE 0
                               CALL    DJDRV               ;SELECT THE DRIVE
                               ENDIF
                               ENDIF

        D12F 3AFCD4            LDA     BUFTRK
        D132 A7               ANA     A                    ;TEST FOR TRACK ZERO
        D133 4F               MOV     C,A
        D134 C5               PUSH    B

                               IF      (MAXHD NE 0) AND (MAXFLOP NE 0)
        D135 2109E4            LXI     H,DJHOME
        D138 11F2D1            LXI     D,HDHOME
        D13B CCD5D1            CZ      DECIDGO
                               ELSE
                               IF      MAXHD NE 0
                               CZ      HDHOME
                               ENDIF
                               IF      MAXFLOP NE 0
                               CZ      DJHOME              ;HOME THE DRIVE IF TRACK 0
                               ENDIF
                               ENDIF

        D13E C1               POP     B                    ;RESTORE TRACK #

                               IF      (MAXHD NE 0) AND (MAXFLOP NE 0)
        D13F 210CE4            LXI     H,DJTRK
        D142 1113D2            LXI     D,HDTRK
        D145 CDD5D1            CALL    DECIDGO
```

```
                         ELSE
                         IF      MAXHD NE 0
                         CALL    HDTRK
                         ENDIF
                         IF      MAXFLOP NE 0
                         CALL    DJTRK              ;SEEK TO PROPER TRACK
                         ENDIF
                         ENDIF

D148 2AFDD4              LHLD    BUFSEC
D14B 7C                  MOV     A,H                ;GET SECTOR INVOLVED IN OPERATION
D14C 07                  RLC                        ;BIT 0 OF A EQUALS SIDE #
D14D E601                ANI     1                  ;STRIP OFF UNNECESSARY BITS
D14F 4F                  MOV     C,A                ;C <- SIDE #

                         IF      (MAXHD NE 0) AND (MAXFLOP NE 0)
D150 2130E4              LXI     H,DJSIDE
D153 113FD2              LXI     D,HDSIDE
D156 CDD5D1              CALL    DECIDGO
                         ELSE
                         IF      MAXHD NE 0
                         CALL    HDSIDE
                         ENDIF
                         IF      MAXFLOP NE 0
                         CALL    DJSIDE             ;SELECT THE SIDE
                         ENDIF
                         ENDIF

D159 2AFDD4              LHLD    BUFSEC
D15C 7C                  MOV     A,H
D15D E67F                ANI     7FH                ;STRIP OFF SIDE BIT
D15F 47                  MOV     B,A                ;C <- SECTOR #
D160 4D                  MOV     C,L

                         IF      (MAXHD NE 0) AND (MAXFLOP NE 0)
D161 210FE4              LXI     H,DJSEC
D164 1148D2              LXI     D,HDSEC
D167 CDD5D1              CALL    DECIDGO
                         ELSE
                         IF      MAXHD NE 0
                         CALL    HDSEC
                         ENDIF
                         IF      MAXFLOP NE 0
                         CALL    DJSEC              ;SELECT THE SIDE
                         ENDIF
                         ENDIF

D16A 01FFD4              LXI     B,BUFFER           ;SET THE DMA ADDRESS

                         IF      (MAXHD NE 0) AND (MAXFLOP NE 0)
D16D 2112E4              LXI     H,DJDMA
D170 113AD2              LXI     D,HDDMA
D173 CDD5D1              CALL    DECIDGO
                         ELSE
                         IF      MAXHD NE 0
                         CALL    HDDMA
```

```
                          ENDIF
                          IF        MAXFLOP NE Ø
                          CALL      DJDMA               ;SELECT THE SIDE
                          ENDIF
                          ENDIF

D176 CDØØØØ               CALL      Ø                   ;GET OPERATION ADDRESS
D177 =          RETRYOP   EQU       $-2
D179 C1                   POP       B                   ;RESTORE THE RETRY COUNTER
D17A 3EØØ                 MVI       A,Ø                 ;NO ERROR EXIT STATUS
D17C DØ                   RNC                           ;RETURN NO ERROR
D17D Ø5                   DCR       B                   ;UPDATE THE RETRY COUNTER
D17E 37                   STC                           ;ASSUME RETRY COUNT EXPIRED
D17F 3EFF                 MVI       A,ØFFH              ;ERROR RETURN
D181 C8                   RZ
D182 78                   MOV       A,B
D183 FEØ5                 CPI       RETRIES/2
D185 C21AD1               JNZ       RETRYLP             ;TRY AGAIN

D188 C5                   PUSH      B
                          IF        (MAXHD NE Ø) AND (MAXFLOP NE Ø)
D189 21Ø9E4               LXI       H,DJHOME
D18C 11F2D1               LXI       D,HDHOME
D18F CDD5D1               CALL      DECIDGO
                          ELSE
                          IF        MAXHD NE Ø
                          CALL      HDHOME
                          ENDIF
                          IF        MAXFLOP NE Ø
                          CALL      DJHOME              ;HOME THE DRIVE IF TRACK Ø
                          ENDIF
                          ENDIF

D192 C1                   POP       B
D193 C31AD1               JMP       RETRYLP
```

```
            ********************************************************************
            *                                                                  *
            * FILL FILLS THE BUFFER WITH A NEW SECTOR FROM THE DISK.            *
            *                                                                  *
            ********************************************************************
```

```
D196 CDØ4D1     FILL      CALL      FLUSH               ;FLUSH BUFFER FIRST
D199 D8                   RC                            ;CHECK FOR ERROR
D19A 11F7D4               LXI       D,CPMDRV            ;UPDATE THE DRIVE, TRACK, AND SECTOR
D19D 21FBD4               LXI       H,BUFDRV
D1AØ Ø6Ø4                 MVI       B,4                 ;NUMBER OF BYTES TO MOVE
D1A2 CDCCD1               CALL      MOVLOP              ;COPY THE DATA

D1A5 3AE8DØ               LDA       RDWR
D1A8 A7                   ANA       A
D1A9 CABED1               JZ        FREAD
D1AC 3AFCDØ               LDA       WRITTYP
D1AF 3D                   DCR       A
D1BØ 3D                   DCR       A
D1B1 C8                   RZ
```

```
D1B2 CD69D0              CALL    GETDPB
D1B5 110F00              LXI     D,15
D1B8 19                  DAD     D
D1B9 7E                  MOV     A,M
D1BA E603               ANI     3
D1BC 3D                  DCR     A
D1BD C8                  RZ

D1BE =          FREAD    EQU     $
                         IF      (MAXHD NE 0) AND (MAXFLOP NE 0)
D1BE 2115E4              LXI     H,DJREAD
D1C1 115DD2              LXI     D,HDREAD
D1C4 CDD9D1              CALL    DECIDE
                         ELSE
                         IF      MAXHD NE 0
                         LXI     H,HDREAD
                         ENDIF
                         IF      MAXFLOP NE 0
                         LXI     H,DJREAD               ;SELECT THE SIDE
                         ENDIF
                         ENDIF
D1C7 C311D1              JMP     PREP                   ;SELECT DRIVE, TRACK, AND SECTOR.
                                                        ;       THEN READ THE BUFFER


        *******************************************************************
        *                                                                 *
        * MOVER MOVES 128 BYTES OF DATA. SOURCE POINTER IN DE, DEST        *
        * POINTER IN HL.                                                   *
        *                                                                 *
        *******************************************************************

