```
*****************************************************************
*                                                               *
* CBIOS FOR CP/M VER 2.2 FOR DISK JOCKEY 2D CONTROLLER (ALL     *
* REVS). HANDLES DISKETTES WITH SECTOR SIZES OF 128 BYTES       *
* SINGLE DENSITY, 256, 512, 1024 BYTES DOUBLE DENSITY.         *
*                                                               *
* WRITTEN BY BOBBY DALE GIFFORD.                                *
* 9/1/79                                                        *
*                                                               *
* CUSTOMIZED BY JAY O'BRIEN FOR USE WITH DAJEN                  *
* 8/3/80 AND MODIFIED 2/16/81 FOR MSDV DRIVER AT E800.          *
*                                                               *
* 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.                     3200H      *
*     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.                                2D00H      *
*     0     10 = CCP+80H.                            2D80H      *
*     0     12 = CCP+100H.                           2E00H      *
*     0     14 = CCP+180H.                           2E80H      *
*     0     16 = CCP+200H.                           2F00H      *
*     0     18 = CCP+280H.                           2F80H      *
*     0     20 = CCP+300H.                           3000H      *
*     0     22 = CCP+380H.                           3080H      *
*     0     24 = CCP+400H.                           3100H      *
*     0     26 = CCP+480H.                           3180H      *
*     1        = REST OF CP/M.                  3200H-4FFFH      *
*                                                               *
*****************************************************************

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

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

001E =          REVNUM  EQU     30              ;CBIOS REVISION NUMBER
0016 =          CPMREV  EQU     22              ;CP/M REVISION NUMBER
```

