```
;       CPM86+ drivers for ZFDC Board (Based on CPM86 V1.3 & CPM3 BIOS'es)
;               John Monahan   (monahan@vitasoft.org)

;       V1.0   6/17/2011            ;Initial Version

;       Notes:-
;       Make sure any commands/data sent to the ZFDC Board with the S100OUT routine are in CL
;       A common error is to send them in AL.

;       As you read below you (may) appreciate how simple it is to interface to the ZFDC board. You
;       just send commands/data over a port. No track re-seeking, sector re-reads, wait state activation
;       DMA's etc. The ZFDC board with its own dedicated Z80 does all that for you!

;       No effort was made to write compact efficient code. Routines are often repeated for
;       simplicity and easy reading. Any optimization (been there, done that!), is for speed rather than code size.
;       One nice thing about 16 bit processors is that allowing a few hundred bytes of "extra" code makes no overall
;       difference.

;       The Sector R\W routines are "Hard Wired" for the IBM 8" on Drives 0 & 1.
;       Drive 2 is currently set for a 5" 512 byte sectors (IBM-PC CPM86 Disk Format)
;       This can be easily changed to other formats (see below).


CR      EQU     0DH                 ;CARRIAGE RETURN
LF      EQU     0AH                 ;LINE FEED
BELL    EQU     7                   ;DING

;----------------- PORTS FOR FOR Z80/WD2793 FDC Board

S100DATAA       EQU     10H         ;IN,  S100 Data port to GET data to from FDC Board
S100DATAB       EQU     10H         ;OUT, S100 Data port to SEND data to FDC Board
S100STATUSA     EQU     11H         ;Status port for A
S100STATUSB     EQU     11H         ;Status port for B
RESETZFDCPORT   EQU     13H         ;Port to reset ZFDC Z80 CPU.

STATUSDELAY     EQU     20          ;Time-out for waiting for ZFDC Board handshake signal (Now, ~0.5 seconds @ 8MHz 8086)

ZFDCUNINITILIZED EQU        0FFH        ;If ZFDC is not yet initilized
ZFDCNOTWORKING      EQU     0FEH        ;If ZFDC is not working
ZFDCNOTPRESENT      EQU     0FDH        ;If ZFDC board is absent
ZFDCINITILIZED      EQU     000H        ;If ZFDC is initilized OK

STD8IBM             EQU     1           ;ZFDC Board Format table # for IBM 8" SDSS Disk
MINCPM          EQU     9H          ;ZFDC Board Format table # for IBM-PC CPM86 5" DDDS Disks
CPM144          EQU     16H         ;ZFDC Board Format table # for CPM86 on 3.5" DDDS Disks (My format see below)

;Commands to the ZFDC Board (Only a few are used here):-

CMDPIOTEST      EQU     0H          ;Simple loop hardware test of PIO #1 Ports
```

```
CMDMONITOR    EQU    1H          ;Jump to internal monitor here.
CMDSHOWSIGNON EQU    2H          ;This will "rotate" in the Sector Display TIL's as a hardware test
CMDRESETZFDC  EQU    3H          ;Reset the WD2793 chip and Board software

CMDSETFORMAT  EQU    4H          ;This will select a specified drive and assign a disk format table to that drive
CMDSETDRIVE   EQU    5H          ;This will select a specified drive (0,1,2,3)
CMDGETFORMAT  EQU    6H          ;Return to S100 System the current Disk paramater format table number
CMDSETTRACK   EQU    7H          ;This will set head request to a specified track
CMDSETSIDE    EQU    8H          ;This will set side request to a specified side
CMDSETSECTOR  EQU    9H          ;This will set sector request to a specified sector

CMDSETHOME    EQU    0AH         ;This will set head request to Track 0 of CURRENT drive
CMDSTEPIN     EQU    0BH         ;Step head in one track of CURRENT drive
CMDSTEPOUT    EQU    0CH         ;Step head out one track of CURRENT drive
CMDSEEKNV     EQU    0DH         ;Seek to track with NO verify of CURRENT drive

CMDSEEKTRACK  EQU    0EH         ;Seek to track to (IY+DRIVETRACK) with the track verify bit set on CURRENT drive/format
CMDGETTRACKID EQU    0FH         ;Read the CURRENT TRACK ID

CMDREADSECTOR EQU    10H         ;Read data from the CURRENT sector (on current track,side,drive).
CMDWRITESECTOR  EQU 11H          ;Write data to the CURRENT sector (on current track,side,drive).

CMDGETWDTRACK EQU    12H         ;Get the WD2793 Track register value
CMDGETWDSECTOR      EQU    13H          ;Get the WD2793 Sector register value
CMDGETWDSTATUS      EQU    14H          ;Get the WD2793 Status register value

CMDTRACKDUMP  EQU    15H         ;Dump complete CURRENT track to S-100 system
CMDFORMATTRACK      EQU    16H          ;Format the disk in the of the CURRENT drive using the current format assigned to that disk

CMDSETDEBUGON EQU    17H         ;Turn on Debug display mode
CMDSETDEBUGOFF      EQU    18H          ;Turn off Debug display mode
CMDRAMDUMP    EQU    19H         ;Command to pass back to S-100 system all memory variables and flags on ZFDC board

CMDABORT      EQU    20H         ;Generalized Abort of the current process command.
CMDHANDSHAKE  EQU    21H         ;Handshake command only sent during board initilization/testing
CMDGETDRIVE   EQU    22H         ;Get the current selected drive number (0..3)
CMDSETTRACKDS EQU    23H         ;Set Track (If a DS Disk, EVEN tracks on Side A, ODD tracks on Side B. Used by CPM)
CMDGETERRORSTRING EQU      24H          ;Return a string explaining the last Error Code sent
CMDCHECKDRIVE EQU    28H         ;Check there is a valid drive present on specified drive (0,1,2,3)

;Possible ERROR codes returned from the ZFDC Board:-
;These will be translated into ASCII strings in the error reporting function.