D1CA 0680       MOVER    MVI     B,128                  ;LENGTH OF TRANSFER
D1CC 1A         MOVLOP   LDAX    D                      ;GET A BTE OF SOURCE
D1CD 77                  MOV     M,A                    ;MOVE IT
D1CE 13                  INX     D                      ;BUMP POINTERS
D1CF 23                  INX     H
D1D0 05                  DCR     B                      ;UPDATE COUNTER
D1D1 C2CCD1              JNZ     MOVLOP                 ;CONTINUE MOVING UNTIL DONE
D1D4 C9                  RET


        *******************************************************************
        *                                                                 *
        * ROUTINES TO DECIDE WHICH CONTROLLER TO USE.                     *
        *                                                                 *
        *******************************************************************

                         IF      (MAXHD NE 0) AND (MAXFLOP NE 0)
D1D5 CDD9D1     DECIDGO  CALL    DECIDE  ;WHICH CONTROLLER ?
D1D8 E9                  PCHL
                         ENDIF

                         IF      (MAXHD NE 0) AND (MAXFLOP NE 0)
D1D9 3AFBD4     DECIDE   LDA     BUFDRV                 ;GET PROPER ROUTINE INTO H&L, BASED
                         IF      FIRST                  ;    ON CURRENTLY SELECTED DRIVE
                         CPI     MAXHD*LOGDSK
                         RNC
```

```
                             ELSE
D1DC FE02               CPI     MAXFLOP
D1DE D8                 RC
                             ENDIF
D1DF EB                 XCHG
D1E0 C9                 RET
                             ENDIF


        ********************************************************************
        *                                                                  *
        * THE FOLLOWING IS THE EQUIVALENT OF THE LOWEST LEVEL DRIVERS      *
        * FOR THE HARD DISK.                                               *
        *                                                                  *
        ********************************************************************


                        IF      MAXHD NE 0
D1E1 79        HDDRV    MOV     A,C                 ;SELECT HARD DISK DRIVE
D1E2 CD60D0             CALL    DIVLOG              ;GET THE PHYSICAL DRIVE #
D1E5 79                 MOV     A,C
D1E6 3222D3             STA     HDDISK              ;SELECT THE DRIVE
D1E9 F6FC               ORI     NULL
D1EB D352               OUT     HDFUNC
D1ED 3E0F               MVI     A,WENABL
D1EF D350               OUT     HDCNTL
D1F1 C9                 RET


D1F2 CD10D3    HDHOME   CALL    DRVPTR
D1F5 3600               MVI     M,0                 ;SET TRACK TO ZERO

                        IF      SDELAY
D1F7 DB50      STEPO    IN      HDSTAT              ;TEST STATUS
D1F9 E601               ANI     TKZERO              ;AT TRACK ZERO ?
D1FB CA07D2             JZ      DELAY
D1FE 3E01               MVI     A,1
D200 37                 STC
D201 CD27D2             CALL    ACCOK               ;TAKE ONE STEP OUT
D204 C3F7D1             JMP     STEPO

                        ELSE

                        IN      HDSTAT
                        ANI     TKZERO
                        RZ
                        XRA     A
                        JMP     ACCOK
                        ENDIF

                        IF      SDELAY
D207 210000    DELAY    LXI     H,0                 ;GET DELAY
D208 =         SETTLE   EQU     $-2
D20A 2B        DELOOP   DCX     H                   ;WAIT 20MS
D20B 7C                 MOV     A,H
D20C B5                 ORA     L
D20D 23                 INX     H
D20E 2B                 DCX     H
D20F C20AD2             JNZ     DELOOP
```

```
D212 C9                       RET
                              ENDIF

D213 CD10D3      HDTRK        CALL    DRVPTR          ;GET POINTER TO CURRENT TRACK
D216 5E                       MOV     E,M             ;GET CURRENT TRACK
D217 71                       MOV     M,C             ;UPDATE THE TRACK
D218 7B                       MOV     A,E             ;NEED TO SEEK AT ALL ?
D219 91                       SUB     C
D21A C8                       RZ
D21B 3F                       CMC                     ;GET CARRY INTO DIRECTION
D21C DA21D2                   JC      HDTRK2
D21F 2F                       CMA
D220 3C                       INR     A
                              IF      NOT SDELAY
                 HDTRK2       JMP     ACCOK
                              ELSE
D221 CD27D2      HDTRK2       CALL    ACCOK
D224 C307D2                   JMP     DELAY
                              ENDIF

D227 47          ACCOK        MOV     B,A             ;PREP FOR BUILD
D228 CD1BD3                   CALL    BUILD
D22B E6FB        SLOOP        ANI     NSTEP           ;GET STEP PULSE LOW
D22D D352                     OUT     HDFUNC          ;OUTPUT LOW STEP LINE
D22F F604                     ORI     PSTEP           ;SET STEP LINE HIGH
D231 D352                     OUT     HDFUNC          ;OUTPUT HIGH STEP LINE
D233 05                       DCR     B               ;UPDATE REPEAT COUNT
D234 C22BD2                   JNZ     SLOOP           ;KEEP GOING THE REQUIRED # OF TRACKS
D237 C340D2                   JMP     WSDONE

D23A 60          HDDMA        MOV     H,B             ;SAVE THE DMA ADDRESS
D23B 69                       MOV     L,C
D23C 2277D2                   SHLD    HDADD
D23F =           HDSIDE       EQU     $
D23F C9                       RET

D240 DB50        WSDONE       IN      HDSTAT          ;WAIT FOR SEEK COMPLETE TO FINISH
D242 E604                     ANI     COMPLT
D244 CA40D2                   JZ      WSDONE
D247 C9                       RET

                              IF      M26
D248 3E1F        HDSEC        MVI     A,01FH          ;FOR COMPATIBILITY WITH CBIOS REV 2.3, 2.4
D24A A1                       ANA     C
D24B CC5AD2                   CZ      GETSPT
D24E 3200D3                   STA     HDSECTR
D251 3EE0                     MVI     A,0E0H
D253 A1                       ANA     C
D254 07                       RLC
D255 07                       RLC
D256 07                       RLC
D257 321CD3                   STA     HEAD
D25A 3E20        GETSPT       MVI     A,HDSPT
D25C C9                       RET

                              ELSE
```

```
                    HDSEC     MOV     A,C
                              CALL    DIVSPT
                              ADI     HDSPT
                              ANA     A
                              CZ      GETSPT
                              STA     HDSECTR
                              MOV     A,C
                              STA     HEAD
                    GETSPT    MVI     A,HDSPT
                              DCR     C
                              RET


                    DIVSPT    MVI     C,0
                    DIVSPTX   SUI     HDSPT
                              RC
                              INR     C
                              JMP     DIVSPTX
                              ENDIF

D25D CDDBD2         HDREAD    CALL    HDPREP
D260 D8                       RC
D261 AF                       XRA     A
D262 D351                     OUT     HDCMND
D264 2F                       CMA
D265 D353                     OUT     HDDATA
D267 D353                     OUT     HDDATA
D269 3E01                     MVI     A,RSECT          ;READ SECTOR COMMAND
D26B D351                     OUT     HDCMND
D26D CDC1D2                   CALL    PROCESS
D270 D8                       RC
D271 AF                       XRA     A
D272 D351                     OUT     HDCMND
D274 0680                     MVI     B,SECLEN/4
D276 210000                   LXI     H,0
D277 =              HDADD     EQU     $-2
D279 DB53                     IN      HDDATA
D27B DB53                     IN      HDDATA
D27D DB53           RTLOOP    IN      HDDATA           ;MOVE FOUR BYTES
D27F 77                       MOV     M,A
D280 23                       INX     H
D281 DB53                     IN      HDDATA
D283 77                       MOV     M,A
D284 23                       INX     H
D285 DB53                     IN      HDDATA
D287 77                       MOV     M,A
D288 23                       INX     H
D289 DB53                     IN      HDDATA
D28B 77                       MOV     M,A
D28C 23                       INX     H
D28D 05                       DCR     B
D28E C27DD2                   JNZ     RTLOOP
D291 C9                       RET