CBIOSDAJ.PRN
CBIOSDAJ.SYM

48K
COMPUTER #2
MSDV at E800
USE DAJEN SERIAL OUTPUT FOR LST
INPUT THRU DAJEN
2/16/81

```
                ;*****************************************************************
                ;*                                                               *
                ;* 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.                *
                ;*                                                               *
                ;*****************************************************************

E000 =          ORIGIN  EQU     0E000H
E400 =          DJRAM   EQU     ORIGIN+400H     ;DISK JOCKEY 2D RAM ADDRESS
E403 =          DJCIN   EQU     DJRAM+3H        ;DISK JOCKEY 2D CHARACTER INPUT ROUTINE
E406 =          DJCOUT  EQU     DJRAM+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
E421 =          DJTSTAT EQU     DJRAM+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

                ;*****************************************************************
                ;*                                                               *
                ;* EQUATES FOR MY SYSTEM.  J.J. O'BRIEN                          *
                ;*                                                               *
                ;*****************************************************************

D01E =          DAJIN   EQU     0D01EH          ;DAJEN INPUT ROUTINE
D286 =          DAJST   EQU     0D286H          ;DAJEN STATUS ROUTINE
D2A5 =          DAJSER  EQU     0D2A5H          ;DAJEN SERIAL OUTPUT ROUTINE
E800 =          MSDV    EQU     0E800H          ;VIDEO DRIVER FOR MSDV

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

0030 =          MSIZE   EQU     48              ;MEMORY SIZE OF TARGET CP/M
7000 =          BIAS    EQU     (MSIZE-20)*1024 ;MEMORY OFFSET FROM 20K SYSTEM
9D00 =          CCP     EQU     2D00H+BIAS      ;CONSOLE COMMAND PROCESSOR
A500 =          BDOS    EQU     CCP+800H        ;BDOS ADDRESS
B300 =          BIOS    EQU     CCP+1600H       ;CBIOS ADDRESS
0004 =          CDISK   EQU     4               ;ADDRESS OF LAST LOGGED DISK
0080 =          BUFF    EQU     80H             ;DEFAULT BUFFER ADDRESS
0100 =          TPA     EQU     100H            ;TRANSIENT MEMORY
0001 =          INTIOBY EQU     1               ;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
0003 =          AETX    EQU     3               ;A ETX CHAR
0006 =          AACK    EQU     6               ;A ACK CHAR
0019 =          CLEAR   EQU     19H             ;CLEAR SCREEN FOR DAJEN
0004 =          MAXDISK EQU     4               ;MAXIMUM # OF DISK DRIVES
0008 =          DBLSID  EQU     8               ;SIDE BIT FROM CONTROLLER

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

B300                    ORG     BIOS            ;CBIOS STARTING ADDRESS

B300 C3A0B3             JMP     CBOOT           ;COLD BOOT ENTRY POINT
B303 C3FCB3     WBOOTE  JMP     WBOOT           ;WARM BOOT ENTRY POINT
B306 C345B6             JMP     CONST           ;CONSOLE STATUS ROUTINE
B309 C351B6             JMP     CONIN           ;CONSOLE INPUT
B30C C366B6     COUT    JMP     CONOUT          ;CONSOLE OUTPUT
B30F C386B6             JMP     LIST            ;LIST DEVICE OUTPUT
B312 C37BB6             JMP     PUNCH           ;PUNCH DEVICE OUTPUT
B315 C371B6             JMP     READER          ;READER DEVICE INPUT
B318 C390B4             JMP     HOME            ;HOME DRIVE
B31B C3C6B4             JMP     SETDRV          ;SELECT DISK
B31E C392B4             JMP     SETTRK          ;SET TRACK
B321 C385B4             JMP     SETSEC          ;SET SECTOR
B324 C38AB4             JMP     SETDMA          ;SET DMA ADDRESS
B327 C369B5             JMP     READ            ;READ THE DISK
B32A C362B5             JMP     WRITE           ;WRITE THE DISK
B32D C391B6             JMP     LISTST          ;LIST DEVICE STATUS
B330 C397B4             JMP     SECTRAN         ;SECTOR TRANSLATION
B333 C31BE4     DJDRV   JMP     DJSEL           ;HOOK FOR SINGLE.COM PROGRAM

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

B336 0D0A0A     PROMPT  DB      ACR,ALF,ALF
B339 34                 DB      '0'+MSIZE/10            ;CP/M MEMORY SIZE
```

```
B33A 38                 DB      '0'+(MSIZE MOD 10)
B33B 4B2043502F         DB      'K CP/M Vers. '         ;CP/M VERSION NUMBER
B348 32                 DB      CPMREV/10+'0'
B349 2E                 DB      '.'
B34A 32                 DB      (CPMREV MOD 10)+'0'
B34B 2C20436269         DB      ', Cbios rev '
B357 332E               DB      REVNUM/10+'0','.'       ;CBIOS REVISION NUMBER
B359 30                 DB      REVNUM MOD 10+'0'
B35A 0D0A               DB      ACR,ALF
B35C 466F722054         DB      'For Thinker Toys Disk Jockey 2D Controller '
B387 402030             DB      '@ 0'

                        IF      ORIGIN/4096 > 10        ;CONTROLLER ORIGIN (HEX)
B38A 45                 DB      ORIGIN/4096+'A'-10
                        ELSE
                        DB      ORIGIN/4096+'0'
                        ENDIF

                        IF      (ORIGIN/256 AND 0FH) > 10
                        DB      (ORIGIN/256 AND 0FH)+'A'-10
                        ELSE
B38B 30                 DB      (ORIGIN/256 AND 0FH)+'0'
                        ENDIF
B38C 3030482E           DB      '00H.'
B390 0D0A00             DB      ACR,ALF,0

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

B393 7E         MESSAGE MOV     A,M             ;GET A CHARACTER OF THE MESSAGE
B394 23                 INX     H               ;BUMP TEXT POINTER
B395 A7                 ANA     A               ;TEST FOR END
B396 C8                 RZ                      ;RETURN IF DONE
B397 E5                 PUSH    H               ;SAVE POINTER TO TEXT
B398 4F                 MOV     C,A             ;OUTPUT CHARACTER IN C
B399 CD0CB3             CALL    COUT            ;OUTPUT THE CHARACTER
B39C E1                 POP     H               ;RESTORE THE POINTER
B39D C393B3             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.                                 *
                ;*                                                               *
                ;*****************************************************************

B3A0 310001     CBOOT   LXI     SP,TPA          ;SET UP STACK
B3A3 CD21B7             CALL    TINIT           ;INITIALIZE THE TERMINAL
B3A6 2136B3             LXI     H,PROMPT        ;PREP FOR SENDING SIGNON MESSAGE
B3A9 CD93B3             CALL    MESSAGE         ;SEND THE PROMPT
B3AC AF                 XRA     A               ;SELECT DISK A
B3AD 32BAB8             STA     CPMDRV
```

```
B3B0 320400             STA     CDISK

                ********************************************************************
                *                                                                  *
                * GOCPM IS THE ENTRY POINT FROM COLD BOOTS, AND WARM BOOTS. IT     *
                * INITIALIZES SOME OF THE LOCATIONS IN PAGE 0, AND SETS UP THE     *
                * INITIAL DMA ADDRESS (80H).                                       *
                *                                                                  *
                ********************************************************************

B3B3 218000     GOCPM   LXI     H,BUFF          ;SET UP INITIAL DMA ADDRESS
B3B6 CD8AB4             CALL    SETDMA
B3B9 3EC3               MVI     A,(JMP)         ;INITIALIZE JUMP TO WARM BOOT
B3BB 320000             STA     WBOT
B3BE 320500             STA     ENTRY           ;INITIALIZE JUMP TO BDOS
B3C1 2103B3             LXI     H,WBOOTE        ;ADDRESS IN WARM BOOT JUMP
B3C4 220100             SHLD    WBOT+1
B3C7 2106A5             LXI     H,BDOS+6        ;ADDRESS IN BDOS JUMP
B3CA 220600             SHLD    ENTRY+1
B3CD AF                 XRA     A               ;A <- 0
B3CE 32BFB8             STA     BUFSEC          ;DISK JOCKEY BUFFER EMPTY
B3D1 32D5B5             STA     BUFWRTN         ;SET BUFFER NOT DIRTY FLAG
B3D4 3A0400             LDA     CDISK           ;JUMP TO CP/M WITH CURRENTLY SELECTED DISK IN C
B3D7 4F                 MOV     C,A
B3D8 11FBB3             LXI     D,CMNDBEG       ;BEGINNING OF INITIAL COMMAND
B3DB 21089D             LXI     H,CCP+8         ;COMMAND BUFFER
B3DE 3E01               MVI     A,CMNDEND-CMNDBEG+1 ;LENGTH OF COMMAND
B3E0 32079D             STA     CCP+7
B3E3 47                 MOV     B,A
B3E4 CD37B6             CALL    MOVLOP
B3E7 3AF9B3             LDA     CWFLG
B3EA A7                 ANA     A
B3EB 3AFAB3             LDA     AUTOFLG
B3EE CAF2B3             JZ      CLDBOT
B3F1 1F                 RAR
B3F2 1F         CLDBOT  RAR
B3F3 DA009D             JC      CCP
B3F6 C3039D             JMP     CCP+3           ;ENTER CP/M

B3F9 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.                     *
                *                                                                  *
                ********************************************************************

B3FA 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:                                            *
                ;*                                                               *
                ;*       CMNDBEG DB      'MBASIC MYPROG'                         *
                ;*       CMNDEND DB      Ø                                       *
                ;*                                                               *
                ;* WILL EXECUTE MICROSOFT BASIC, AND MBASIC WILL EXECUTE THE     *
                ;* "MYPROG" BASIC PROGRAM.                                       *
                ;*                                                               *
                ;*****************************************************************

                CMNDBEG DB      ''              ;INITIAL COMMAND GOES HERE
B3FB ØØ         CMNDEND DB      Ø

                ;*****************************************************************
                ;*                                                               *
                ;* 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.  *
                ;*                                                               *
                ;*****************************************************************

B3FC 310001     WBOOT   LXI     SP,TPA          ;SET UP STACK POINTER
B3FF 3E01               MVI     A,1
B400 =          WFLG    EQU     $-1             ;TEST IF BEGINNING OR
B401 A7                 ANA     A               ;       ENDING A WARM BOOT
B402 3E01               MVI     A,1
B404 3200B4             STA     WFLG
B407 32F9B3             STA     CWFLG           ;SET COLD/WARM BOOT FLAG
B40A CAB3B3             JZ      GOCPM
B40D AF                 XRA     A
B40E 3200B4             STA     WFLG
B411 4F                 MOV     C,A
B412 CD33B3             CALL    DJDRV           ;SELECT DRIVE A
B415 ØE00               MVI     C,Ø             ;SELECT SINGLE DENSITY
B417 CD2DE4             CALL    DJDEN
B41A ØE00               MVI     C,Ø             ;SELECT SIDE Ø
B41C CD30E4             CALL    DJSIDE
B41F 3E0F               MVI     A,15            ;INITIALIZE THE SECTOR TO READ
B421 323FB4             STA     NEWSEC
B424 21009C             LXI     H,CCP-100H      ;AND THE DMA ADDRESS
B427 225EB4             SHLD    NEWDMA
B42A CD3EB4             CALL    WARMLOD         ;READ IN CP/M
B42D 0100A2             LXI     B,CCP+500H      ;LOAD ADDRESS FOR REST OF WARM BOOT
B430 CD12E4             CALL    DJDMA
B433 ØE08               MVI     C,8
B435 CD0FE4             CALL    DJSEC
B438 CD72B4             CALL    WARMRD
B43B C303A2             JMP     CCP+503H

B43E 3E0F       WARMLOD MVI     A,15            ;PREVIOUS SECTOR
B43F =          NEWSEC  EQU     $-1
B440 3C                 INR     A               ;UPDATE THE PREVIOUS SECTOR
```

```
B441 3C                 INR     A
B442 FE1B               CPI     27              ;WAS IT THE LAST ?
B444 DA56B4             JC      NOWRAP
B447 D609               SUI     9               ;YES
B449 FE13               CPI     19
B44B C8                 RZ
B44C 2A5EB4             LHLD    NEWDMA
B44F 1180FB             LXI     D,-480H
B452 19                 DAD     D
B453 225EB4             SHLD    NEWDMA
B456 323FB4     NOWRAP  STA     NEWSEC          ;SAVE THE NEW SECTOR TO READ
B459 4F                 MOV     C,A
B45A CD0FE4             CALL    DJSEC
B45D 21009C             LXI     H,CCP-100H      ;GET THE PREVIOUS DMA ADDRESS
B45E =          NEWDMA  EQU     $-2
B460 110001             LXI     D,100H          ;UPDATE THE DMA ADDRESS
B463 19                 DAD     D
B464 225EB4             SHLD    NEWDMA          ;SAVE THE DMA ADDRESS
B467 44                 MOV     B,H
B468 4D                 MOV     C,L
B469 CD12E4             CALL    DJDMA           ;SET THE DMA ADDRESS
B46C CD72B4             CALL    WARMRD
B46F C33EB4             JMP     WARMLOD

B472 01000A     WARMRD  LXI     B,RETRIES*100H+0;MAXIMUM # OF ERRORS
B475 C5         WRMREAD PUSH    B
B476 CD0CE4             CALL    DJTRK           ;SET THE TRACK
B479 CD15E4             CALL    DJREAD          ;READ THE SECTOR
B47C C1                 POP     B
B47D D0                 RNC                     ;CONTINUE IF SUCCESSFUL
B47E 05                 DCR     B
B47F C275B4             JNZ     WRMREAD         ;KEEP TRYING
B482 C32AE4             JMP     DJERR
```

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

B485 79         SETSEC  MOV     A,C             ;SAVE THE SECTOR NUMBER
B486 32B9B8             STA     CPMSEC          ;CP/M SECTOR #
B489 C9                 RET