NO$ERRORS$FLAG      EQU    00H          ;No Errors flag for previous cmd, sent back to S-100 BIOS
BUSY$ERR      EQU    01H         ;WD2793 Timeout Error before CMD was started
HUNG$ERR      EQU    02H         ;General WD2793 Timeout Error after CMD was sent
TABLE$ERR     EQU    03H         ;Disk parameter table error
DRIVE$ERR     EQU    04H         ;Drive not 0-3
TRACK$RANGE$ERR EQU 05H          ;Drive track not valid for this disk
SECTOR$RANGE$ERR EQU       06H          ;Drive sector not valid for this disk
```

```
SIDE$ERR      EQU    07H          ;No B side on this disk
SIDE$ERR1     EQU    08H          ;Invalid Side Paramater
SECTOR$SIZE$ERR EQU 09H          ;Size of sector > 1024 Bytes

RESTORE$HUNG  EQU    0AH          ;WD2793 Timeout Error after RESTORE Command
RESTORE$ERR   EQU    0BH          ;Restore to track 0 error

STEP$IN$HUNG  EQU    0CH          ;WD2793 Timeout Error after STEP-IN Command
STEP$IN$ERR   EQU    0DH          ;Head Step In Error, DRIVE NOT READY ERROR
STEP$OUT$HUNG EQU    0EH          ;WD2793 Timeout Error after STEP-OUT Command
STEP$OUT$ERR  EQU    0FH          ;Head Step Out Error, NOT READY ERROR

SEEK$NV$HUNG  EQU    10H          ;WD2793 Timeout Error after SEEK-NV Command
SEEK$NV$ERR1  EQU    11H          ;Seek with No Verify Error, NOT READY ERROR
SEEK$NV$ERR2  EQU    12H          ;Seek with No Verify with SEEK error bit set

SEEK$TRK$HUNG EQU    13H          ;WD2793 Timeout Error after SEEK with Verify Command
SEEK$TRK$ERR1 EQU    14H          ;Seek to a track with Verify error, DRIVE NOT READY ERROR bit set
SEEK$TRK$ERR2 EQU    15H          ;Seek to a track with Verify error with SEEK ERROR bit set
SEEK$REST$HUNG       EQU    16H          ;WD2793 Timeout Error after RESTORE within SEEK with Verify Command
SEEK$REST$ERR EQU    17H          ;Restore to track 0, DRIVE NOT READY ERROR within SEEK with Verify Command

ID$ERR$HUNG   EQU    18H          ;WD2793 Timeout Error after READ TRACK ID Command
ID$ERR1              EQU    19H          ;Track ID Error, DRIVE NOT READY ERROR
ID$ERR2              EQU    1AH          ;Track ID Error, RNF ERROR
ID$ERR3              EQU    1BH          ;Track ID Error, LOST DATA ERROR
ID$ERR4              EQU    1CH          ;Track ID Error, CRC ERROR

SEC$READ$HUNG EQU    1DH          ;WD2793 Timeout Error after Read Sector Command was sent
SEC$READ$ERR1 EQU    1EH          ;Sector read error, DRIVE NOT READY ERROR
SEC$READ$ERR2 EQU    1FH          ;Sector read error, RNF ERROR
SEC$READ$ERR3 EQU    20H          ;Sector read error, LOST DATA ERROR
SEC$READ$ERR4 EQU    21H          ;Sector read error, CRC ERROR
RS$SEEK$TRK$HUNG EQU       22H          ;WD2793 Timeout Error after SEEK within READ SECTOR Command
RS$RESTORE$HUNG EQU 23H          ;WD2793 Timeout Error after RESTORE command within READ SECTOR Command
RS$RESTORE$ERR       EQU    24H          ;Restore to track 0 Error, within READ SECTOR Command
RS$SEEK$TRK$ERR1 EQU       25H          ;Seek to track error, within READ SECTOR Command
RS$SEEK$TRK$ERR2 EQU       26H          ;Seek to track error with SEEK ERROR bit set within READ SECTOR Command

SEC$WRITE$HUNG       EQU    27H          ;WD2793 Timeout Error after Read Sector Command was sent
SEC$WRITE$ERR1       EQU    28H          ;Sector write error, DRIVE NOT READY ERROR
SEC$WRITE$ERR2       EQU    29H          ;Sector write error, RNF ERROR
SEC$WRITE$ERR3       EQU    2AH          ;Sector write error, LOST DATA ERROR
SEC$WRITE$ERR4       EQU    2BH          ;Sector write error, CRC ERROR
WS$SEEK$TRK$HUNG EQU       2CH          ;WD2793 Timeout Error after SEEK within WRITE SECTOR Command
WS$RESTORE$HUNG EQU 2DH          ;WD2793 Timeout Error after RESTORE command within WRITE SECTOR Command
WS$RESTORE$ERR       EQU    2EH          ;Restore to track 0 Error, within WRITE SECTOR Command
WS$SEEK$TRK$ERR1 EQU       2FH          ;Seek to track error, within WRITE SECTOR Command
WS$SEEK$TRK$ERR2 EQU       30H          ;Seek to track error with SEEK ERROR bit set within WRITE SECTOR Command
DISK$WP$ERR   EQU    31H          ;Sector write error, Disk is write protected
```

```
CONFIRM$FORMAT      EQU    32H          ;Confirm disk format cmd request
FORMAT$HUNG   EQU    33H          ;WD2793 Timeout Error after Track Format Command was sent
FORMAT1$ERR   EQU    34H          ;Disk format request error
FORMAT2$ERR   EQU    35H          ;Track format error (Side A)
FORMAT3$ERR   EQU    36H          ;Track format error (Side B)
FORMAT4$ERR   EQU    37H          ;Restore error after formatting disk
RT$ERR$HUNG   EQU    38H          ;Disk Read Track hung error
RT$ERR        EQU    39H          ;Disk Read track error

DRIVE$INACTIVE      EQU    3AH          ;Drive is inactive
DRIVE$DOOR    EQU    3BH          ;Drive door open

ABORT$FLAG    EQU    3AH          ;Special error flag to signify the user aborted a command

ZFDC$ABSENT   EQU    3BH          ;If ZFDC Board is absent
ZFDC$INIT$ERROR     EQU    3CH          ;If ZFDC initilization error

TIMEOUT$ERROR EQU    3DH          ;Error flag to signify the previous command timed out
CMD$RANGE$ERR EQU    3EH          ;CMD out or range.',0

MAX_ERRORS    EQU    3FH          ;0 to 3FH errors only



;For Sector R/W's CPM86+ obtains ALL its Disk Bios variables using the 8086 SS:[BP] register + an offset
;Here are the crital offset values:-

iopb_mcnt     equ     byte ptr 15 ;Sector count for multi sector R/W's
iopb_drive    equ     byte ptr 14 ;Selected Drive
iopb_track    equ     word ptr 12 ;Selected track
iopb_sector   equ     word ptr 10 ;Selected Sector
iopb_dmaseg   equ     word ptr 8  ;Selected DMA Segment
iopb_dmaoff   equ     word ptr 6  ;Selected Segment offset

;However these values ONLY are valid for Sector R/W's (Not drive Login & Initilization)

       CSEG

       EXTRN  ?PMSG:NEAR, ?SMSG:NEAR
       EXTRN  ?CO:NEAR, ?CSTS:NEAR, ?CI:NEAR


;------FLOPPY DISK INITILIZATION ROUTINE---------------------------------

       ;;;;; INIT:
FL_INIT0:                     ;For the first floppy so initilize the ZFDC board
       MOV    AL,0
       MOV    Byte Ptr ZFDC_INIT_FLAG,AL ;Do not initilize yet as BIOS is still not setup for
       RET                    ;error messages etc
```

```
FL_INIT1:                       ;Next the second floppy drive, Just return
FL_INIT2:                       ;Then the third floppy drive, Just return
FL_INIT3:                       ;Then the last floppy drive, Just return
       RET

;------FLOPPY DISK LOGIN ROUTINE-----------------------------------------
;      Entry  BX  =  DPH OFFSET
;             DS  =  BIOS DATA SEGMENT
;
;      Exit   BX  =  0 IF ILLEGAL SELECT
;                 =  OFFSET OF DPH RELATIVE TO O.S. DATA SEGMENT
;
; Note Login is called only once (or again after a ^C). Normal drive selection should be done
; in the actual Sector R/W Routines.