D292 CDDBD2         HDWRITE   CALL    HDPREP           ;PREPARE HEADER
D295 D8                       RC
```

```
D296 AF                 XRA     A
D297 D351               OUT     HDCMND
D299 2A77D2             LHLD    HDADD
D29C 0680               MVI     B,SECLEN/4
D29E 7E        WTLOOP   MOV     A,M             ;MOVE 4 BYTES
D29F D353               OUT     HDDATA
D2A1 23                 INX     H
D2A2 7E                 MOV     A,M
D2A3 D353               OUT     HDDATA
D2A5 23                 INX     H
D2A6 7E                 MOV     A,M
D2A7 D353               OUT     HDDATA
D2A9 23                 INX     H
D2AA 7E                 MOV     A,M
D2AB D353               OUT     HDDATA
D2AD 23                 INX     H
D2AE 05                 DCR     B
D2AF C29ED2             JNZ     WTLOOP
D2B2 3E05               MVI     A,WSECT         ;ISSUE WRITE SECTOR COMMAND
D2B4 D351               OUT     HDCMND
D2B6 CDC1D2             CALL    PROCESS
D2B9 D8                 RC
D2BA 3E10               MVI     A,WFAULT
D2BC A0                 ANA     B
D2BD 37                 STC
D2BE C8                 RZ
D2BF AF                 XRA     A
D2C0 C9                 RET


D2C1 DB50      PROCESS  IN      HDSTAT          ;WAIT FOR COMMAND TO FINISH
D2C3 47                 MOV     B,A
D2C4 E602               ANI     OPDONE
D2C6 CAC1D2             JZ      PROCESS
D2C9 3E07               MVI     A,DSKCLK
D2CB D350               OUT     HDCNTL
D2CD DB50               IN      HDSTAT
D2CF E608               ANI     TMOUT           ;TIMED OUT ?
D2D1 37                 STC
D2D2 C0                 RNZ
D2D3 DB51               IN      HDRESLT
D2D5 E602               ANI     RETRY           ;ANY RETRIES ?
D2D7 37                 STC
D2D8 C0                 RNZ
D2D9 AF                 XRA     A
D2DA C9                 RET


D2DB DB50      HDPREP   IN      HDSTAT
D2DD E620               ANI     DRVRDY
D2DF 37                 STC
D2E0 C0                 RNZ
D2E1 3E08               MVI     A,ISBUFF        ;INITIALIZE POINTER
D2E3 D351               OUT     HDCMND
D2E5 CD1BD3             CALL    BUILD
D2E8 F60C               ORI     0CH
D2EA D352               OUT     HDFUNC
D2EC 3A1CD3             LDA     HEAD
```

```
D2EF D353            OUT     HDDATA          ;FORM HEAD BYTE
D2F1 CD10D3          CALL    DRVPTR
D2F4 7E              MOV     A,M             ;FORM TRACK BYTE
D2F5 D353            OUT     HDDATA
D2F7 A7              ANA     A
D2F8 0680            MVI     B,80H
D2FA CAFFD2          JZ      ZKEY
D2FD 0600            MVI     B,0
D2FF 3E00     ZKEY   MVI     A,0             ;FORM SECTOR BYTE
D300 =        HDSECTR EQU    $-1
D301 D353            OUT     HDDATA
D303 78              MOV     A,B
D304 D353            OUT     HDDATA
D306 3E07            MVI     A,DSKCLK
D308 D350            OUT     HDCNTL
D30A 3E0F            MVI     A,WENABL
D30C D350            OUT     HDCNTL
D30E AF              XRA     A
D30F C9              RET

D310 2A22D3   DRVPTR LHLD    HDDISK
D313 EB              XCHG
D314 1600            MVI     D,0
D316 2126D3          LXI     H,DRIVES
D319 19              DAD     D
D31A C9              RET

D31B 3E00     BUILD  MVI     A,0
D31C =        HEAD   EQU     $-1
D31D 17              RAL
D31E 17              RAL
D31F 17              RAL
D320 17              RAL
D321 F600            ORI     0
D322 =        HDDISK EQU     $-1
D323 EEF0            XRI     0F0H
D325 C9              RET

D326 =        DRIVES EQU     $
                     REPT    MAXHD
                     DB      0FFH
                     ENDM
D326+FF              DB      0FFH
                     ENDIF
```

```
***********************************************************************
*                                                                     *
* XLT TABLES (SECTOR SKEW TABLES) FOR CP/M 2.0. THESE TABLES          *
* DEFINE THE SECTOR TRANSLATION THAT OCCURS WHEN MAPPING CP/M         *
* SECTORS TO PHYSICAL SECTORS ON THE DISK. THERE IS ONE SKEW          *
* TABLE FOR EACH OF THE POSSIBLE SECTOR SIZES. CURRENTLY THE          *
* TABLES ARE LOCATED ON TRACK 0 SECTORS 6 AND 8. THEY ARE             *
* LOADED INTO MEMORY IN THE CBIOS RAM BY THE COLD BOOT ROUTINE.       *
*                                                                     *
***********************************************************************
```

```
                        IF      MAXFLOP NE 0
D327 00        XLT128 DB        0
D328 01070D1319        DB       1,7,13,19,25
D32D 050B1117          DB       5,11,17,23
D331 03090F15          DB       3,9,15,21
D335 02080E141A        DB       2,8,14,20,26
D33A 060C1218          DB       6,12,18,24
D33E 040A1016          DB       4,10,16,22

D342 00        XLT256 DB        0
D343 0102131425        DB       1,2,19,20,37,38
D349 0304151627        DB       3,4,21,22,39,40
D34F 0506171829        DB       5,6,23,24,41,42
D355 0708191A2B        DB       7,8,25,26,43,44
D35B 090A1B1C2D        DB       9,10,27,28,45,46
D361 0B0C1D1E2F        DB       11,12,29,30,47,48
D367 0D0E1F2031        DB       13,14,31,32,49,50
D36D 0F10212233        DB       15,16,33,34,51,52
D373 11122324          DB       17,18,35,36

D377 00        XLT512 DB        0
D378 0102030411        DB       1,2,3,4,17,18,19,20
D380 2122232431        DB       33,34,35,36,49,50,51,52
D388 0506070815        DB       5,6,7,8,21,22,23,24
D390 2526272835        DB       37,38,39,40,53,54,55,56
D398 090A0B0C19        DB       9,10,11,12,25,26,27,28
D3A0 292A2B2C39        DB       41,42,43,44,57,58,59,60
D3A8 0D0E0F101D        DB       13,14,15,16,29,30,31,32
D3B0 2D2E2F30          DB       45,46,47,48