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

B48A 60         SETDMA  MOV     H,B             ;HL <- BC
B48B 69                 MOV     L,C
B48C 22B5B5             SHLD    CPMDMA          ;CP/M DMA ADDRESS
B48F C9                 RET
```

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

B49Ø ØE ØØ      HOME    MVI     C,Ø             ;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.          *
                ;*                                                               *
                ;*****************************************************************

B492 79         SETTRK  MOV     A,C             ;A <- TRACK #
B493 32BBB8             STA     CPMTRK          ;CP/M TRACK #
B496 C9                 RET

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

B497 Ø3         SECTRAN INX     B
B498 D5                 PUSH    D               ;SAVE TABLE ADDRESS
B499 C5                 PUSH    B               ;SAVE SECTOR #
B49A CD41B5             CALL    GETDPB          ;GET DPB ADDRESS INTO HL
B49D 7E                 MOV     A,M             ;GET # OF CP/M SECTORS/TRACK
B49E B7                 ORA     A               ;CLEAR CARY
B49F 1F                 RAR                     ;DIVIDE BY TWO
B4AØ 91                 SUB     C
B4A1 F5                 PUSH    PSW             ;SAVE ADJUSTED SECTOR
B4A2 FAAEB4             JM      SIDETWO
B4A5 F1         SIDEA   POP     PSW             ;DISCARD ADJUSTED SECTOR
B4A6 C1                 POP     B               ;RESTORE SECTOR REQUESTED
B4A7 D1                 POP     D               ;RESTOR ADDRESS OF XLT TABLE
B4A8 EB         SIDEONE XCHG                    ;HL <- &(TRANSLATION TABLE)
B4A9 Ø9                 DAD     B               ;BC = OFFSET INTO TABLE
B4AA 6E                 MOV     L,M             ;HL <- PHYSICAL SECTOR
B4AB 26ØØ               MVI     H,Ø
B4AD C9                 RET

B4AE Ø1ØFØØ     SIDETWO LXI     B,15            ;OFFSET TO SIDE BIT
B4B1 Ø9                 DAD     B
B4B2 7E                 MOV     A,M
B4B3 E6Ø8               ANI     8               ;TEST FOR DOUBLE SIDED
B4B5 CAA5B4             JZ      SIDEA           ;MEDIA IS ONLY SINGLE SIDED
B4B8 F1                 POP     PSW             ;RETRIEVE ADJUSTED SECTOR
B4B9 C1                 POP     B
B4BA 2F                 CMA                     ;MAKE SECTOR REQUEST POSITIVE
B4BB 3C                 INR     A
B4BC 4F                 MOV     C,A             ;MAKE NEW SECTOR THE REQUESTED SECTOR
B4BD D1                 POP     D
```

```
B4BE CDA8B4             CALL    SIDEONE
B4C1 3E80               MVI     A,80H           ;SIDE TWO BIT
B4C3 B5                 ORA     L               ;       AND SECTOR
B4C4 6F                 MOV     L,A
B4C5 C9                 RET

                ;*****************************************************************
                ;*                                                               *
                ;* 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.                           *
                ;*                                                               *
                ;*****************************************************************

B4C6 79         SETDRV  MOV     A,C             ;SAVE THE DRIVE #
B4C7 32BAB8             STA     CPMDRV
B4CA FE04               CPI     MAXDISK         ;CHECK FOR A VALID DRIVE #
B4CC D23DB5             JNC     ZRET            ;ILLEGAL DRIVE #
B4CF 7B                 MOV     A,E             ;TEST IF DRIVE EVER LOGGED IN BEFORE
B4D0 E601               ANI     1
B4D2 C224B5             JNZ     SETDRV1         ;BIT 0 OF E = 0 -> NEVER SELECTED BEFORE
B4D5 3E01               MVI     A,1             ;SELECT SECTOR 1 OF TRACK 1
B4D7 32BCB8             STA     TRUESEC
B4DA 32BBB8             STA     CPMTRK
B4DD CD20B6             CALL    FILL            ;FLUSH BUFFER AND REFILL
B4E0 DA3DB5             JC      ZRET            ;TEST FOR ERROR RETURN
B4E3 CD27E4             CALL    DJSTAT          ;GET STATUS ON CURRENT DRIVE
B4E6 E60C               ANI     0CH             ;STRIP OFF UNWANTED BITS
B4E8 F5                 PUSH    PSW             ;USED TO SELECT A DPB
B4E9 1F                 RAR
B4EA 215AB5             LXI     H,XLTS          ;TABLE OF XLT ADDRESSES
B4ED 5F                 MOV     E,A
B4EE 1600               MVI     D,0
B4F0 19                 DAD     D
B4F1 E5                 PUSH    H               ;SAVE POINTER TO PROPER XLT
B4F2 CD41B5             CALL    GETDPB          ;GET DPH POINTER INTO DE
B4F5 EB                 XCHG                    ;
B4F6 D1                 POP     D
B4F7 0602               MVI     B,2             ;NUMBER OF BYTES TO MOVE
B4F9 CD37B6             CALL    MOVLOP          ;MOVE THE ADDRESS OF XLT
B4FC 110800             LXI     D,8             ;OFFSET TO DPB POINTER
B4FF 19                 DAD     D               ;HL <- &DPH.DPB
B500 E5                 PUSH    H
B501 2A07E0             LHLD    ORIGIN+7        ;GET ADDRESS OF DJ TERMINAL OUT ROUTINE
B504 23                 INX     H               ;BUMP TO LOOK AT ADDRESS OF
                                                ;       UART STATUS LOCATION
B505 7E                 MOV     A,M
B506 EE03               XRI     3               ;ADJUST FOR PROPER REV DJ
B508 6F                 MOV     L,A
B509 26E3               MVI     H,(ORIGIN+300H)/100H
```

```
B50B 7E                 MOV     A,M
B50C E608               ANI     DBLSID          ;CHECK DOUBLE SIDED BIT
B50E 11F9B7             LXI     D,DPB128S       ;BASE FOR SINGLE SIDED DPB'S
B511 C217B5             JNZ     SIDEOK
B514 1139B8             LXI     D,DPB128D       ;BASE OF DOUBLE SIDED DPB'S
B517 EB         SIDEOK  XCHG                    ;HL <- DBP BASE, DE <- &DPH.DPB
B518 D1                 POP     D               ;RESTORE DE (POINTER INTO DPH)
B519 F1                 POP     PSW             ;OFFSET TO CORRECT DPB
B51A 17                 RAL
B51B 17                 RAL
B51C 4F                 MOV     C,A
B51D 0600               MVI     B,0
B51F 09                 DAD     B
B520 EB                 XCHG                    ;PUT DPB ADDRESS IN DPH
B521 73                 MOV     M,E
B522 23                 INX     H
B523 72                 MOV     M,D
B524 CD41B5     SETDRV1 CALL    GETDPB          ;GET ADDRESS OF DPB IN HL
B527 010F00             LXI     B,15            ;OFFSET TO SECTOR SIZE
B52A 09                 DAD     B
B52B 7E                 MOV     A,M             ;GET SECTOR SIZE
B52C E607               ANI     7H
B52E 326EB5             STA     SECSIZ
B531 7E                 MOV     A,M
B532 1F                 RAR
B533 1F                 RAR
B534 1F                 RAR
B535 1F                 RAR
B536 E60F               ANI     0FH
B538 32A4B5             STA     SECPSEC
B53B EB                 XCHG                    ;HL <- DPH
B53C C9                 RET

B53D 210000     ZRET    LXI     H,0             ;SELDRV ERROR EXIT
B540 C9                 RET

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

B541 3ABAB8     GETDPB  LDA     CPMDRV          ;GET DRIVE #
B544 6F                 MOV     L,A             ;FORM OFFSET
B545 2600               MVI     H,0
B547 29                 DAD     H
B548 29                 DAD     H
B549 29                 DAD     H
B54A 29                 DAD     H