FL_LOGIN0:                      ;Login Floppy Drive 0 (CPM B:)
       PUSH   BX                ;Save DPH
       CMP    Byte Ptr ZFDC_INIT_FLAG,0 ;Has ZFDC Board been Initilized
       JNZ    LOG0
       CALL   RESET_ZFDC        ;If Not Do So
       JZ     LOG0
       JMP    ZFDC_LOGIN_ERROR
LOG0:  MOV    CL,CMDSETDRIVE           ;Send a "Set Drive CMD" to ZFDC board
       CALL   S100OUT
       MOV    CL,0              ;Get Requested Disk in CL
       CALL   S100OUT
       CALL   WAIT_FOR_ACK      ;Return Z (or NZ with error # in [AH])
       JZ     LOG1
       JMP    ZFDC_LOGIN_ERROR

LOG1:  MOV    CL,CMDCHECKDRIVE  ;Send a "Check a Drive is valid" to ZFDC board
       CALL   S100OUT
       MOV    CL,0              ;Get Requested Disk
       CALL   S100OUT
       CALL   WAIT_FOR_ACK      ;Return Z (or NZ with error # in [AH])
       JZ     LOG2
       JMP    ZFDC_LOGIN_ERROR

LOG2:  MOV    SI,Offset SAYSIGNON0
       CALL   ?SMSG             ;Announce on speaker (If present)
       POP    BX                ;Return with valid DPH indicating all is OK
       RET

FL_LOGIN1:                      ;Login Floppy Drive 1 (CPM C:)
       PUSH   BX                ;Save DPH
       CMP    Byte Ptr ZFDC_INIT_FLAG,0 ;Has ZFDC Board been Initilized
       JNZ    LOG3
       CALL   RESET_ZFDC        ;If Not Do So
       JZ     LOG3
```

```
       JMP    ZFDC_LOGIN_ERROR
LOG3:  MOV    CL,CMDSETDRIVE            ;Send a "Set Drive CMD" to ZFDC board
       CALL   S100OUT
       MOV    CL,1                ;Get Requested Disk in CL
       CALL   S100OUT
       CALL   WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])
       JZ     LOG4
       JMP    ZFDC_LOGIN_ERROR

LOG4:  MOV    CL,CMDCHECKDRIVE    ;Send a "Check a Drive is valid" to ZFDC board
       CALL   S100OUT
       MOV    CL,1                ;Get Requested Disk
       CALL   S100OUT
       CALL   WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])
       JZ     LOG5
       JMP    ZFDC_LOGIN_ERROR

LOG5:  MOV    SI,Offset SAYSIGNON1
       CALL   ?SMSG               ;Announce on speaker (If present)
       POP    BX                  ;Return with valid DPH indicating all is OK
       RET


FL_LOGIN2:                        ;Login Floppy Drive 2 (CPM D:)
       PUSH   BX                  ;Save DPH
       CMP    Byte Ptr ZFDC_INIT_FLAG,0 ;Has ZFDC Board been Initilized
       JNZ    LOG6
       CALL   RESET_ZFDC          ;If Not Do So
       JZ     LOG6
       JMP    ZFDC_LOGIN_ERROR
LOG6:  MOV    CL,CMDSETDRIVE            ;Send a "Set Drive CMD" to ZFDC board
       CALL   S100OUT
       MOV    CL,2                ;Get Requested Disk in CL
       CALL   S100OUT
       CALL   WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])
       JZ     LOG7
       JMP    ZFDC_LOGIN_ERROR

LOG7:  MOV    CL,CMDCHECKDRIVE    ;Send a "Check a Drive is valid" to ZFDC board
       CALL   S100OUT
       MOV    CL,2                ;Get Requested Disk
       CALL   S100OUT
       CALL   WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])
       JZ     LOG8
       JMP    ZFDC_LOGIN_ERROR

LOG8:  MOV    SI,Offset SAYSIGNON2
       CALL   ?SMSG               ;Announce on speaker (If present)
       POP    BX                  ;Return with valid DPH indicating all is OK
       RET
```

```
FL_LOGIN3:                      ;Login Floppy Drive 2 (CPM H:)
       PUSH   BX                ;Save DPH
       CMP    Byte Ptr ZFDC_INIT_FLAG,0 ;Has ZFDC Board been Initilized
       JNZ    LOG9
       CALL   RESET_ZFDC        ;If Not Do So
       JZ     LOG9
       JMP    ZFDC_LOGIN_ERROR
LOG9:  MOV    CL,CMDSETDRIVE           ;Send a "Set Drive CMD" to ZFDC board
       CALL   S100OUT
       MOV    CL,3              ;Get Requested Disk in CL
       CALL   S100OUT
       CALL   WAIT_FOR_ACK      ;Return Z (or NZ with error # in [AH])
       JZ     LOG10
       JMP    ZFDC_LOGIN_ERROR

LOG10: MOV    CL,CMDCHECKDRIVE  ;Send a "Check a Drive is valid" to ZFDC board
       CALL   S100OUT
       MOV    CL,3              ;Get Requested Disk
       CALL   S100OUT
       CALL   WAIT_FOR_ACK      ;Return Z (or NZ with error # in [AH])
       JZ     LOG11
       JMP    ZFDC_LOGIN_ERROR

LOG11: MOV    SI,Offset SAYSIGNON3
       CALL   ?SMSG             ;Announce on speaker (If present)
       POP    BX                ;Return with valid DPH indicating all is OK
       RET


ZFDC_LOGIN_ERROR:               ;ZFDC returned Error Code is in AH
       PUSH   AX                ;Save error Code is in AH
       MOV    SI,Offset FLOPPYLOGINERR ;"Drive Login Error"
       CALL   ?PMSG
       POP    AX
       CALL   PRINT_ERROR_STRING  ;AH -> Error string with details
       POP    BX                ;get back BX saved above at the start
       MOV    BX,0              ;This flags CPM86+ there is a problem
       RET




;-------- ROUTINE READS N SECTORS FROM THE DISK:------------------------------
;      Entry  BX  = DPH offset
;             [BP] = BIOS disk paramater pointers
;
;      Exit:  AL  = 0 IF NO ERROR
;                 = 1 IF PHYSICAL SECTOR READ ERROR
```

```
;                   BX,DS,ES Unchanged
;-------------------------------------------------------------------------

READSECTOR0:                    ;For 8" IBM 128 Byte Sectors SSSD
       PUSH   BX
       MOV    Word Ptr SECTOR_SIZE,128 ;sector size for below
       MOV    word Ptr SEC_PER_TRACK,26 ;Sectors/track

       MOV    CL,CMDSETDRIVE            ;<<< Set Drive Drive B: ZFDC will just return if current drive
       CALL   S100OUT
       MOV    CL,0
       CALL   S100OUT
       CALL   WAIT_FOR_ACK       ;Return Z (or NZ with error # in [AH])
       JZ     COMMON_READ_SEC
       JMP    ZFDC_READ_ERROR

READSECTOR1:                    ;For 8" IBM 128 Byte Sectors SSSD
       PUSH   BX
       MOV    Word Ptr SECTOR_SIZE,128 ;sector size for below
       MOV    word Ptr SEC_PER_TRACK,26 ;;Sectors/track

       MOV    CL,CMDSETDRIVE            ;<<< Set Drive Drive C: ZFDC will just return if current drive
       CALL   S100OUT
       MOV    CL,1
       CALL   S100OUT
       CALL   WAIT_FOR_ACK       ;Return Z (or NZ with error # in [AH])
       JZ     COMMON_READ_SEC
       JMP    ZFDC_READ_ERROR

READSECTOR2:                    ;For 5" CPM-86 for IBM-PC 5" (360K SDSDS Disks)
       PUSH   BX
       MOV    Word Ptr SECTOR_SIZE,512 ;sector size for below
       MOV    word Ptr SEC_PER_TRACK,16 ;Sectors/track (8X2 sides)

       MOV    CL,CMDSETDRIVE            ;<<< Set Drive Drive D: ZFDC will just return if current drive
       CALL   S100OUT
       MOV    CL,2
       CALL   S100OUT
       CALL   WAIT_FOR_ACK       ;Return Z (or NZ with error # in [AH])
       JZ     COMMON_READ_SEC
       JMP    ZFDC_READ_ERROR

READSECTOR3:                    ;CPM-86 on 1.4M Disks, My Format (see below)
       PUSH   BX
       MOV    Word Ptr SECTOR_SIZE,512 ;sector size for below
       MOV    word Ptr SEC_PER_TRACK,36 ;Sectors/track (18X2 sides)

       MOV    CL,CMDSETDRIVE            ;<<< Set Drive Drive D: ZFDC will just return if current drive
       CALL   S100OUT
       MOV    CL,3
```

```
        CALL    S100OUT
        CALL    WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])
        JZ      COMMON_READ_SEC
        JMP     ZFDC_READ_ERROR


COMMON_READ_SEC:                    ;----------- READ DISK SECTORS ----------------------
        MOV     CL,CMDSETTRACK              ;<<< Set Track
        CALL    S100OUT
        MOV     CX,iopb_track [BP]  ;Get Requested Track (Only CL used here)
        CALL    S100OUT                     ;Send Selected track number
        CALL    WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])
        JZ      RS2
        JMP     ZFDC_READ_ERROR

RS2:    MOV     CL,CMDSETSECTOR             ;<<< Set Sector
        CALL    S100OUT
        MOV     CX,iopb_sector [BP] ;Get Requested Sector (Only CL used here)
        CMP     WORD PTR SECTOR_SIZE,128;Is it an 8" SSSD IBM disk?