D3B4 00        XLT124 DB        0
D3B5 0102030405        DB       1,2,3,4,5,6,7,8
D3BD 191A1B1C1D        DB       25,26,27,28,29,30,31,32
D3C5 3132333435        DB       49,50,51,52,53,54,55,56
D3CD 090A0B0C0D        DB       9,10,11,12,13,14,15,16
D3D5 2122232425        DB       33,34,35,36,37,38,39,40
D3DD 393A3B3C3D        DB       57,58,59,60,61,62,63,64
D3E5 1112131415        DB       17,18,19,20,21,22,23,24
D3ED 292A2B2C2D        DB       41,42,43,44,45,46,47,48
```

```
*****************************************************************
*                                                               *
* EACH OF THE FOLLOWING TABLES DESCRIBES A DISKETTE WITH THE     *
* SPECIFIED CHARACTERISTICS.                                     *
*                                                               *
*****************************************************************



*****************************************************************
*                                                               *
* THE FOLLOWING DPB DEFINES A  DISKETTE FOR 128 BYTE SECTORS,    *
* SINGLE DENSITY, AND SINGLE SIDED.                              *
*                                                               *
*****************************************************************
```

```
D3F5 1A00      DPB128S DW       26              ;CP/M SECTORS/TRACK
```

```
D3F7 03                DB      3              ;BSH
D3F8 07                DB      7              ;BLM
D3F9 00                DB      0              ;EXM
D3FA F200              DW      242            ;DSM
D3FC 3F00              DW      63             ;DRM
D3FE C0                DB      0C0H           ;AL0
D3FF 00                DB      0              ;AL1
D400 1000              DW      16             ;CKS
D402 0200              DW      2              ;OFF
D404 01                DB      1H             ;16*((#CPM SECTORS/PHYSICAL SECTOR) -1) +
                                              ;LOG2(#BYTES PER SECTOR/128) + 1 +
                                              ;8 IF DOUBLE SIDED.


        ********************************************************************
        *                                                                  *
        * THE FOLLOWING DPB DEFINES A DISKETTE FOR 256 BYTE SECTORS,       *
        * DOUBLE DENSITY, AND SINGLE SIDED.                                *
        *                                                                  *
        ********************************************************************

D405 3400     DPB256S  DW      52             ;CP/M SECTORS/TRACK
D407 04                DB      4              ;BSH
D408 0F                DB      15             ;BLM
D409 00                DB      0              ;EXM
D40A F200              DW      242            ;DSM
D40C 7F00              DW      127            ;DRM
D40E C0                DB      0C0H           ;AL0
D40F 00                DB      0              ;AL1
D410 2000              DW      32             ;CKS
D412 0200              DW      2              ;OFF
D414 12                DB      12H            ;16*((#CPM SECTORS/PHYSICAL SECTOR) -1) +
                                              ;LOG2(#BYTES PER SECTOR/128) + 1 +
                                              ;8 IF DOUBLE SIDED.


        ********************************************************************
        *                                                                  *
        * THE FOLLOWING DPB DEFINES A DISKETTE AS 512 BYTE SECTORS,        *
        * DOUBLE DENSITY, AND SINGLE SIDED.                                *
        *                                                                  *
        ********************************************************************

D415 3C00     DPB512S  DW      60             ;CP/M SECTORS/TRACK
D417 04                DB      4              ;BSH
D418 0F                DB      15             ;BLM
D419 00                DB      0              ;EXM
D41A 1801              DW      280            ;DSM
D41C 7F00              DW      127            ;DRM
D41E C0                DB      0C0H           ;AL0
D41F 00                DB      0              ;AL1
D420 2000              DW      32             ;CKS
D422 0200              DW      2              ;OFF
D424 33                DB      33H            ;16*((#CPM SECTORS/PHYSICAL SECTOR) -1) +
                                              ;LOG2(#BYTES PER SECTOR/128) + 1 +
                                              ;8 IF DOUBLE SIDED.


        ********************************************************************
```

```
                          *                                                    *
                          * THE FOLLOWING DPB DEFINES A DISKETTE AS 1024 BYTE SECTORS,   *
                          * DOUBLE DENSITY, AND SINGLE SIDED.                            *
                          *                                                    *
                          ************************************************************************

D425 4000      DP1024S DW      64            ;CP/M SECTORS/TRACK
D427 04                DB      4             ;BSH
D428 0F                DB      15            ;BLM
D429 00                DB      0             ;EXM
D42A 2B01              DW      299           ;DSM
D42C 7F00              DW      127           ;DRM
D42E C0                DB      0C0H          ;AL0
D42F 00                DB      0             ;AL1
D430 2000              DW      32            ;CKS
D432 0200              DW      2             ;OFF
D434 74                DB      74H           ;16*((#CPM SECTORS/PHYSICAL SECTOR) -1) +
                                             ;LOG2(#BYTES PER SECTOR/128) + 1 +
                                             ;8 IF DOUBLE SIDED.


                          ************************************************************************
                          *                                                    *
                          * THE FOLLOWING DPB DEFINES A DISKETTE FOR 128 BYTE SECTORS,   *
                          * SINGLE DENSITY, AND DOUBLE SIDED.                           *
                          *                                                    *
                          ************************************************************************

D435 3400      DPB128D DW      52            ;CP/M SECTORS/TRACK
D437 04                DB      4             ;BSH
D438 0F                DB      15            ;BLM
D439 01                DB      1             ;EXM
D43A F200              DW      242           ;DSM
D43C 7F00              DW      127           ;DRM
D43E C0                DB      0C0H          ;AL0
D43F 00                DB      0             ;AL1
D440 2000              DW      32            ;CKS
D442 0200              DW      2             ;OFF
D444 09                DB      9H


                          ************************************************************************
                          *                                                    *
                          * THE FOLLOWING DPB DEFINES A DISKETTE AS 256 BYTE SECTORS,   *
                          * DOUBLE DENSITY, AND DOUBLE SIDED.                          *
                          *                                                    *
                          ************************************************************************

D445 6800      DPB256D DW      104           ;CP/M SECTORS/TRACK
D447 04                DB      4             ;BSH
D448 0F                DB      15            ;BLM
D449 00                DB      0             ;EXM
D44A E601              DW      486           ;DSM
D44C FF00              DW      255           ;DRM
D44E F0                DB      0F0H          ;AL0
D44F 00                DB      0             ;AL1
D450 4000              DW      64            ;CKS
D452 0200              DW      2             ;OFF
```

```
D454 1A                 DB      1AH

          ********************************************************************
          *                                                                  *
          * THE FOLLOWING DPB DEFINES A DISKETTE AS 512 BYTE SECTORS,         *
          * DOUBLE DENSITY, AND DOUBLE SIDED.                                 *
          *                                                                  *
          ********************************************************************

D455 7800    DPB512D DW      120                 ;CP/M SECTORS/TRACK
D457 04              DB      4                   ;BSH
D458 0F              DB      15                  ;BLM
D459 00              DB      0                   ;EXM
D45A 3102            DW      561                 ;DSM
D45C FF00            DW      255                 ;DRM
D45E F0              DB      0F0H                ;AL0
D45F 00              DB      0                   ;AL1
D460 4000            DW      64                  ;CKS
D462 0200            DW      2                   ;OFF
D464 3B              DB      3BH


          ********************************************************************
          *                                                                  *
          * THE FOLLOWING DPB DEFINES A DISKETTE AS 1024 BYTE SECTORS,        *
          * DOUBLE DENSITY, AND DOUBLE SIDED.                                 *
          *                                                                  *
          ********************************************************************