B54B 1179B8             LXI     D,DPZERO        ;BASE OF DPH'S
B54E 19                 DAD     D
B54F E5                 PUSH    H               ;SAVE ADDRESS OF DPH
B550 110A00             LXI     D,10            ;OFFSET TO DPB
B553 19                 DAD     D
B554 7E                 MOV     A,M             ;GET LOW BYTE OF DPB ADDRESS
```

```
B555 23              INX     H
B556 66              MOV     H,M             ;GET LOW BYTE OF DPB
B557 6F              MOV     L,A
B558 D1              POP     D
B559 C9              RET

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

B55A 2BB7    XLTS    DW      XLT128          ;XLT FOR 128 BYTE SECTORS
B55C 46B7            DW      XLT256          ;XLT FOR 256 BYTE SECTORS
B55E 7BB7            DW      XLT512          ;XLT FOR 512 BYTE SECTORS
B56Ø B8B7            DW      XLT124          ;XLT FOR 1Ø24 BYTE SECTORS

                     ****************************************************************
                     *                                                              *
                     * 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.       *
                     *                                                              *
                     ****************************************************************

B562 79      WRITE   MOV     A,C             ;SAVE WRITE COMMAND TYPE
B563 32CCB5          STA     WRITTYP
B566 3EØ1            MVI     A,1             ;SET WRITE COMMAND
B568 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.                                         *
                     *                                                              *
                     ****************************************************************

B569 AF      READ    XRA     A               ;SET THE COMMAND TYPE TO READ
B56A 32B8B5          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.                                                   *
                *                                                                  *
                ********************************************************************

B56D Ø6ØØ       REDWRT  MVI     B,Ø             ;THE Ø IS MODIFIED TO CONTAIN THE LOG2
B56E =          SECSIZ  EQU     $-1             ;       OF THE PHYSICAL SECTOR SIZE/128
                                                ;       ON THE CURRENTLY SELECTED DISK.
B56F 3AB9B8             LDA     CPMSEC          ;GET THE DESIRED CP/M SECTOR #
B572 F5                 PUSH    PSW             ;TEMPORARY SAVE
B573 E68Ø               ANI     8ØH             ;SAVE ONLY THE SIDE BIT
B575 4F                 MOV     C,A             ;REMEMBER THE SIDE
B576 F1                 POP     PSW             ;GET THE SECTOR BACK
B577 E67F               ANI     7FH             ;FORGET THE SIDE BIT
B579 3D                 DCR     A               ;TEMPORARY ADJUSTMENT
B57A Ø5         DIVLOOP DCR     B               ;UPDATE REPEAT COUNT
B57B CA83B5             JZ      DIVDONE
B57E B7                 ORA     A               ;CLEAR THE CARY FLAG
B57F 1F                 RAR                     ;DIVIDE THE CP/M SECTOR # BY THE SIZE
                                                ;       OF THE PHYSICAL SECTORS
B58Ø C37AB5             JMP     DIVLOOP         ;
B583 3C         DIVDONE INR     A
B584 B1                 ORA     C               ;RESTORE THE SIDE BIT
B585 32BCB8             STA     TRUESEC         ;SAVE THE PHYSICAL SECTOR NUMBER
B588 21BAB8             LXI     H,CPMDRV        ;POINTER TO DESIRED DRIVE,TRACK, AND SECTOR
B58B 11BDB8             LXI     D,BUFDRV        ;POINTER TO BUFFER DRIVE,TRACK, AND SECTOR
B58E Ø6Ø4               MVI     B,4             ;COUNT LOOP
B59Ø Ø5         DTSLOP  DCR     B               ;TEST IF DONE WITH COMPARE
B591 CA9FB5             JZ      MOVE            ;YES, MATCH. GO MOVE THE DATA
B594 1A                 LDAX    D               ;GET A BYTE TO COMPARE
B595 BE                 CMP     M               ;TEST FOR MATCH
B596 23                 INX     H               ;BUMP POINTERS TO NEXT DATA ITEM
B597 13                 INX     D
B598 CA9ØB5             JZ      DTSLOP          ;MATCH, CONTINUE TESTING

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

B59B CD2ØB6             CALL    FILL            ;FILL THE BUFFER WITH CORRECT PHYSICAL SECTOR
B59E D8                 RC                      ;NO GOOD, RETURN WITH ERROR INDICATION

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

B59F 3AB9B8     MOVE    LDA     CPMSEC          ;GET THE CP/M SECTOR TO TRANSFER
B5A2 3D                 DCR     A               ;ADJUST TO PROPER SECTOR IN BUFFER
B5A3 E6ØØ               ANI     Ø               ;STRIP OFF HIGH ORDERED BITS
```

```
B5A4 =          SECPSEC EQU     $-1             ;THE Ø IS MODIFIED TO REPRESENT THE # OF
                                                ;       CP/M SECTORS PER PHYSICAL SECTORS
B5A5 6F                 MOV     L,A             ;PUT INTO HL
B5A6 2600               MVI     H,Ø
B5A8 29                 DAD     H               ;FORM OFFSET INTO BUFFER
B5A9 29                 DAD     H
B5AA 29                 DAD     H
B5AB 29                 DAD     H
B5AC 29                 DAD     H
B5AD 29                 DAD     H
B5AE 29                 DAD     H
B5AF 11C0B8             LXI     D,BUFFER        ;BEGINNING ADDRESS OF BUFFER
B5B2 19                 DAD     D               ;FORM BEGINNING ADDRESS OF SECTOR TO TRANSFER
B5B3 EB                 XCHG                    ;DE = ADDRESS IN BUFFER
B5B4 210000             LXI     H,Ø             ;GET DMA ADDRESS, THE Ø IS MODIFIED TO
                                                ;       CONTAIN THE DMA ADDRESS
B5B5 =          CPMDMA  EQU     $-2
B5B7 3E00               MVI     A,Ø             ;THE ZERO GETS MODIFIED TO CONTAIN
                                                ;       A ZERO IF A READ, OR A 1 IF WRITE
B5B8 =          RDWR    EQU     $-1
B5B9 A7                 ANA     A               ;TEST WHICH KIND OF OPERATION
B5BA C2C2B5             JNZ     INTO            ;TRANSFER DATA INTO THE BUFFER
B5BD CD35B6     OUTOF   CALL    MOVER
B5C0 AF                 XRA     A
B5C1 C9                 RET

B5C2 EB         INTO    XCHG                    ;
B5C3 CD35B6             CALL    MOVER           ;MOVE THE DATA, HL = DESTINATION
                                                ;       DE = SOURCE
B5C6 3E01               MVI     A,1
B5C8 32D5B5             STA     BUFWRTN         ;SET BUFFER WRITTEN INTO FLAG
B5CB 3E00               MVI     A,Ø             ;CHECK FOR DIRECTORY WRITE
B5CC =          WRITTYP EQU     $-1
B5CD 3D                 DCR     A
B5CE 3E00               MVI     A,Ø
B5D0 32CCB5             STA     WRITTYP         ;SET NO DIRECTORY WRITE
B5D3 C0                 RNZ                     ;NO ERROR EXIT

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

B5D4 3E00       FLUSH   MVI     A,Ø             ;THE Ø IS MODIFIED TO REFLECT IF
                                                ;       THE BUFFER HAS BEEN WRITTEN INTO
B5D5 =          BUFWRTN EQU     $-1
B5D6 A7                 ANA     A               ;TEST IF WRITTEN INTO
B5D7 C8                 RZ                      ;NOT WRITTEN, ALL DONE
B5D8 2118E4             LXI     H,DJWRITE       ;WRITE OPERATION

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