        JNZ     NO_RD_SKEW
        CALL    TRANSLATE_128       ;Return in CL the translated sector # (unique the 8" IBM SSSD Disks)
NO_RD_SKEW:
        INC     CX                  ;Because sectors on disks are numbered 1,2,3...
        CALL    S100OUT                     ;Send Selected sector number to ZFDC board
        CALL    WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])
        JZ      RS3
        JMP     ZFDC_READ_ERROR

RS3:    MOV     CL,CMDSEEKTRACK             ;<<< Seek to that track (if not already there)
        CALL    S100OUT
        CALL    WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])
        JZ      RS4
        JMP     ZFDC_READ_ERROR

RS4:    MOV     CL,CMDREADSECTOR    ;<<<< Routine assumes required Drive Table,Drive,(Side),Track, and sector are already sent to board
        CALL    S100OUT
        CALL    WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])
        JZ      RS5
        JMP     ZFDC_READ_ERROR

RS5:    CLI                         ;Just in case

        PUSH    ES
        PUSH    DI
        MOV     CX,Word Ptr SECTOR_SIZE ;sector size for above (BIOS DS: still valid)

        les     DI,DWORD PTR iopb_dmaoff [BP]

RDSEC: MOV      BX,0400H            ;Put in a timeout count (Loop for status reads at most 256X4 times)
RDSEC1:         DEC    BX                   ;Dec BC
```

```
       JNZ    RDSEC2              ;Will wait 400H times before timing out
       MOV    AH,TIMEOUTERROR           ;Send Timeout error
       POP    DI                  ;Be sure to balance up stack before leaving
       POP    ES
       JMP    ZFDC_READ_ERROR

RDSEC2:       IN     AL,S100STATUSB            ;Send data to ZFDC output
       TEST   AL,80H              ;Is ZFDC in INPUT mode, if not wait
       JZ     RDSEC1
       TEST   AL,01H              ;Has previous (if any) character been read.
       JZ     RDSEC1              ;Z if not yet ready

       IN     AL,S100DATAB        ;Get data
       STOS   AL                  ;READ 1 BYTE BYTE, AL->ES:[DI++]
       LOOP   RDSEC

       POP    DI
       POP    ES                  ;GET BACK OLD VALUE OF [ES]
       STI
       CALL   WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])
       JNZ    ZFDC_READ_ERROR

NEXT_SECTOR_RD:                         ;We have done one sector, are there more
       dec    iopb_mcnt [BP]           ;Decrease multi-sector count
       jz     RD_SEC_DONE         ;Stay here until all sectors read
       mov    ax,iopb_sector [BP]
       inc    ax
       cmp    ax,Word Ptr SEC_PER_TRACK
       jb     SameRtrak           ;Still on same track
       inc    iopb_track [BP]           ;Next track
       xor    ax,ax               ;Note how sectors are numbered in CPM86+ 0,1,2,3...
SameRtrak:
       mov    iopb_sector [BP],ax
       MOV    AX,Word Ptr SECTOR_SIZE
       add    iopb_dmaoff [BP],AX ;Note how we inc. the DMA address
       jmp    COMMON_READ_SEC           ;CPM86+ says it will never cross a segment boundry


RD_SEC_DONE:
       POP    BX                  ;Balance up stack
       XOR    AL,AL
       RET                        ;Ret Z (from WAIT_FOR_ACK)


ZFDC_READ_ERROR:                   ;General read sector error reporting routine
       PUSH   AX                  ;ZFDC returned Error Code is in AH
       MOV    SI, Offset FLOPPYREADERR ;"Sector Read Error. Error Code = "
       CALL   ?PMSG
       POP    AX
       CALL   PRINT_ERROR_STRING  ;AH -> Error string
```

```
       POP    BX                  ;Balance up stack from the start above
       XOR    AL,AL
       MOV    Byte Ptr ZFDC_INIT_FLAG,AL ;Re-initilize ZFDC Board next time
       INC    AL                  ;Set to NZ & 1 for read error back to CPM86
       RET



;-------- ROUTINE WRITES N SECTORS TO THE DISK:-------------------------------
;      Entry  BX  = DPH offset
;             [BP] = BIOS disk paramater pointers
;
;      Exit:  AL  = 0 IF NO ERROR
;                 = 1 IF PHYSICAL ERROR
;                     BX,DS,ES Unchanged
;-------------------------------------------------------------------------------
WRITESECTOR0:                      ;For 8" IBM 128 Byte Sectors SSSD
       PUSH   BX
       MOV    Word Ptr SECTOR_SIZE,128 ;sector size for below
       MOV    word Ptr SEC_PER_TRACK,26 ;sectors/track

       MOV    CL,CMDSETDRIVE             ;<<< Set Drive Drive B: ZFDC will just return if current drive
       CALL   S100OUT
       MOV    CL,0
       CALL   S100OUT
       CALL   WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])
       JZ     COMMON_WRITE_SEC
       JMP    ZFDC_WRITE_ERROR

WRITESECTOR1:                      ;For 8" IBM 128 Byte Sectors SSSD
       PUSH   BX
       MOV    Word Ptr SECTOR_SIZE,128 ;sector size for below
       MOV    word Ptr SEC_PER_TRACK,26 ;sectors/track

       MOV    CL,CMDSETDRIVE             ;<<< Set Drive Drive C: ZFDC will just return if current drive
       CALL   S100OUT
       MOV    CL,1
       CALL   S100OUT
       CALL   WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])
       JZ     COMMON_WRITE_SEC
       JMP    ZFDC_WRITE_ERROR

WRITESECTOR2:                      ;For 5" CPM-86 IBM-PC 5" DSDS Disks
       PUSH   BX
       MOV    Word Ptr SECTOR_SIZE,512 ;sector size for below
       MOV    word Ptr SEC_PER_TRACK,16 ;Sectors/track (8X2 sides)

       MOV    CL,CMDSETDRIVE             ;<<< Set Drive Drive D: ZFDC will just return if current drive
       CALL   S100OUT
       MOV    CL,2
```

```
        CALL    S100OUT
        CALL    WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])
        JZ      COMMON_WRITE_SEC
        JMP     ZFDC_WRITE_ERROR

WRITESECTOR3:                       ;CPM-86 on 1.4M Disks, My Format (see below)
        PUSH    BX
        MOV     Word Ptr SECTOR_SIZE,512 ;sector size for below
        MOV     word Ptr SEC_PER_TRACK,36 ;Sectors/track (18X2 sides)

        MOV     CL,CMDSETDRIVE              ;<<< Set Drive Drive D: ZFDC will just return if current drive
        CALL    S100OUT
        MOV     CL,3
        CALL    S100OUT
        CALL    WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])
        JZ      COMMON_WRITE_SEC
        JMP     ZFDC_WRITE_ERROR



COMMON_WRITE_SEC:                   ;------------ WRITE DISK SECTORS ----------------------
        MOV     CL,CMDSETTRACK              ;<<< Set Track
        CALL    S100OUT
        MOV     CX,iopb_track [BP]  ;Get Requested Track (Only CL used here)
        CALL    S100OUT                     ;Send Selected track number
        CALL    WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])
        JZ      WS2
        JMP     ZFDC_WRITE_ERROR

WS2:    MOV     CL,CMDSETSECTOR             ;<<< Set Sector
        CALL    S100OUT
        MOV     CX,iopb_sector [BP] ;Get Requested Sector (Only CL used here)
        CMP     WORD PTR SECTOR_SIZE,128;Is it an 8" SSSD IBM disk?