D465 8000    DP1024D DW      128                 ;CP/M SECTORS/TRACK
D467 04              DB      4                   ;BSH
D468 0F              DB      15                  ;BLM
D469 00              DB      0                   ;EXM
D46A 5702            DW      599                 ;DSM
D46C FF00            DW      255                 ;DRM
D46E F0              DB      0F0H                ;AL0
D46F 00              DB      0                   ;AL1
D470 4000            DW      64                  ;CKS
D472 0200            DW      2                   ;OFF
D474 7C              DB      7CH
                     ENDIF


          ********************************************************************
          *                                      26                          *
          * THE FOLLOWING DPB DEFINES A 20 MEGABYTE HARD DISK, WITH 512       *
          * BYTE SECTORS.                                                     *
          *                                                                  *
          ********************************************************************

                     IF      MAXHD NE 0
                     IF      M26 NE 0
D475 0004    DPBHD1  DW      1024                ;CP/M SECTORS/TRACK
D477 05              DB      5                   ;BSH
D478 1F              DB      31                  ;BLM
D479 01              DB      1                   ;EXM
D47A B507            DW      1973                ;DSM
D47C FF01            DW      511                 ;DRM
```

```
D47E FF                    DB      ØFFH             ;ALØ
D47F FF                    DB      ØFFH             ;AL1
D48Ø ØØØØ                  DW      Ø                ;CKS
D482 Ø1ØØ                  DW      1                ;OFF
D484 33                    DB      33H              ;16*((#CPM SECTORS/PHYSICAL SECTOR) -1) +
                                                    ;LOG2(#BYTES PER SECTOR/128) + 1 +
                                                    ;8 IF DOUBLE SIDED.

D485 ØØØ4      DPBHD2      DW      1Ø24             ;CP/M SECTORS/TRACK
D487 Ø5                    DB      5                ;BSH
D488 1F                    DB      31               ;BLM
D489 Ø1                    DB      1                ;EXM
D48A B5Ø7                  DW      1973             ;DSM
D48C FFØ1                  DW      511              ;DRM
D48E FF                    DB      ØFFH             ;ALØ
D48F FF                    DB      ØFFH             ;AL1
D49Ø ØØØØ                  DW      Ø                ;CKS
D492 4ØØØ                  DW      64               ;OFF
D494 33                    DB      33H              ;16*((#CPM SECTORS/PHYSICAL SECTOR) -1) +
                                                    ;LOG2(#BYTES PER SECTOR/128) + 1 +
                                                    ;8 IF DOUBLE SIDED.

D495 ØØØ4      DPBHD3      DW      1Ø24             ;CP/M SECTORS/TRACK
D497 Ø5                    DB      5                ;BSH
D498 1F                    DB      31               ;BLM
D499 Ø1                    DB      1                ;EXM
D49A B5Ø7                  DW      1973             ;DSM
D49C FFØ1                  DW      511              ;DRM
D49E FF                    DB      ØFFH             ;ALØ
D49F FF                    DB      ØFFH             ;AL1
D4AØ ØØØØ                  DW      Ø                ;CKS
D4A2 7FØØ                  DW      127              ;OFF
D4A4 33                    DB      33H              ;16*((#CPM SECTORS/PHYSICAL SECTOR) -1) +
                                                    ;LOG2(#BYTES PER SECTOR/128) + 1 +
                                                    ;8 IF DOUBLE SIDED.

               ENDIF
               IF      M1Ø NE Ø
               DPBHD1      DW      336              ;CP/M SECTORS/TRACK
                           DB      5                ;BSH
                           DB      31               ;BLM
                           DB      1                ;EXM
                           DW      1269             ;DSM
                           DW      511              ;DRM
                           DB      ØFFH             ;ALØ
                           DB      ØFFH             ;AL1
                           DW      Ø                ;CKS
                           DW      1                ;OFF
                           DB      33H              ;16*((#CPM SECTORS/PHYSICAL SECTOR) -1) +
                                                    ;LOG2(#BYTES PER SECTOR/128) + 1 +
                                                    ;8 IF DOUBLE SIDED.
               DPBHD2      DW      336              ;CP/M SECTORS/TRACK
                           DB      5                ;BSH
                           DB      31               ;BLM
                           DB      1                ;EXM
                           DW      128Ø             ;DSM
                           DW      511              ;DRM
                           DB      ØFFH             ;ALØ
```

```
                DB      ØFFH            ;AL1
                DW      Ø               ;CKS
                DW      122             ;OFF
                DB      33H             ;16*((#CPM SECTORS/PHYSICAL SECTOR) -1) +
                                        ;LOG2(#BYTES PER SECTOR/128) + 1 +
                                        ;8 IF DOUBLE SIDED.
                ENDIF
                IF      M20 NE Ø
DPBHD1          DW      672             ;CP/M SECTORS/TRACK
                DB      5               ;BSH
                DB      31              ;BLM
                DB      1               ;EXM
                DW      2Ø15            ;DSM
                DW      511             ;DRM
                DB      ØFFH            ;ALØ
                DB      ØFFH            ;AL1
                DW      Ø               ;CKS
                DW      1               ;OFF
                DB      33H             ;16*((#CPM SECTORS/PHYSICAL SECTOR) -1) +
                                        ;LOG2(#BYTES PER SECTOR/128) + 1 +
                                        ;8 IF DOUBLE SIDED.
DPBHD2          DW      672             ;CP/M SECTORS/TRACK
                DB      5               ;BSH
                DB      31              ;BLM
                DB      1               ;EXM
                DW      2Ø15            ;DSM
                DW      511             ;DRM
                DB      ØFFH            ;ALØ
                DB      ØFFH            ;AL1
                DW      Ø               ;CKS
                DW      98              ;OFF
                DB      33H             ;16*((#CPM SECTORS/PHYSICAL SECTOR) -1) +
                                        ;LOG2(#BYTES PER SECTOR/128) + 1 +
                                        ;8 IF DOUBLE SIDED.

DPBHD3          DW      672             ;CP/M SECTORS/TRACK
                DB      5               ;BSH
                DB      31              ;BLM
                DB      1               ;EXM
                DW      1Ø28            ;DSM
                DW      511             ;DRM
                DB      ØFFH            ;ALØ
                DB      ØFFH            ;AL1
                DW      Ø               ;CKS
                DW      195             ;OFF
                DB      33H             ;16*((#CPM SECTORS/PHYSICAL SECTOR) -1) +
                                        ;LOG2(#BYTES PER SECTOR/128) + 1 +
                                        ;8 IF DOUBLE SIDED.
                ENDIF
                ENDIF

        *****************************************************************
        *                                                               *
        * CP/M DISK PARAMETER HEADERS, UNITIALIZED.                     *
        *                                                               *
        *****************************************************************
```

```
                    HEADER   MACRO    ND,DPB
                             DW       0                      ;TRANSLATION TABLE FILLED IN LATER
                             DW       0,0,0                  ;SCRATCH
                             DW       DIRBUF                 ;DIRECTORY BUFFER
                             DW       DPB                    ;DPB FILLED IN LATER
                             DW       CSV&ND                 ;DIRECTORY CHECK VECTOR
                             DW       ALV&ND                 ;ALLOCATION VECTOR
                             ENDM