```
                ********************************************************************
                * ADDRESS.                                                         *
                *                                                                  *
                ********************************************************************

B5DB AF         PREP    XRA     A               ;RESET BUFFER WRITTEN FLAG
B5DC 32D5B5             STA     BUFWRTN
B5DF 2212B6             SHLD    RETRYOP         ;SET UP THE READ/WRITE OPERATION
B5E2 Ø60A               MVI     B,RETRIES       ;MAXIMUM NUMBER OF RETRIES TO ATTEMPT
B5E4 C5         RETRYLP PUSH    B               ;SAVE THE RETRY COUNT
B5E5 3ABDB8             LDA     BUFDRV          ;GET DRIVE NUMBER INVOLVED IN THE OPERATION
B5E8 4F                 MOV     C,A
B5E9 CD33B3             CALL    DJDRV           ;SELECT THE DRIVE
B5EC 3ABEB8             LDA     BUFTRK
B5EF A7                 ANA     A               ;TEST FOR TRACK ZERO
B5FØ 4F                 MOV     C,A
B5F1 C5                 PUSH    B
B5F2 CCØ9E4             CZ      DJHOME          ;HOME THE DRIVE IF TRACK Ø
B5F5 C1                 POP     B               ;RESTORE TRACK #
B5F6 CDØCE4             CALL    DJTRK           ;SEEK TO PROPER TRACK
B5F9 3ABFB8             LDA     BUFSEC          ;GET SECTOR INVOLVED IN OPERATION
B5FC F5                 PUSH    PSW             ;SAVE THE SECTOR #
B5FD Ø7                 RLC                     ;BIT Ø OF A EQUALS SIDE #
B5FE E6Ø1               ANI     1               ;STRIP OFF UNNECESSARY BITS
B6ØØ 4F                 MOV     C,A             ;C <- SIDE #
B6Ø1 CD3ØE4             CALL    DJSIDE          ;SELECT THE SIDE
B6Ø4 F1                 POP     PSW             ;A <- SECTOR #
B6Ø5 E67F               ANI     7FH             ;STRIP OFF SIDE BIT
B6Ø7 4F                 MOV     C,A             ;C <- SECTOR #
B6Ø8 CDØFE4             CALL    DJSEC           ;SET THE SECTOR TO TRANSFER
B6ØB Ø1CØB8             LXI     B,BUFFER        ;SET THE DMA ADDRESS
B6ØE CD12E4             CALL    DJDMA
B611 CD15E4             CALL    DJREAD          ;THE READ OPERATION IS MODIFIED TO WRITE
B612 =          RETRYOP EQU     $-2
B614 C1                 POP     B               ;RESTORE THE RETRY COUNTER
B615 3EØØ               MVI     A,Ø             ;NO ERROR EXIT STATUS
B617 DØ                 RNC                     ;RETURN NO ERROR
B618 Ø5                 DCR     B               ;UPDATE THE RETRY COUNTER
B619 37                 STC                     ;ASSUME RETRY COUNT EXPIRED
B61A 3EFF               MVI     A,ØFFH          ;ERROR RETURN
B61C C8                 RZ
B61D C3E4B5             JMP     RETRYLP         ;TRY AGAIN

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

B62Ø CDD4B5     FILL    CALL    FLUSH           ;FLUSH BUFFER FIRST
B623 D8                 RC                      ;CHECK FOR ERROR
B624 11BAB8             LXI     D,CPMDRV        ;UPDATE THE DRIVE, TRACK, AND SECTOR
B627 21BDB8             LXI     H,BUFDRV
B62A Ø6Ø3               MVI     B,3             ;NUMBER OF BYTES TO MOVE
B62C CD37B6             CALL    MOVLOP          ;COPY THE DATA
B62F 2115E4             LXI     H,DJREAD
B632 C3DBB5             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.                                                *
                ;*                                                               *
                ;*****************************************************************

B635 Ø68Ø       MOVER   MVI     B,128           ;LENGTH OF TRANSFER
B637 1A         MOVLOP  LDAX    D               ;GET A BTE OF SOURCE
B638 77                 MOV     M,A             ;MOVE IT
B639 13                 INX     D               ;BUMP POINTERS
B63A 23                 INX     H
B63B Ø5                 DCR     B               ;UPDATE COUNTER
B63C C237B6             JNZ     MOVLOP          ;CONTINUE MOVING UNTIL DONE
B63F C9                 RET

                ;*****************************************************************
                ;*                                                               *
                ;* 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.                             *
                ;*                                                               *
                ;*****************************************************************

                ;*****************************************************************

E4Ø3 =          CITTY   EQU     DJCIN           ;INPUT FROM THE DISK JOCKEY 2D
B64Ø 79         COTTY   MOV     A,C             ;DAJSER WANTS DATA IN A
B641 F5                 PUSH    PSW             ;BECAUSE DAJEN POPS IT LATER
B642 C3A5D2             JMP     DAJSER          ;OUTPUT CHARACTER TO DAJEN

                ;*****************************************************************
                ;*                                                               *
                ;* 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.                                             *
                ;*                                                               *
                ;*****************************************************************

B645 21BFB6     CONST   LXI     H,CSTBLE        ;BEGINNING OF JUMP TABLE
B648 C357B6             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.                 *
                ;*                                                               *
                ;*****************************************************************
```

```
B64B 21C7B6    CSREADR LXI     H,CSRTBLE       ;BEGINNING OF READER STATUS TABLE
B64E C374B6            JMP     READERA

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

B651 CDD4B5    CONIN   CALL    FLUSH           ;FLUSH THE DISK BUFFER
B654 2197B6            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.
                       *

B657 3A0300    CONIN1  LDA     IOBYTE
B65A 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.
                       *

B65B E606      SELDEV  ANI     6H              ;STRIP OFF UNWANTED BITS
B65D 1600              MVI     D,Ø             ;FORM OFFSET
B65F 5F                MOV     E,A
B660 19                DAD     D               ;ADD OFFSET
B661 7E                MOV     A,M             ;PICK UP HIGH BYTE
B662 23                INX     H
B663 66                MOV     H,M             ;PICK UP LOW BYTE
B664 6F                MOV     L,A             ;FORM ADDRESS
B665 E9                PCHL                    ;GO THERE !

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

B666 C5        CONOUT  PUSH    B               ;SAVE THE CHARACTER
B667 CDD4B5            CALL    FLUSH           ;FLUSH THE DISK BUFFER
B66A C1                POP     B               ;RESTORE THE CHARACTER
B66B 219FB6            LXI     H,COTBLE        ;BEGINNING OF THE CHARACTER OUT TABLE
B66E C357B6            JMP     CONIN1          ;DO THE DECODE