        JNZ     NO_WR_SKEW
        CALL    TRANSLATE_128       ;Return in CL the translated sector # (unique the 8" IBM SSSD Disks)
NO_WR_SKEW:
        INC     CX                  ;Because sectors on disks are numbered 1,2,3...
        CALL    S100OUT                     ;Send Selected sector HEX number
        CALL    WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])
        JZ      WS3
        JMP     ZFDC_WRITE_ERROR

WS3:    MOV     CL,CMDSEEKTRACK             ;<<< Seek to that track (if not already there)
        CALL    S100OUT
        CALL    WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])
        JZ      WS4
        JMP     ZFDC_WRITE_ERROR

WS4:    MOV     CL,CMDWRITESECTOR   ;<<<< Routine assumes required Drive Table,Drive,(Side),Track, and sector are already sent to board
        CALL    S100OUT
```

```
        CALL    WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])
        JZ      WS5
        JMP     ZFDC_WRITE_ERROR

WS5:    CLI                         ;Just in case
        PUSH    DS
        PUSH    SI
        MOV     CX,Word Ptr SECTOR_SIZE ;sector size for above. >> Do this now! BEFORE we loose the BIOS DS: segment <<

        lds     SI,DWORD PTR iopb_dmaoff [BP]  ;Note from here on the BIOS:DS is not valid

WRSEC:  MOV     BX,0400H            ;Put in a timeout count (Loop for status read at most 256X4 times)
WRSEC1:         DEC     BX                  ;Dec BC
        JNZ     WRSEC2              ;Will wait 400H times before timing out
        MOV     AH,TIMEOUTERROR             ;Send Timeout error
        POP     SI                  ;Be sure and get these back before leaving
        POP     DS
        JMP     ZFDC_WRITE_ERROR

WRSEC2:         IN      AL,S100STATUSB              ;Send data to ZFDC output
        TEST    AL,80H              ;Is ZFDC in OUTPUT mode, if not wait
        JNZ     WRSEC1
        TEST    AL,02H              ;Has previous (if any) character been read.
        JZ      WRSEC1              ;Z if not yet ready

        LODS    AL                  ;WRITE 1 BYTE BYTE, DS:[SI++] -> AL
        OUT     S100DATAB,AL        ;Send it
        LOOP    WRSEC

        POP     SI
        POP     DS                  ;GET BACK OLD VALUE OF [DS]. BIOS:DS is valid again
        STI
        CALL    WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])
        JNZ     ZFDC_WRITE_ERROR

NEXT_SECTOR_WR:
        dec     iopb_mcnt [BP]              ;Decrease multi-sector count
        jz      WR_SEC_DONE         ;Stay here until all sectors read
        mov     ax,iopb_sector [BP]
        inc     ax
        cmp     ax,Word Ptr SEC_PER_TRACK ;We are OK here the BIOS DS: is back
        jb      SameWtrak           ;Still on same track
        inc     iopb_track [BP]
        xor     ax,ax               ;Note how sectors are numbered in CPM86+ 0,1,2,3...
SameWtrak:
        mov     iopb_sector [BP],ax
        MOV     AX,Word Ptr SECTOR_SIZE
        add     iopb_dmaoff [BP],AX ;Note how we inc. the DMA address
        jmp     COMMON_WRITE_SEC    ;CPM86+ says it will never cross a segment boundry
```

```
WR_SEC_DONE:
       POP    BX                  ;Balance up stack
       xor    al,al
       ret

ZFDC_WRITE_ERROR:
       CMP    AH,DISKWPERR        ;Special case for Write Protected Disk error
       JZ     DISK_WP_ERROR
       MOV    SI, Offset FLOPPYWRITEERR ;"Sector Write Error Error Code = "
       CALL   ?PMSG
       CALL   PRINT_ERROR_STRING  ;AH -> Error string
       POP    BX                  ;Balance up stack
       XOR    AL,AL
       MOV    Byte Ptr ZFDC_INIT_FLAG,AL ;Re-initilize ZFDC Board next time
       INC    AL                  ;Set to NZ & 1 for read error
       RET

DISK_WP_ERROR:
       MOV    SI, Offset DISKWPERROR
       CALL   ?PMSG
       POP    BX                  ;Balance up stack
       XOR    AL,AL
       INC    AL                  ;Set to NZ & 1 for read error
       RET



;<<<<<<<<<<<<<<<<<<<< ZFDC CORE ROUTINES >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
RESET_ZFDC:
       OUT    RESETZFDCPORT,AL    ;Do a hardware reset. Does not matter what is in [AL]

       MOV    AX,5                ;~0.5 second at 10 MHz
       MOV    CX,0                ;Delay to allow board to setup hardware
WAITD: LOOP   WAITD               ;Delay for ~0.5 seconds
       DEC    AX
       JNZ     WAITD

       IN     AL,S100DATAB        ;Check the board is there
       CMP    AL,CMDHANDSHAKE            ;Make sure we get HANDSHAKE byte back
       MOV    AH,ZFDCABSENT       ;If not then no ZFDC board present
       JNZ    BADZFDC                    ;If not there, just abort

       MOV    AL,CMDHANDSHAKE            ;Send another byte just to be sure.