D4A5 =              DPBASE   EQU      $
0000 #              DN       SET      0
                             IF       FIRST
                             REPT     MAXHD                  ;GENERATE HARD DISK DPH'S FOLLOWED
                             HEADER   %DN,DPBHD1             ;       BY FLOPPY DPH'S
                    DN       SET      DN+1
                             HEADER   %DN,DPBHD2
                    DN       SET      DN+1
                             IF       (M26 NE 0) OR (M20 NE 0)
                             HEADER   %DN,DPBHD3
                    DN       SET      DN+1
                             ENDIF
                             ENDM
                             REPT     MAXFLOP
                             HEADER   %DN,0
                    DN       SET      DN+1
                             ENDM
                             ELSE
                             REPT     MAXFLOP                ;GENERATE FLOPPY DPH'S FOLLOWED BY
                             HEADER   %DN,0                  ;       HARD DISK DPH'S
                    DN       SET      DN+1
                             ENDM
D4A5+0000                    DW       0                      ;TRANSLATION TABLE FILLED IN LATER
D4A7+0000000000              DW       0,0,0                  ;SCRATCH
D4AD+FFD8                    DW       DIRBUF                 ;DIRECTORY BUFFER
D4AF+0000                    DW       0                      ;DPB FILLED IN LATER
D4B1+CAD9                    DW       CSV0                   ;DIRECTORY CHECK VECTOR
D4B3+7FD9                    DW       ALV0                   ;ALLOCATION VECTOR
D4B5+0000                    DW       0                      ;TRANSLATION TABLE FILLED IN LATER
D4B7+0000000000              DW       0,0,0                  ;SCRATCH
D4BD+FFD8                    DW       DIRBUF                 ;DIRECTORY BUFFER
D4BF+0000                    DW       0                      ;DPB FILLED IN LATER
D4C1+55DA                    DW       CSV1                   ;DIRECTORY CHECK VECTOR
D4C3+0ADA                    DW       ALV1                   ;ALLOCATION VECTOR
                             REPT     MAXHD
                             HEADER   %DN,DPBHD1
                    DN       SET      DN+1
                             HEADER   %DN,DPBHD2
                    DN       SET      DN+1
                             IF       (M26 NE 0) OR (M20 NE 0)
                             HEADER   %DN,DPBHD3
                    DN       SET      DN+1
                             ENDIF
                             ENDM
D4C5+0000                    DW       0                      ;TRANSLATION TABLE FILLED IN LATER
D4C7+0000000000              DW       0,0,0                  ;SCRATCH
```

```
D4CD+FFD8                   DW       DIRBUF                ;DIRECTORY BUFFER
D4CF+75D4                   DW       DPBHD1                ;DPB FILLED IN LATER
D4D1+8CDB                   DW       CSV2                  ;DIRECTORY CHECK VECTOR
D4D3+95DA                   DW       ALV2                  ;ALLOCATION VECTOR
D4D5+0000                   DW       0                     ;TRANSLATION TABLE FILLED IN LATER
D4D7+0000000000             DW       0,0,0                 ;SCRATCH
D4DD+FFD8                   DW       DIRBUF                ;DIRECTORY BUFFER
D4DF+85D4                   DW       DPBHD2                ;DPB FILLED IN LATER
D4E1+83DC                   DW       CSV3                  ;DIRECTORY CHECK VECTOR
D4E3+8CDB                   DW       ALV3                  ;ALLOCATION VECTOR
D4E5+0000                   DW       0                     ;TRANSLATION TABLE FILLED IN LATER
D4E7+0000000000             DW       0,0,0                 ;SCRATCH
D4ED+FFD8                   DW       DIRBUF                ;DIRECTORY BUFFER
D4EF+95D4                   DW       DPBHD3                ;DPB FILLED IN LATER
D4F1+7ADD                   DW       CSV4                  ;DIRECTORY CHECK VECTOR
D4F3+83DC                   DW       ALV4                  ;ALLOCATION VECTOR
                            ENDIF


                   *********************************************************************
                   *                                                                   *
                   * CBIOS RAM LOCATIONS THAT DON'T NEED INITIALIZATION.                *
                   *                                                                   *
                   *********************************************************************


D4F5 0000          CPMSEC   DW       0                     ;CP/M SECTOR #
D4F7 00            CPMDRV   DB       0                     ;CP/M DRIVE #
D4F8 00            CPMTRK   DB       0                     ;CP/M TRACK #
D4F9 0000          TRUESEC  DW       0                     ;DISK JOCKEY SECTOR THAT CONTAINS CP/M SECTOR
D4FB 00            BUFDRV   DB       0                     ;DRIVE THAT BUFFER BELONGS TO
D4FC 00            BUFTRK   DB       0                     ;TRACK THAT BUFFER BELONGS TO
D4FD 0000          BUFSEC   DW       0                     ;SECTOR THAT BUFFER BELONGS TO
D4FF =             BUFFER   EQU      $


                   *********************************************************************
                   *                                                                   *
                   * SIGNON MESSAGE OUTPUT DURING COLD BOOT.                            *
                   *                                                                   *
                   *********************************************************************


                   HEXNUM   MACRO    NUM
                            IF       (NUM/16) > 9
                            DB       (NUM/16 AND 0FH) + 'A' - 10
                            ELSE
                            DB       (NUM/16 AND 0FH) + '0'
                            ENDIF
                            IF       (NUM AND 0FH) > 9
                            DB       (NUM AND 0FH) + 'A' - 10
                            ELSE
                            DB       (NUM AND 0FH) + '0'
                            ENDIF
                            ENDM

D4FF 0D0A0A        PROMPT   DB       ACR,ALF,ALF
D502 4D6F72726F             DB       'Morrow Designs '
D511 35                     DB       '0'+MSIZE/10                    ;CP/M MEMORY SIZE
D512 36                     DB       '0'+(MSIZE MOD 10)
```

```
        D513 4B2043502F      DB      'K CP/M '                    ;CP/M VERSION NUMBER
        D51A 32              DB      CPMREV/10+'0'
        D51B 2E              DB      '.'
        D51C 32              DB      (CPMREV MOD 10)+'0'
        D51D 2C20436269      DB      ', Cbios rev '
        D529 322E            DB      REVNUM/10+'0','.'            ;CBIOS REVISION NUMBER
        D52B 38              DB      REVNUM MOD 10+'0'
                             IF      MAXHD NE 0
        D52C 2E              DB      '.'
        D52D 32              DB      MREV/10+'0'
        D52E 36              DB      MREV MOD 10+'0'
                             IF      (M10 OR M20) AND SDELAY
                             DB      'M'
                             ENDIF
                             IF      (M10 OR M20) AND NOT SDELAY
                             DB      'F'
                             ENDIF
                             ENDIF
        D52F 0D0A            DB      ACR,ALF
        D531 466F7220        DB      'For '

                             IF      MAXFLOP NE 0
        D535 6120446973      DB      'a Disk Jockey 2D @ '
                             HEXNUM  %(ORIGIN/256)
        D548+45              DB      (224/16 AND 0FH) + 'A' - 10
        D549+30              DB      (224 AND 0FH) + '0'
        D54A 30304820        DB      '00H '
                             ENDIF

                             IF      (MAXHD NE 0) AND (MAXFLOP NE 0)
        D54E 616E6420        DB      'and '
                             ENDIF

                             IF      MAXHD NE 0
                             IF      MAXHD EQ 1
        D552 616E20          DB      'an '
                             ENDIF
                             IF      MAXHD EQ 2
                             DB      'two '
                             ENDIF
                             IF      MAXHD EQ 3
                             DB      'three '
                             ENDIF
                             IF      MAXHD EQ 4
                             DB      'four '
                             ENDIF
                             IF      MREV EQ 10
                             DB      'M10 '
                             ENDIF
                             IF      MREV EQ 20
                             DB      'M20 '
                             ENDIF
                             IF      MREV EQ 26
        D555 4D323620        DB      'M26 '
                             ENDIF
        D559 6861726420      DB      'hard disk'
```