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

```
B671 21B7B6    READER  LXI     H,RTBLE         ;BEGINNING OF READER INPUT TABLE
                       *
                       * ENTRY AT READERA WILL DECODE BITS 2 & 3 OF IOBYTE, USED
                       * BY CSREADER.
                       *
B674 3A0300    READERA LDA     IOBYTE
                       *
                       * ENTRY AT READER1 WILL SHIFT THE BITS INTO POSITION, USED
                       * BY LIST AND PUNCH.
                       *
B677 1F        READR1  RAR
B678 C35BB6            JMP     SELDEV
                       ***************************************************************
                       *                                                             *
                       * PUNCH: SELECT THE CORRECT PUNCH DEVICE. THE SELECTION COMES  *
                       *        FROM BITS 4&5 OF IOBYTE.                             *
                       *                                                             *
                       ***************************************************************
B67B 21AFB6    PUNCH   LXI     H,PTBLE         ;BEGINNING OF PUNCH TABLE
B67E 3A0300            LDA     IOBYTE
                       *
                       * ENTRY AT PNCH1 ROTATES BITS A LITTLE MORE IN PREP FOR
                       * SELDEV, USED BY LIST.
                       *
B681 1F        PNCH1   RAR
B682 1F                RAR
B683 C377B6            JMP     READR1
                       ***************************************************************
                       *                                                             *
                       * LIST: SELECT A LIST DEVICE BASED ON BITS 6&7 OF IOBYTE       *
                       *                                                             *
                       ***************************************************************
B686 21A7B6    LIST    LXI     H,LTBLE         ;BEGINNING OF THE LIST DEVICE ROUTINES
B689 3A0300    LIST1   LDA     IOBYTE
B68C 1F                RAR
B68D 1F                RAR
B68E C381B6            JMP     PNCH1
                       ***************************************************************
                       *                                                             *
                       * LISTST: GET THE STATUS OF THE CURRENTLY ASSIGNED LIST DEVICE  *
                       *                                                             *
                       ***************************************************************
B691 21CFB6    LISTST  LXI     H,LSTBLE        ;BEGINNING OF THE LIST DEVICE STATUS
```

```
B694 C389B6            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
                *

B697 03E4       CITBLE  DW      CITTY           ;INPUT FROM TTY (CURRENTLY ASSIGNED
                                                ;       BY INTIOBY,INPUT FROM 2D)
B699 05B7               DW      CICRT           ;INPUT FROM DAJEN
                                                ;
B69B 71B6               DW      READER          ;INPUT FROM READER (DEPENDS ON READER
                                                ;       SELECTION)
B69D 05B7               DW      CIUC1           ;INPUT FROM USER CONSOLE 1 (CURRENTLY
                                                ;       SWITCHBOARD SERIAL PORT 1)

                *
                * CONSOLE OUTPUT TABLE
                *

B69F 40B6       COTBLE  DW      COTTY           ;OUTPUT TO TTY (CURRENTLY ASSIGNED
                                                ;       BY INTIOBY,OUTPUT TO 2D)
B6A1 D7B6               DW      COCRT           ;OUTPUT TO CRT (DAJEN)
                                                ;
B6A3 86B6               DW      LIST            ;OUTPUT TO LIST DEVICE (DEPENDS ON
                                                ;       BITS 6&7 OF IOBYTE)
B6A5 DBB6               DW      COUC1           ;OUTPUT TO USER CONSOLE 1 (CURRENTLY
                                                ;       SWITCHBOARD SERIAL PORT 1)

                *
                * LIST DEVICE TABLE
                *

B6A7 40B6       LTBLE   DW      COTTY           ;OUTPUT TO TTY (CURRENTLY ASSIGNED
                                                ;       BY INTIOBY,OUTPUT TO 2D)
B6A9 D7B6               DW      COCRT           ;OUTPUT TO CRT (DAJEN)
                                                ;
B6AB DBB6               DW      COLPT           ;OUTPUT TO LINE PRINTER (CURRENTLY
                                                ;       SWITCHBOARD SERIAL PORT 1)
B6AD E6B6               DW      COUL1           ;OUTPUT TO USER LINE PRINTER 1 (CURRENTLY
                                                ;       SWITCHBOARD SERIAL PORT 1)

                *
                * PUNCH DEVICE TABLE
                *

B6AF 40B6       PTBLE   DW      COTTY           ;OUTPUT TO THE TTY (CURRENTLY ASSIGNED
                                                ;       BY INTIOBY,OUTPUT TO 2D)
```

```
B6B1 DBB6              DW      COPTP           ;OUTPUT TO PAPER TAPE PUNCH (CURRENTLY
                                               ;       SWITCHBOARD SERIAL PORT 1)
B6B3 DBB6              DW      COUP1           ;OUTPUT TO USER PUNCH 1 (CURRENTLY
                                               ;       SWITCHBOARD SERIAL PORT 1)
B6B5 DBB6              DW      COUP2           ;OUTPUT TO USER PUNCH 2 (CURRNTLLY
                                               ;       SWITCHBOARD SERIAL PORT 1)

                       *
                       * READER DEVICE INPUT TABLE
                       *

B6B7 Ø3E4      RTBLE   DW      CITTY           ;INPUT FROM TTY (CURRENTLY ASSIGNED
                                               ;       BY INTIOBY, INPUT FROM 2D)
B6B9 Ø5B7              DW      CIPTR           ;INPUT FROM PAPER TAPE READER (CURRENTLY
                                               ;       SWITCHBOARD SERIAL PORT 1)
B6BB Ø5B7              DW      CIUR1           ;INPUT FROM USER READER 1 (CURRENTLY
                                               ;       SWITCHBOARD SERIAL PORT 1)
B6BD Ø5B7              DW      CIUR2           ;INPUT FROM USER READER 2 (CURRENTLY
                                               ;       SWITCHBOARD SERIAL PORT 1)

                       *
                       * CONSOLE STATUS TABLE
                       *

B6BF Ø8B7      CSTBLE  DW      CSTTY           ;STATUS OF TTY (CURRENTLY ASSIGNED
                                               ;       BY INTIOBY, STSTUS FROM 2D)
B6C1 1ØB7              DW      CSCRT           ;STATUS FROM CRT (DAJEN)
                                               ;
B6C3 4BB6              DW      CSREADR         ;STATUS FROM READER (DEPENDS ON READER DEVICE )
B6C5 1ØB7              DW      CSUC1           ;STATUS FROM USER CONSOLE 1 (CURRENTLY
                                               ;       SWITCHBOARD SERIAL PORT 1)

                       *
                       * STATUS FROM READER DEVICE
                       *

B6C7 Ø8B7      CSRTBLE DW      CSTTY           ;STATUS FROM TTY (CURRENTLY ASSIGNED
                                               ;       BY INTIOBY, STATUS OF 2D)
B6C9 1ØB7              DW      CSPTR           ;STATUS FROM PAPER TAPE READER (CURRENTLY
                                               ;       SWITCHBOARD SERIAL PORT 1)
B6CB 1ØB7              DW      CSUR1           ;STATUS FROM USER READER 1 (CURRENTLY
                                               ;       SWITCHBOARD SERIAL PORT 1)
B6CD 1ØB7              DW      CSUR2           ;STATUS OF USER READER 2 (CURRENTLY
                                               ;       SWITCHBOARD SERIAL PORT 1)

                       *
                       * STATUS FROM LIST DEVICE
                       *

B6CF 1EB7      LSTBLE  DW      READY           ;CONSOLE ALWAYS READY
B6D1 1EB7              DW      READY           ;GET LIST STATUS
B6D3 19B7              DW      LSLPT
B6D5 19B7              DW      LSLPT

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

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

                *
                * MSDV VIDEO DRIVER
                *

B6D7 79         COCRT   MOV     A,C             ;MSDV WANTS DATA IN A
B6D8 C300E8             JMP     MSDV            ;GO THERE


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

B6DB =          COUC1   EQU     $               ;OUTPUT FROM USER CONSOLE 1
B6DB =          COPTP   EQU     $               ;OUTPUT FROM PAPER TAPE PUNCH
B6DB =          COUP1   EQU     $               ;OUTPUT FROM USER PUNCH 1
B6DB =          COUP2   EQU     $               ;OUTPUT FROM USER PUNCH 2
B6DB DB02       COLPT   IN      2               ;OUTPUT FROM LINE PRINTER,GET STATUS
B6DD E680               ANI     80H             ;WAIT UNTIL OK TO SEND
B6DF CADBB6             JZ      COLPT
B6E2 79                 MOV     A,C             ;OUTPUT THE CHARACTER
B6E3 D301               OUT     1
B6E5 C9                 RET

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

B6E6 CDDBB6     COUL1   CALL    COLPT           ;OUTPUT THE CHARACTER
B6E9 3A04B7             LDA     COUNT
B6EC 3D                 DCR     A
B6ED 3204B7             STA     COUNT
B6F0 C0                 RNZ
B6F1 3E32               MVI     A,50
B6F3 3204B7             STA     COUNT
B6F6 0E03               MVI     C,AETX
B6F8 CDDBB6             CALL    COLPT
B6FB CD05B7     PWAIT   CALL    CIPTR
B6FE FE06               CPI     AACK
B700 C2FBB6             JNZ     PWAIT
B703 C9                 RET

B704 32         COUNT   DB      50

                ********************************************************************
                *                                                                  *
                * THE FOLLOWING EQUATES SET THE INPUT FROM THE DEVICES TO COME     *
```

```