       OUT    S100DATAB,AL        ;This clears up ints on ZFDC board
       CALL   WAIT_FOR_ACK        ;Return Z (or NZ with error # in [AH])

       OR     AL,AL
       MOV    AH,ZFDCINITERROR    ;If not then no ZFDC board present
       JNZ    BADZFDC                    ;just abort
```

```
                              ;Set Disk Formats. Do in reverse order so we end up with Drive 0 selected
        MOV     CL,CMDSETFORMAT         ;Send Set Disk Format to Drive CMD
        CALL    S100OUT
        MOV     CL,3                    ;Floppy Drive 2, (ZFDC Board expects a 0H, 1H, 2H or 3H)
        CALL    S100OUT
        MOV     CL,CPM144               ;CPM-86 on 1.4M Disks, My Format (see below),See ZFDC Board Code for more info
        CALL    S100OUT
        CALL    WAIT_FOR_ACK            ;Return Z (or NZ with error # in [AH])
        JNZ     BADZFDC

        MOV     CL,CMDSETFORMAT         ;Send Set Disk Format to Drive CMD
        CALL    S100OUT
        MOV     CL,2                    ;Floppy Drive 2, (ZFDC Board expects a 0H, 1H, 2H or 3H)
        CALL    S100OUT
        MOV     CL,MINCPM               ;Set 5" Drive to 5" IBM-PC CPM86 Format (See ZFDC Board Code for more info)
        CALL    S100OUT
        CALL    WAIT_FOR_ACK            ;Return Z (or NZ with error # in [AH])
        JNZ     BADZFDC

        MOV     CL,CMDSETFORMAT         ;Send Set Disk Format to Drive CMD
        CALL    S100OUT
        MOV     CL,1                    ;Floppy Drive 1, (ZFDC Board expects a 0H, 1H, 2H or 3H)
        CALL    S100OUT
        MOV     CL,STD8IBM              ;ZFDC Board expects a Disk Format Table Number (0,1,2...13H)
        CALL    S100OUT
        CALL    WAIT_FOR_ACK            ;Return Z (or NZ with error # in [AH])
        JNZ     BADZFDC

        MOV     CL,CMDSETFORMAT         ;Send Set Disk Format to Drive CMD
        CALL    S100OUT
        MOV     CL,0                    ;Floppy Drive 0, (ZFDC Board expects a 0H, 1H, 2H or 3H)
        CALL    S100OUT
        MOV     CL,STD8IBM              ;ZFDC Board expects a Disk Format Table Number (0,1,2...13H)
        CALL    S100OUT
        CALL    WAIT_FOR_ACK            ;Return Z (or NZ with error # in [AH])
        JNZ     BADZFDC
        MOV     AL,0FFH                 ;Flag to indicate ZFDC board is setup OK
        MOV     Byte Ptr ZFDC_INIT_FLAG,AL
        MOV     SI,Offset SAYINITILIZED ;Announce on speaker
        CALL    ?SMSG
        XOR     Al,AL
        RET                             ;Return Z

BADZFDC:
        MOV     AL,0H                   ;Flag to indicate ZFDC board is NOT OK
        MOV     Byte Ptr ZFDC_INIT_FLAG,AL
        MOV     SI,Offset SAYINITFAIL   ;Announce on speaker
        CALL    ?SMSG
        XOR     AL,AL
        DEC     AL
```

```
      RET                       ;Return NZ WITH ERROR IN AH


 TRANSLATE_128:                         ;Sector translation for IBM SSSD 8" Drives
      MOV    SI,Offset IBMSKEW
      ADD    SI,CX
      MOV    CL, Byte Ptr [SI]
      RET


PRINT_ERROR_STRING:             ;Translate error code in AH to a string (in DS:SI)
      MOV    BX,Offset ZFDCERRORS       ;Point to DW table of string pointers
      MOV    AL,AH              ;Note the strings take up quite a bit of space. If you
      CMP    AL,MAX_ERRORS      ;are low on RAM just return the code # or use the
      JBE    POINTER_OK         ;ZFDC boards 24H CMD. However if the board "locks up"
                                ;that command will not work!
      MOV    AL,CMD$RANGE$ERR   ;Send out of range warning MSG
POINTER_OK:
      SHL    AL,1               ;X2 for table
      MOV    AH,0
      ADD    BX,AX              ;Offset into table
      MOV    SI, Word Ptr [BX]  ;Get start of string pointer from lookup table
      CALL   ?PMSG
      RET


S100STAT:                       ;Check if ZFDC has any data for S-100 system
      IN     AL,S100STATUSB
      TEST   AL,01H             ;Anything there ?
      JZ     S100ST1                  ;Return 0 if nothing
      XOR    AL,AL
      DEC    AL                 ;Return NZ, & 0FFH in AL if something there
S100ST1:RET


S100IN:       IN     AL,S100STATUSB            ;Check if ZFDC has any data for S-100 system
      TEST   AL,80H             ;Is ZFDC in input mode, if not, wait
      JZ     S100IN             ;If low then ZFDC board is still in input mode, wait
      TEST   AL,01H
      JZ     S100IN
      IN     AL,S100DATAA       ;return with character in AL
      RET

S100OUT:IN    AL,S100STATUSB            ;Send data to ZFDC output (arrive with character to be sent in C)
      TEST   AL,80H             ;Is ZFDC in output mode, if not wait
      JNZ    S100OUT
      TEST   AL,02H             ;Has previous (if any) character been read.
      JZ      S100OUT                 ;Z if not yet ready
      MOV    AL,CL
      OUT    S100DATAB,AL
```

```
       RET

WAIT_FOR_ACK:                   ;Delay to wait for ZFDC to return data. There is a timeout of about 2 sec.
       PUSH   BX                ;This can be increased if you are displaying debugging info on the ZFDC
       PUSH   DX                ;HEX LED display.
       MOV    BX,0
       MOV    DL,STATUSDELAY          ;Timeout, (about 2 seconds)
WAIT1: IN     AL,S100STATUSB          ;Check if ZFDC has any data for S-100 system
       TEST   AL,80H           ;Is ZFDC in input mode
       JZ     WAIT2            ;if low then ZFDC is still in input mode
       CALL   S100STAT         ;Wait until ZFDC Board sends something
       JZ     WAIT2
       CALL   S100IN           ;Get returned Error # (Note this releases the SENDDATA routine on the ZFDC board)
       MOV    AH,AL            ;<<< Store Error Code (if any) in AH
       CMP    AL,NOERRORSFLAG          ;Was SENDOK/NOERRORSFLAG sent back from ZFDC Board
       POP    DX               ;Balance up stack
       POP    BX
       RET                     ;Return NZ if problem, Z if no problem
WAIT2: DEC    BH
       JNZ    WAIT1            ;Try for ~2 seconds
       DEC    BH
       DEC    BL
       JNZ    WAIT1
       DEC    BH
       DEC    BL
       DEC    DL
       JNZ    WAIT1
       XOR    AL,AL
       DEC    AL
       MOV    AH,3FH           ;Flag as local Time out error
       POP    DX               ;Balance up stack
       POP    BX
       RET                     ;Return NZ flag set if timeout & 0FFH in [AL]
                               ;Error code in AH


;      AX_HEXOUT               ;Output the 4 hex digits in [AX] (For debugging etc. )
AX_hexout:
       PUSH   AX
       MOV    AL,AH
       CALL   AL_hexout
       POP    AX
       CALL   AL_hexout
       RET


;      AL_HEXOUT               ;Output the 2 hex digits in [AL]
AL_hexout:                     ;No registers altered (except AL)
       push   cx
       push   ax
```

```
       mov    cl,4                ;first isolate low nibble
       shr    al,cl
       call   hexdigout
       pop    ax
       call   hexdigout           ;get upper nibble
       pop    cx
       ret

hexdigout:                        ;Convert nibble to ascii
       and    al,0fh
       add    al,90h
       daa
       adc    al,40h
       daa
       mov    cl,al
       call   ?CO
       ret

;      SIMPLE SEND CRLF
CRLF:  push   cx
       push   bx
       mov    cl,cr
       call   ?CO
       mov    cl,lf
       call   ?CO
       pop    bx
       pop    cx
       ret