```
                         IF       MAXHD NE 1
                         DB       's'
                         ENDIF
D562 204020              DB       ' @ '
                         HEXNUM   %HDORG
D565+35                  DB       (80/16 AND 0FH) + '0'
D566+30                  DB       (80 AND 0FH) + '0'
D567 482E                DB       'H.'
                         ENDIF
D569 0D0A                DB       ACR,ALF
D56B 0D0A                DB       ACR,ALF
D56D 2020202020          DB       '          THE W6GO/K6HHD LIST'
D587 0D0A                DB       ACR,ALF
D589 2020202020          DB       '        Electronics Enterprises'
D5A5 0D0A                DB       ACR,ALF
D5A7 2020202020          DB       '          Rio Linda, California'
D5C2 0D0A                DB       ACR,ALF
D5C4 00                  DB       0


         ********************************************************************
         *                                                                  *
         * UTILITY ROUTINE TO OUTPUT THE MESSAGE POINTED AT BY H&L,          *
         * TERMINATED WITH A NULL.                                           *
         *                                                                  *
         ********************************************************************

D5C5 7E      MESSAGE MOV    A,M               ;GET A CHARACTER OF THE MESSAGE
D5C6 23              INX    H                 ;BUMP TEXT POINTER
D5C7 A7              ANA    A                 ;TEST FOR END
D5C8 C8              RZ                       ;RETURN IF DONE
D5C9 E5              PUSH   H                 ;SAVE POINTER TO TEXT
D5CA 4F              MOV    C,A               ;OUTPUT CHARACTER IN C
D5CB CD0CCD          CALL   COUT              ;OUTPUT THE CHARACTER
D5CE E1              POP    H                 ;RESTORE THE POINTER
D5CF C3C5D5          JMP    MESSAGE           ;CONTINUE UNTIL NULL REACHED


         ********************************************************************
         *                                                                  *
         * CBOOT IS THE COLD BOOT LOADER. ALL OF CP/M HAS BEEN LOADED IN     *
         * WHEN CONTROL IS PASSED HERE.                                      *
         *                                                                  *
         ********************************************************************

D5D2 310001  CBOOT   LXI    SP,TPA            ;SET UP STACK

D5D5 3EC0            MVI    A,INTIOBY
D5D7 320300          STA    IOBYTE
D5DA CD30CE          CALL   TINIT             ;INITIALIZE THE TERMINAL

D5DD 21FFD4          LXI    H,PROMPT          ;PREP FOR SENDING SIGNON MESSAGE
D5E0 CDC5D5          CALL   MESSAGE           ;SEND THE PROMPT
D5E3 AF              XRA    A                 ;SELECT DISK A
D5E4 32F7D4          STA    CPMDRV
D5E7 320400          STA    CDISK

                     IF     (MAXFLOP NE 0) AND FIRST
```

```
                             STA     FLOPFLG
                             ENDIF
D5EA 2103CD                  LXI     H,BIOS+3
D5ED 2201CD                  SHLD    BIOS+1
D5F0 C350CE                  JMP     GOCPM

D5F3                         DS      512-($-BUFFER)   ;MAXIMUM SIZE BUFFER FOR 512 BYTE SECTORS

                             IF      MAXFLOP NE 0
D6FF                         DS      512                 ;ADDITIONAL SPACE FOR FLOPPIES 1K SECTORS
                             ENDIF

                             IF      (MAXFLOP NE 0) OR (MAXHD NE 0)
D8FF           DIRBUF        DS      128                 ;DIRECTORY BUFFER
                             ENDIF

               ALLOC         MACRO   ND,AL,CS
               ALV&ND        DS      AL
               CSV&ND        DS      CS
                             ENDM

0000 #         DN            SET     0

                             IF      NOT FIRST
                             REPT    MAXFLOP
                             ALLOC   %DN,75,64
               DN            SET     DN+1
                             ENDM
D97F+          ALV0          DS      75
D9CA+          CSV0          DS      64
DA0A+          ALV1          DS      75
DA55+          CSV1          DS      64
                             REPT    MAXHD
                             IF      M26 NE 0
                             ALLOC   %DN,247,0
               DN            SET     DN+1
                             ALLOC   %DN,247,0
               DN            SET     DN+1
                             ALLOC   %DN,247,0
               DN            SET     DN+1
                             ENDIF
                             IF      M10 NE 0
                             ALLOC   %DN,159,0
               DN            SET     DN+1
                             ALLOC   %DN,161,0
               DN            SET     DN+1
                             ENDIF
                             IF      M20 NE 0
                             ALLOC   %DN,252,0
               DN            SET     DN+1
                             ALLOC   %DN,252,0
               DN            SET     DN+1
                             ALLOC   %DN,129,0
               DN            SET     DN+1
                             ENDIF
                             ENDM
```

```
DA95+          ALV2    DS      247
DB8C+          CSV2    DS      Ø
DB8C+          ALV3    DS      247
DC83+          CSV3    DS      Ø
DC83+          ALV4    DS      247
DD7A+          CSV4    DS      Ø