                ********************************************************************
                *  FROM THE DAJEN                                                  *
                *                                                                  *
                ********************************************************************

B7Ø5 =          CIUC1   EQU     $               ;INPUT FROM USER CONSOLE 1
B7Ø5 =          CICRT   EQU     $               ;INPUT FROM CRT
B7Ø5 =          CIUR1   EQU     $               ;INPUT FROM USER READER 1
B7Ø5 =          CIUR2   EQU     $               ;INPUT FROM USER READER 2
B7Ø5 C31ED0     CIPTR   JMP     DAJIN           ;INPUT FROM DAJEN

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

B7Ø8 CD21E4     CSTTY   CALL    DJTSTAT         ;STATUS FROM DISK JOCKEY 2D
B7ØB 3EØØ       STAT    MVI     A,Ø             ;PREP FOR ZERO RETURN
B7ØD CØ                 RNZ                     ;NOTHING FOUND
B7ØE 3D                 DCR     A               ;RETURN WITH ØFFH
B7ØF C9                 RET

                ********************************************************************
                *                                                                  *
                *  THE FOLLOWING EQUATES CAUSE THE DEVICES TO GET STATUS FROM      *
                *  THE DAJEN                                                       *
                *                                                                  *
                ********************************************************************

B71Ø =          CSUR1   EQU     $               ;STATUS OF USER READER 1
B71Ø =          CSUR2   EQU     $               ;STATUS OF USER READER 2
B71Ø =          CSPTR   EQU     $               ;STATUS OF PAPER TAPE READER
B71Ø =          CSUC1   EQU     $               ;STATUS OF USER CONSOLE 1
B71Ø CD86D2     CSCRT   CALL    DAJST           ;GET DAJEN STATUS
B713 E68Ø               ANI     8ØH             ;TEST STATUS
B715 C8                 RZ                      ;RETURN NOT READY
B716 3EFF               MVI     A,ØFFH          ;SET A
B718 C9                 RET                     ;RETURN WITH READY

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

B719 DBØ2       LSLPT   IN      2               ;ALL OTHER DEVICES WAIT
B71B E68Ø               ANI     8ØH
B71D C8                 RZ
B71E 3EFF       READY   MVI     A,ØFFH
B72Ø C9                 RET

                ********************************************************************
                *                                                                  *
                *  TINIT CAN BE MODIFIED FOR DIFFERENT I/O SETUPS.                 *
                *                                                                  *
                ********************************************************************
```

```
B721 ØE19      TINIT   MVI     C,CLEAR         ;INITIALIZE THE TERMINAL ROUTINE
B723 3EØ1              MVI     A,INTIOBY       ;INITIALIZE IOBYTE
B725 32Ø3ØØ            STA     IOBYTE
B728 C3ØCB3            JMP     COUT

                ****************************************************************
                *                                                              *
                * XLT TABLES (SECTOR SKEW TABLES) FOR CP/M 2.Ø. 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 Ø SECTORS 6 AND 8. THEY ARE      *
                * LOADED INTO MEMORY IN THE CBIOS RAM BY THE COLD BOOT ROUTINE. *
                *                                                              *
                ****************************************************************

B72B ØØ        XLT128  DB      Ø
B72C Ø107ØD1319        DB      1,7,13,19,25
B731 Ø5ØB1117          DB      5,11,17,23
B735 Ø3Ø9ØF15          DB      3,9,15,21
B739 Ø2Ø8ØE141A        DB      2,8,14,2Ø,26
B73E Ø6ØC1218          DB      6,12,18,24
B742 Ø4ØA1Ø16          DB      4,1Ø,16,22

B746 ØØ        XLT256  DB      Ø
B747 Ø1Ø2131425        DB      1,2,19,2Ø,37,38
B74D Ø3Ø4151627        DB      3,4,21,22,39,4Ø
B753 Ø5Ø6171829        DB      5,6,23,24,41,42
B759 Ø7Ø8191A2B        DB      7,8,25,26,43,44
B75F Ø9ØA1B1C2D        DB      9,1Ø,27,28,45,46
B765 ØBØC1D1E2F        DB      11,12,29,3Ø,47,48
B76B ØDØE1F2Ø31        DB      13,14,31,32,49,5Ø
B771 ØF1Ø212233        DB      15,16,33,34,51,52
B777 11122324          DB      17,18,35,36

B77B ØØ        XLT512  DB      Ø
B77C Ø1Ø2Ø3Ø411        DB      1,2,3,4,17,18,19,2Ø
B784 2122232431        DB      33,34,35,36,49,5Ø,51,52
B78C Ø5Ø6Ø7Ø815        DB      5,6,7,8,21,22,23,24
B794 2526272835        DB      37,38,39,4Ø,53,54,55,56
B79C Ø9ØAØBØC19        DB      9,1Ø,11,12,25,26,27,28
B7A4 292A2B2C39        DB      41,42,43,44,57,58,59,6Ø
B7AC ØDØEØF1Ø1D        DB      13,14,15,16,29,3Ø,31,32
B7B4 2D2E2F3Ø          DB      45,46,47,48

B7B8 ØØ        XLT124  DB      Ø
B7B9 Ø1Ø2Ø3Ø4Ø5         DB      1,2,3,4,5,6,7,8
B7C1 191A1B1C1D        DB      25,26,27,28,29,3Ø,31,32
B7C9 3132333435        DB      49,5Ø,51,52,53,54,55,56
B7D1 Ø9ØAØBØCØD        DB      9,1Ø,11,12,13,14,15,16
B7D9 2122232425        DB      33,34,35,36,37,38,39,4Ø
B7E1 393A3B3C3D        DB      57,58,59,6Ø,61,62,63,64
B7E9 1112131415        DB      17,18,19,2Ø,21,22,23,24
B7F1 292A2B2C2D        DB      41,42,43,44,45,46,47,48
```

```
                ;*****************************************************************
                ;*                                                               *
                ;* EACH OF THE FOLLOWING TABLES DESCRIBES A DISKETTE WITH THE    *
                ;* SPECIFIED CHARACTERISTICS. THE TABLES ARE CURRENTLY STORED    *
                ;* ON TRACK Ø SECTOR 13. THEY ARE READ INTO MEMORY BY THE GOCPM  *
                ;* ROUTINE IN THE CBIOS FOR CP/M VER 2.Ø.                        *
                ;*                                                               *
                ;*****************************************************************


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

B7F9 1AØØ       DPB128S DW      26              ;CP/M SECTORS/TRACK
B7FB Ø3                 DB      3               ;BSH
B7FC Ø7                 DB      7               ;BLM
B7FD ØØ                 DB      Ø               ;EXM
B7FE F2ØØ               DW      242             ;DSM
B8ØØ 3FØØ               DW      63              ;DRM
B8Ø2 CØ                 DB      ØC0H            ;ALØ
B8Ø3 ØØ                 DB      Ø               ;AL1
B8Ø4 1ØØØ               DW      16              ;CKS
B8Ø6 Ø2ØØ               DW      2               ;OFF
B8Ø8 Ø1                 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.                             *
                ;*                                                               *
                ;*****************************************************************

B8Ø9 34ØØ       DPB256S DW      52              ;CP/M SECTORS/TRACK
B8ØB Ø4                 DB      4               ;BSH
B8ØC Ø F                DB      15              ;BLM
B8ØD ØØ                 DB      Ø               ;EXM
B8ØE F2ØØ               DW      242             ;DSM
B81Ø 7FØØ               DW      127             ;DRM
B812 CØ                 DB      ØC0H            ;ALØ
B813 ØØ                 DB      Ø               ;AL1
B814 2ØØØ               DW      32              ;CKS
B816 Ø2ØØ               DW      2               ;OFF
B818 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.                             *
```

```
                ;*                                                              *
                ;****************************************************************

B819 3C00       DPB512S DW      60              ;CP/M SECTORS/TRACK
B81B 04                 DB      4               ;BSH
B81C 0F                 DB      15              ;BLM
B81D 00                 DB      0               ;EXM
B81E 1801               DW      280             ;DSM
B820 7F00               DW      127             ;DRM
B822 C0                 DB      0C0H            ;AL0
B823 00                 DB      0               ;AL1
B824 2000               DW      32              ;CKS
B826 0200               DW      2               ;OFF
B828 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.                            *
                ;*                                                              *
                ;****************************************************************

B829 4000       DP1024S DW      64              ;CP/M SECTORS/TRACK
B82B 04                 DB      4               ;BSH
B82C 0F                 DB      15              ;BLM
B82D 00                 DB      0               ;EXM
B82E 2B01               DW      299             ;DSM
B830 7F00               DW      127             ;DRM
B832 C0                 DB      0C0H            ;AL0
B833 00                 DB      0               ;AL1
B834 2000               DW      32              ;CKS
B836 0200               DW      2               ;OFF
B838 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.                            *
                ;*                                                              *
                ;****************************************************************

B839 3400       DPB128D DW      52              ;CP/M SECTORS/TRACK
B83B 04                 DB      4               ;BSH
B83C 0F                 DB      15              ;BLM
B83D 01                 DB      1               ;EXM
B83E F200               DW      242             ;DSM
B840 7F00               DW      127             ;DRM
B842 C0                 DB      0C0H            ;AL0
B843 00                 DB      0               ;AL1
B844 2000               DW      32              ;CKS
B846 0200               DW      2               ;OFF
B848 09                 DB      9H
```