;************************************************************************
;*                                                                      *
;*     Disk parameter header definition:                                *
;************************************************************************

       DSEG

       Public @DPHB,@DPHC,@DPHD, @DPHH

;------------------------- DATA STORAGE AREA --------------------------------


;------      Disk Parameter Header Equates --------------------------------
;
;      +------+------+------+------+------+------+------+------+
;  00h |     XLT     |                    | DOPEN|             |
;      +------+------+------+------+------+------+------+------+
;  08h        |     DPB     |     CSV     |     ALV     |    DIRBCB   |
```

```
;       +------+------+------+------+------+------+------+------+
;   10h |    DATBCB   |    HSHTBL   |     INIT    |    LOGIN    |
;       +------+------+------+------+------+------+------+------+
;   18h |     READ    |    WRITE    | UNIT | CHNNL|NFLAGS|
;       +------+------+------+------+------+------+------+
;
;       Use two different tables (in case we wish to modify later for a HD or something)

@DPHB           EQU     OFFSET $        ;8" SSSD IBM Format Drive B:
                DW      SD128trans      ;TRANSLATE TABLE
                DB      0,0,0H          ;SCRATCH AREA
                DB      0H              ;DOOR OPEN FLAG
                DB      0,0H            ;SCRATCH AREA
                DW      DPB0            ;DISK PARAMETER TABLE FOR FLOPPY B:
                DW      0FFFFH,0FFFFH   ;CHECKSUM, ALLOCATION VECTOR
                DW      0FFFFH          ;DIRECTORY BCB
                DW      0FFFFH          ;DATA BCB
                DW      0FFFFH          ;HASH TABLE
                DW      FL_INIT0
                DW      FL_LOGIN0
                DW      READSECTOR0
                DW      WRITESECTOR0
                DB      0               ;UNIT 0 ON THIS CONTROLLER
                DB      0               ;ALWAYS CHANNEL 0
                DB      0               ;ZERO FLAGS AT THE MOMENT

@DPHC           EQU     OFFSET $        ;8" SSSD IBM Format Drive C:
                DW      SD128trans      ;TRANSLATE TABLE
                DB      0,0,0H          ;SCRATCH AREA
                DB      0H              ;DOOR OPEN FLAG
                DB      0,0H            ;SCRATCH AREA
                DW      DPB1            ;DISK PARAMETER TABLE FOR FLOPPY C:
                DW      0FFFFH,0FFFFH   ;CHECKSUM, ALLOCATION VECTOR
                DW      0FFFFH          ;DIRECTORY BCB
                DW      0FFFFH          ;DATA BCB
                DW      0FFFFH          ;HASH TABLE
                DW      FL_INIT1
                DW      FL_LOGIN1
                DW      READSECTOR1
                DW      WRITESECTOR1
                DB      0               ;UNIT 0 ON THIS CONTROLLER
                DB      0               ;ALWAYS CHANNEL 0
                DB      0               ;ZERO FLAGS AT THE MOMENT

@DPHD           EQU     OFFSET $        ;5" CPM-86 360K Format Drive D:
                DW      0000H           ;TRANSLATE TABLE (NONE 5" & 3" Drives)
                DB      0,0,0H          ;SCRATCH AREA
                DB      0H              ;DOOR OPEN FLAG
                DB      0,0H            ;SCRATCH AREA
                DW      DPB2            ;DISK PARAMETER TABLE FOR 360K FLOPPY C:
```

```
            DW      0FFFFH,0FFFFH;CHECKSUM, ALLOCATION VECTOR
            DW      0FFFFH        ;DIRECTORY BCB
            DW      0FFFFH        ;DATA BCB
            DW      0FFFFH        ;HASH TABLE
            DW      FL_INIT2
            DW      FL_LOGIN2
            DW      READSECTOR2
            DW      WRITESECTOR2
            DB      0             ;UNIT 0 ON THIS CONTROLLER
            DB      0             ;ALWAYS CHANNEL 0
            DB      0             ;ZERO FLAGS AT THE MOMENT

@DPHH       EQU     OFFSET $      ;3.5" 1.44H (My Format) Drive H:
            DW      0000H         ;TRANSLATE TABLE (NONE 5" & 3" Drives)
            DB      0,0,0H        ;SCRATCH AREA
            DB      0H            ;DOOR OPEN FLAG
            DB      0,0H          ;SCRATCH AREA
            DW      DPB3          ;DISK PARAMETER TABLE FOR 1.4M FLOPPY H:
            DW      0FFFFH,0FFFFH;CHECKSUM, ALLOCATION VECTOR
            DW      0FFFFH        ;DIRECTORY BCB
            DW      0FFFFH        ;DATA BCB
            DW      0FFFFH        ;HASH TABLE
            DW      FL_INIT3
            DW      FL_LOGIN3
            DW      READSECTOR3
            DW      WRITESECTOR3
            DB      0             ;UNIT 0 ON THIS CONTROLLER
            DB      0             ;ALWAYS CHANNEL 0
            DB      0             ;ZERO FLAGS AT THE MOMENT


;CPM+ equivalent DPB's
;           DISKDEF      512,61,256,2048,1024,1,8000H     ;Both IDE drives
;           DISKDEF      512,16,40,2048,64,1        ;5" Disks 360K (16=8X2, because 1 Track, but both sides)
;           DISKDEF      512,36,80,2048,256,1             ;3" Disks 1.4M (36=18X2, because 1 Track, but both sides)
;            DISKDEF 0,0,63,0,2048,3996,1024,0,2    ;Memory disk

DPB0        EQU     OFFSET $      ;DISK PARAMETER BLOCK DRIVE 0
            DW      26            ;PHYSICAL sec/track  (Note different from CPM3)
            DB      03h,07h             ;block shift and mask
            DB      0H            ;extent mask
            DW      0f2h          ;maximum block number
            DW      3fh           ;maximum dir. number
            DB      0c0h,0h             ;alloc
            DW      10h           ;check size
            DW      2h            ;track offset
            DB      0,0           ;physical sector size shift

DPB1        EQU     OFFSET $      ;DISK PARAMETER BLOCK DRIVE 1
            DW      26            ;PHYSICAL sec/track  (Note different from CPM3)
```

```
            DB      03h,07h             ;block shift and mask
            DB      0H              ;extent mask
            DW      0f2h            ;maximum block number
            DW      3fh             ;maximum dir. number
            DB      0c0h,0h             ;alloc
            DW      10h             ;check size
            DW      2h              ;track offset
            DB      0,0             ;physical sector size shift

                                    ;This is a double sided disk. So 8 sectors on both sides
                                    ;the ZFDC switches sides for sectors 9,10,11...
                                    ;but only 40 tracks
DPB2        EQU     OFFSET $        ;DISK PARAMETER BLOCK DRIVE 2
            DW      16              ;PHYSICAL sec/track  (Note different from CPM3)
            DB      04h,0fh             ;block shift and mask
            DB      01H             ;extent mask
            DW      09BH            ;maximum block number
            DW      03FH            ;maximum dir. number
            DB      080h,00h        ;alloc
            DW      10h             ;check size
            DW      1H              ;track offset
            DB      02,03           ;physical sector size shift

                                    ;This is a 1.4M  double sided disk. So 18 sectors on both sides
                                    ;the ZFDC switches sides for sectors 19,20,21...