                       ELSE

                       REPT    MAXHD
                       IF      M26 NE Ø
                       ALLOC   %DN,247,Ø
               DN      SET     DN+1
                       ALLOC   %DN,247,Ø
               DN      SET     DN+1
                       ALLOC   %DN,247,Ø
               DN      SET     DN+1
                       ENDIF
                       IF      M1Ø NE Ø
                       ALLOC   %DN,159,Ø
               DN      SET     DN+1
                       ALLOC   %DN,161,Ø
               DN      SET     DN+1
                       ENDIF
                       IF      M2Ø NE Ø
                       ALLOC   %DN,252,Ø
               DN      SET     DN+1
                       ALLOC   %DN,252,Ø
               DN      SET     DN+1
                       ALLOC   %DN,129,Ø
               DN      SET     DN+1
                       ENDIF
                       ENDM
                       REPT    MAXFLOP
                       ALLOC   %DN,75,64
               DN      SET     DN+1
                       ENDM
                       ENDIF
DD7A                   END
```

Handwritten annotations:

DD7B

286H er 646D Bytes

DFFF
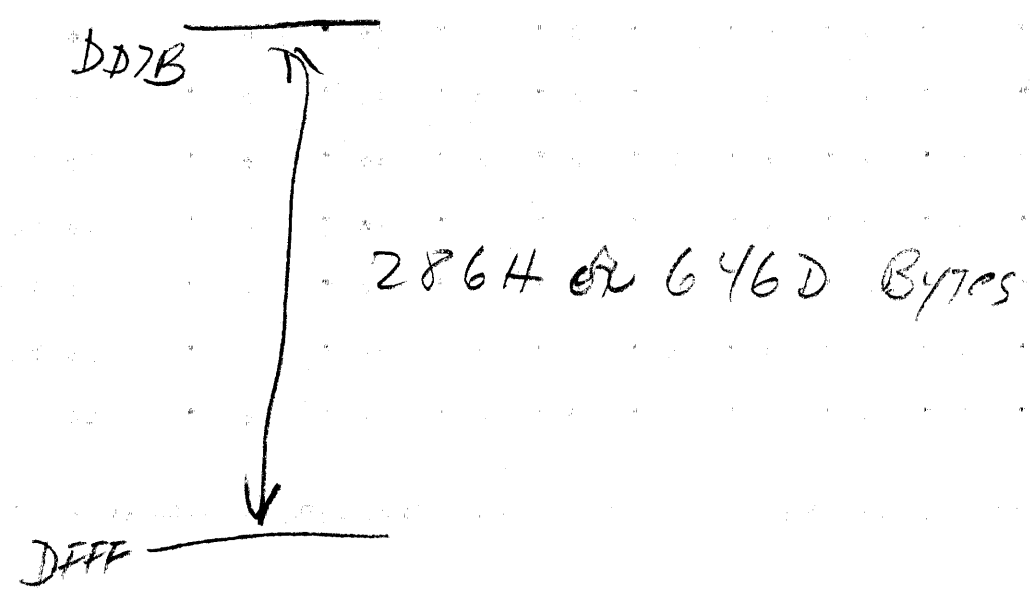
DD6D ↑
CB105(?A
293h
659d

0006 AACK	D227 ACCOK	000D ACR	0003 AETX	000A ALF
D97F ALV0	DA0A ALV1	DA95 ALV2	DB8C ALV3	DC83 ALV4
CEA3 AUTOFLG	BF00 BDOS	9000 BIAS	CD00 BIOS	D4FB BUFDRV
0080 BUFF	D4FF BUFFER	D4FD BUFSEC	D4FC BUFTRK	D105 BUFWRTN
D31B BUILD	D5D2 CBOOT	B700 CCP	0004 CDISK	CE0B CICRT
CE0B CIPTR	CD88 CITBLE	E003 CITTY	CDF6 CIUC1	CE0B CIUR1
CE0B CIUR2	CE9B CLDBOT	CE86 CLDCMND	0019 CLEAR	CDC8 COCRT
CEA4 COLDBEG	CEB2 COLDEND	CDCC COLPT	0004 COMPLT	CD42 CONIN
CD48 CONIN1	CD57 CONOUT	CD36 CONST	CDCC COPTP	CE3E COPTR
CE45 COPTR1	CD90 COTBLE	E006 COTTY	CDD7 COUL1	CDF5 COUNT
CDCC COUP1	CDCC COUP2	CD0C COUT	D0E5 CPMDMA	D4F7 CPMDRV
0016 CPMREV	D4F5 CPMSEC	D4F8 CPMTRK	CE1F CSCRT	CE1F CSPTR
CD3C CSREADR	CDB8 CSRTBLE	CDB0 CSTBLE	CE17 CSTTY	CE02 CSUC1
CE1F CSUR1	CE1F CSUR2	D9CA CSV0	DA55 CSV1	DB8C CSV2
DC83 CSV3	DD7A CSV4	CEA2 CWFLG	0008 DBLSID	D05E DCRC
D1D9 DECIDE	D1D5 DECIDGO	D207 DELAY	D20A DELOOP	D8FF DIRBUF
D0B1 DIVDONE	D060 DIVLOG	D062 DIVLOGX	D0A3 DIVLOOP	E400 DJBOOT
E003 DJCIN	E006 DJCOUT	E42D DJDEN	E412 DJDMA	CD33 DJDRV
E42A DJERR	E409 DJHOME	E400 DJRAM	E415 DJREAD	E40F DJSEC
E41B DJSEL	E430 DJSIDE	E427 DJSTAT	E40C DJTRK	E021 DJTSTAT
E418 DJWRITE	CF42 DONOP	D465 DP1024D	D425 DP1024S	D435 DPB128D
D3F5 DPB128S	D445 DPB256D	D405 DPB256S	D455 DPB512D	D415 DPB512S
D4A5 DPBASE	D475 DPBHD1	D485 DPBHD2	D495 DPBHD3	D326 DRIVES
CFF9 DRVHD	D310 DRVPTR	0020 DRVRDY	0007 DSKCLK	D0C0 DTSLOP
0005 ENTRY	D196 FILL	0000 FIRST	CFA2 FLOPOK	D104 FLUSH
D1BE FREAD	D069 GETDPB	D25A GETSPT	CE50 GOCPM	D277 HDADD
0051 HDCMND	0050 HDCNTL	0053 HDDATA	D322 HDDISK	D23A HDDMA
D1E1 HDDRV	0052 HDFUNC	D1F2 HDHOME	0050 HDORG	D2DB HDPREP
D25D HDREAD	0051 HDRESLT	0004 HDRLEN	D248 HDSEC	D300 HDSECTR
D23F HDSIDE	0020 HDSPT	0050 HDSTAT	D213 HDTRK	D221 HDTRK2
D292 HDWRITE	D31C HEAD	CF49 HOME	0000 IDBUFF	0040 INDEX
D02C INDX1	D033 INDX2	00C0 INTIOBY	D0F2 INTO	0003 IOBYTE
0008 ISBUFF	CD77 LIST	CD7A LIST1	CD82 LISTST	0003 LOGDSK
CE28 LSLPT	CDC0 LSTBLE	CD98 LTBLE	0000 M10	0000 M20
0001 M26	0002 MAXFLOP	0001 MAXHD	00F7 MDIR	D5C5 MESSAGE
D0CF MOVE	D1CA MOVER	D1CC MOVLOP	001A MREV	E800 MSDV
0038 MSIZE	CF16 NEWDMA	CEF7 NEWSEC	D125 NOADJST	CF0E NOWRAP
00FB NSTEP	00FC NULL	5A00 OFFSETC	0002 OPDONE	E000 ORIGIN
D0ED OUTOF	CD72 PNCH1	D111 PREP	D2C1 PROCESS	D4FF PROMPT
0004 PSTEP	CDA0 PTBLE	CD6C PUNCH	CDEC PWAIT	D0E8 RDWR
CD62 READER	D091 READ	CD65 READERA	CD68 READR1	CE2D READY
D095 REDWRT	000A RETRIES	0002 RETRY	D11A RETRYLP	D177 RETRYOP
001C REVNUM	0001 RSECT	CDA8 RTBLE	D27D RTLOOP	0005 SCENBL
0001 SDELAY	0200 SECLEN	D0D4 SECPSEC	D096 SECSIZ	CF50 SECTRAN
CD4C SELDEV	CF43 SETDMA	CF8B SETDRV	D041 SETDRV1	CF3D SETSEC
D208 SETTLE	CF4B SETTRK	CF66 SIDEA	CFE7 SIDEOK	CF69 SIDEONE
CF6F SIDETWO	D22B SLOOP	CE1A STAT	D1F7 STEPO	CFF7 SUBFP
D015 TDELAY	CE30 TINIT	0001 TKZERO	0008 TMOUT	0100 TPA
CF58 TRANFP	CF87 TRANHD	D4F9 TRUESEC	CEB3 WARMBEG	CEB3 WARMEND
CEF6 WARMLOD	CF2A WARMRD	CD03 WBOOTE	CEB4 WBOOT	0000 WBOT
000F WENABL	0010 WFAULT	CEB8 WFLG	000B WRESET	D08A WRITE
D0FC WRITTYP	CF2D WRMREAD	D240 WSDONE	0005 WSECT	D29E WTLOOP
D3B4 XLT124	D327 XLT128	D342 XLT256	D377 XLT512	D082 XLTS
D2FF ZKEY	D05A ZRET			