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

B849 6800       DPB256D DW      104             ;CP/M SECTORS/TRACK
B84B 04                 DB      4               ;BSH
B84C 0F                 DB      15              ;BLM
B84D 00                 DB      0               ;EXM
B84E E601               DW      486             ;DSM
B850 FF00               DW      255             ;DRM
B852 F0                 DB      0F0H            ;AL0
B853 00                 DB      0               ;AL1
B854 4000               DW      64              ;CKS
B856 0200               DW      2               ;OFF
B858 1A                 DB      1AH

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

B859 7800       DPB512D DW      120             ;CP/M SECTORS/TRACK
B85B 04                 DB      4               ;BSH
B85C 0F                 DB      15              ;BLM
B85D 00                 DB      0               ;EXM
B85E 3102               DW      561             ;DSM
B860 FF00               DW      255             ;DRM
B862 F0                 DB      0F0H            ;AL0
B863 00                 DB      0               ;AL1
B864 4000               DW      64              ;CKS
B866 0200               DW      2               ;OFF
B868 3B                 DB      3BH

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

B869 8000       DP1024D DW      128             ;CP/M SECTORS/TRACK
B86B 04                 DB      4               ;BSH
B86C 0F                 DB      15              ;BLM
B86D 00                 DB      0               ;EXM
B86E 5702               DW      599             ;DSM
B870 FF00               DW      255             ;DRM
B872 F0                 DB      0F0H            ;AL0
B873 00                 DB      0               ;AL1
B874 4000               DW      64              ;CKS
B876 0200               DW      2               ;OFF
```

```
B878 7C                 DB      7CH

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

B879 0000       DPZERO  DW      0               ;ADDRESS OF TRANSLATION TABLE (FILLED
                                                ;       IN BY SETDRV)
B87B 0000000000         DW      0,0,0           ;USED BY BDOS
B881 ECBE               DW      DIRBUF          ;ADDRESS OF DIRECTORY BUFFER
B883 0000               DW      0               ;ADDRESS OF DPB (FILLED IN BY SETDRV)
B885 ECBD               DW      CSV0            ;DIRECTORY CHECK VECTOR
B887 C0BC               DW      ALV0            ;ALLOCATION VECTOR

B889 0000       DPONE   DW      0
B88B 0000000000         DW      0,0,0
B891 ECBE               DW      DIRBUF
B893 0000               DW      0
B895 2CBE               DW      CSV1
B897 0BBD               DW      ALV1

B899 0000       DPTWO   DW      0
B89B 0000000000         DW      0,0,0
B8A1 ECBE               DW      DIRBUF
B8A3 0000               DW      0
B8A5 6CBE               DW      CSV2
B8A7 56BD               DW      ALV2

B8A9 0000       DPTHRE  DW      0
B8AB 0000000000         DW      0,0,0
B8B1 ECBE               DW      DIRBUF
B8B3 0000               DW      0
B8B5 ACBE               DW      CSV3
B8B7 A1BD               DW      ALV3

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

B8B9 00         CPMSEC  DB      0               ;CP/M SECTOR #
B8BA 00         CPMDRV  DB      0               ;CP/M DRIVE #
B8BB 00         CPMTRK  DB      0               ;CP/M TRACK #
B8BC 00         TRUESEC DB      0               ;DISK JOCKEY SECTOR THAT CONTAINS CP/M SECTOR
B8BD 00         BUFDRV  DB      0               ;DRIVE THAT BUFFER BELONGS TO
B8BE 00         BUFTRK  DB      0               ;TRACK THAT BUFFER BELONGS TO
B8BF 00         BUFSEC  DB      0               ;SECTOR THAT BUFFER BELONGS TO
B8C0            BUFFER  DS      1024            ;MAXIMUM SIZE BUFFER FOR 1K SECTORS

BCC0            ALV0    DS      75              ;ALLOCATION VECTOR FOR DRIVE A
BD0B            ALV1    DS      75              ;ALLOCATION VECTOR FOR DRIVE B
BD56            ALV2    DS      75              ;ALLOCATION VECTOR FOR DRIVE C
BDA1            ALV3    DS      75              ;ALLOCATION VECTOR FOR DRIVE D
```

```
BDEC            CSV0    DS      64              ;DIRECTORY CHECK VECTOR FOR DRIVE A
BE2C            CSV1    DS      64              ;DIRECTORY CHECK VECTOR FOR DRIVE B
BE6C            CSV2    DS      64              ;DIRECTORY CHECK VECTOR FOR DRIVE C
BEAC            CSV3    DS      64              ;DIRECTORY CHECK VECTOR FOR DRIVE D
BEEC            DIRBUF  DS      128             ;DIRECTORY BUFFER

BF6C                    END
```

0006 AACK 0080 ACR 0003 AETX 000A ALF BCC0 ALV0
BD00 ALV1 BD56 ALV2 BDA1 ALV3 B3FA AUTOFLG A500 BDOS
7000 BIAS B300 BIOS B8BD BUFDRV 0080 BUFF B8C0 BUFFER
B8BF BUFSEC B8BE BUFTRK B5D5 BUFWRTN B3A0 CBOOT 9D00 CCP
0004 CDISK B705 CICRT B705 CIPTR B697 CITBLE E403 CITTY
B705 CIUC1 B705 CIUR1 B705 CIUR2 B3F2 CLDBOT 0019 CLEAR
B3FB CMNDBEG B3FB CMNDEND B6D7 COCRT B6DB COLPT B651 CONIN
B657 CONIN1 B666 CONOUT B645 CONST B6DB COPTP B69F COTBLE
B640 COTTY B6DB COUC1 B6E6 COUL1 B704 COUNT B6DB COUP1
B6DB COUP2 B30C COUT B5B5 CPMDMA B8BA CPMDRV 0016 CPMREV
B8B9 CPMSEC B8BB CPMTRK B710 CSCRT B710 CSPTR B64B CSREADR
B6C7 CSRTBLE B6BF CSTBLE B708 CSTTY B710 CSUC1 B710 CSUR1
B710 CSUR2 BDEC CSV0 BE2C CSV1 BE6C CSV2 BEAC CSV3
B3F9 CWFLG D01E DAJIN D2A5 DAJSER D286 DAJST 0008 DBLSID
BEEC DIRBUF B583 DIVDONE B57A DIVLOOP E403 DJCIN E406 DJCOUT
E42D DJDEN E412 DJDMA B333 DJDRV E42A DJERR E409 DJHOME
E400 DJRAM E415 DJREAD E40F DJSEC E41B DJSEL E430 DJSIDE
E427 DJSTAT E40C DJTRK E421 DJTSTAT E418 DJWRITE B869 DP1024D
B829 DP1024S B839 DPB128D B7F9 DPB128S B849 DPB256D B809 DPB256S
B859 DPB512D B819 DPB512S B889 DPONE B8A9 DPTHRE B899 DPTWO
B879 DPZERO B590 DTSLOP 0005 ENTRY B620 FILL B5D4 FLUSH
B541 GETDPB B3B3 GOCPM B490 HOME 0001 INTIOBY B5C2 INTO
0003 IOBYTE B686 LIST B689 LIST1 B691 LISTST B719 LSLPT
B6CF LSTBLE B6A7 LTBLE 0004 MAXDISK B393 MESSAGE B59F MOVE
B635 MOVER B637 MOVLOP E800 MSDV 0030 MSIZE B45E NEWDMA
B43F NEWSEC B456 NOWRAP E000 ORIGIN B5BD OUTOF B681 PNCH1
B5DB PREP B336 PROMPT B6AF PTBLE B67B PUNCH B6FB PWAIT
B5B8 RDWR B671 READER B569 READ B674 READERA B677 READR1
B71E READY B56D REDWRT 000A RETRIES B5E4 RETRYLP B612 RETRYOP
001E REVNUM B6B7 RTBLE B5A4 SECPSEC B56E SECSIZ B497 SECTRAN
B65B SELDEV B48A SETDMA B4C6 SETDRV B524 SETDRV1 B485 SETSEC
B492 SETTRK B4A5 SIDEA B517 SIDEOK B4A8 SIDEONE B4AE SIDETWO
B70B STAT B721 TINIT 0100 TPA B8BC TRUESEC B43E WARMLOD
B472 WARMRD B303 WBOOTE B3FC WBOOT 0000 WBOT B400 WFLG
B562 WRITE B5CC WRITTYP B475 WRMREAD B7B8 XLT124 B72B XLT128
B746 XLT256 B77B XLT512 B55A XLT8 B53D ZMET