                                    ;and 80 tracks
DPB3        EQU     OFFSET $        ;DISK PARAMETER BLOCK DRIVE 3
            DW      36              ;PHYSICAL sec/track  (Note different from CPM3)
            DB      04h,0fh             ;block shift and mask
            DB      0H              ;extent mask
            DW      02C6H           ;maximum block number
            DW      0FFH            ;maximum dir. number
            DB      0F0h,00h        ;alloc
            DW      40h             ;check size
            DW      1H              ;track offset
            DB      02,03           ;physical sector size shift


SD128trans  EQU     OFFSET $        ;TRANSLATE TABLE FOR IBM 8" DISK
IBMSKEW             DB      0,6,12,18       ;Note this table is unusual in that it
            DB      24,4,10,16      ;starts with sector 0-25. We increase
            DB      22,2,8,14       ;the sector number in the actual R/W routine
            DB      20,1,7,13       ;Typically these XLT tables are 1-26
            DB      19,25,5,11
            DB      17,23,3,9
            DB      15,21

ZFDC_INIT_FLAG      DB      0H              ;Zero if ZFDC board is not yet initilized
SECTOR_SIZE DW      0H              ;Bytes per sector for the current drive
SEC_PER_TRACK DW    0H              ;Sectore/Track  (26 for 8" and 16 for 5")
```

```
;      ---MESSAGES---

SAYSIGNON0    DB     'ibm format  128 byte sectors 8 inch disk on drive B',0           ;For speech synthesis
SAYSIGNON1    DB     'ibm format  128 byte sectors 8 inch disk on drive C',0           ;For speech synthesis
SAYSIGNON2    DB     'cpm-86 format  512 byte sectors 5 inch disk on drive D',0  ;For speech synthesis
SAYSIGNON3    DB     'cpm-86 format  512 byte sectors 3.5 inch disk on drive H',0      ;For speech synthesis
SAYINITILIZED DB     'ZFDC Board Initilized OK!',0                               ;For speech synthesis
SAYINITFAIL   DB     'ZFDC Board Initilization Failed',0                         ;For speech synthesis

FLOPPYLOGINERR      DB     CR,LF,BELL,'Floppy Disk Drive Login Error.',0

FLOPPYWRITEERR      DB     CR,LF,'ZFDC Board Sector Write Error.',0
FLOPPYREADERR DB     CR,LF,'ZFDC Board Sector Read Error.',0
DISKWPERROR   DB     CR,LF,'This Disk is Write Protected.',0


;Table OF ZFDC Returned Error Codes (Do not change order):-

ZFDCERRORS    DW      PT_NES_FLAG        ;00H,  No Errors flag for previous cmd, sent back to S-100 BIOS
              DW       PT_BUSY_ERR              ;01H, WD2793 Timeout Error Before CMD was started
              DW       PT_HUNG_ERR              ;02H, General WD2793 Timeout Error After CMD was sent
              DW       PT_TABLE_ERR             ;03H, Disk parameter table error
              DW       PT_DRIVE_ERR             ;04H, Drive not 0-3
              DW       PT_TRACK_RANGE_ERR     ;05H,   Drive track not valid for this disk
              DW       PT_SECTOR_RANGE_ERR    ;06H,   Drive sector not valid for this disk
              DW       PT_SIDE_ERR              ;07H, No B side on this disk
              DW       PT_SIDE_ERR1             ;08H, Invalid Side Paramater
              DW       PT_SECTOR_SIZE_ERR     ;09H,   Size of sector > 1024 Bytes

              DW       PT_RESTORE_HUNG   ;0AH,  WD2793 Timeout Error after RESTORE Command
              DW       PT_RESTORE_ERR           ;0BH, Restore to track 0 Error

              DW       PT_STEPIN_HUNG          ;0CH,  WD2793 Timeout Error after STEP-IN Command
              DW       PT_STEPIN_ERR           ;0DH, Head Step In Error, DRIVE NOT READY ERROR
              DW       PT_STEPOUT_HUNG   ;0EH,  WD2793 Timeout Error after STEP-OUT Command
              DW       PT_STEPOUT_ERR          ;0FH,  Head Step Out Error, NOT READY ERROR

              DW       PT_SEEK_NV_HUNG   ;10H,  WD2793 Timeout Error after SEEK-NV Command
              DW       PT_SEEK_NV_ERR1   ;11H,  Seek with No Verify Error, NOT READY ERROR
              DW       PT_SEEK_NV_ERR2   ;12H,  Seek with No Verify with SEEK error bit set

              DW       PT_SEEK_TRK_HUNG  ;13H,  WD2793 Timeout Error after SEEK with Verify Command
              DW       PT_SEEK_TRK_ERR1  ;14H,  Seek to track in [B'] with Verify error, DRIVE NOT READY ERROR bit set
              DW       PT_SEEK_TRK_ERR2  ;15H,  Seek to track in [B'] with Verify error with SEEK ERROR bit set
              DW       PT_SEEK_REST_HUNG ;16H,  WD2793 Timeout Error after RESTORE within SEEK with Verify Command
              DW       PT_SEEK_REST_ERR  ;17H,  Restore to track 0, DRIVE NOT READY ERROR within SEEK with Verify Command
```

```
DW       PT_ID_ERR_HUNG            ;18H, WD2793 Timeout Error after READ TRACK ID Command
DW       PT_ID_ERR1        ;19H,  Track ID Error, DRIVE NOT READY ERROR
DW       PT_ID_ERR2        ;1AH,  Track ID Error, RNF ERROR
DW       PT_ID_ERR3        ;1BH,  Track ID Error, LOST DATA ERROR
DW       PT_ID_ERR4        ;1CH,  Track ID Error, CRC ERROR

DW       PT_RS_HUNG        ;1DH,  WD2793 Timeout Error after Read Sector Command was sent
DW       PT_RS_ERR1        ;1EH,  Sector read error, DRIVE NOT READY ERROR
DW       PT_RS_ERR2        ;1FH,  Sector read error, RNF ERROR
DW       PT_RS_ERR3        ;20H,  Sector read error, LOST DATA ERROR
DW       PT_RS_ERR4        ;21H,  Sector read error, CRC ERROR
DW       PT_RS_SK_TRK_HUNG ;22H,  WD2793 Timeout Error after SEEK within READ SECTOR Command
DW       PT_RS_RES_HUNG            ;23H,  WD2793 Timeout Error after RESTORE command within READ SECTOR Command
DW       PT_RS_RES_ERR             ;24H,  Restore to track 0, DRIVE NOT READY ERROR within READ SECTOR Command
DW       PT_RS_SKTRK_ERR1  ;25H,  Seek to track error, DRIVE NOT READY ERROR bit set within READ SECTOR Command
DW       PT_RS_SKTRK_ERR2  ;26H,  Seek to track error with SEEK ERROR bit set within READ SECTOR Command

DW       PT_WS_HUNG        ;27H,  WD2793 Timeout Error after Read Sector Command was sent
DW       PT_WS_ERR1        ;28H,  Sector write error, DRIVE NOT READY ERROR
DW       PT_WS_ERR2        ;29H,  Sector write error, RNF ERROR
DW       PT_WS_ERR3        ;2AH,  Sector write error, LOST DATA ERROR
DW       PT_WS_ERR4        ;2BH,  Sector write error, CRC ERROR
DW       PT_WS_SK_TRK_HUNG ;2CH,  WD2793 Timeout Error after SEEK within WRITE SECTOR Command
DW       PT_WS_RES_HUNG            ;2DH,  WD2793 Timeout Error after RESTORE command within WRITE SECTOR Command
DW       PT_WS_RES_ERR             ;2EH,  Restore to track 0, DRIVE NOT READY ERROR within WRITE SECTOR Command
DW       PT_WS_SKTRK_ERR1  ;2FH,  Seek to track error, DRIVE NOT READY ERROR bit set within WRITE SECTOR Command
DW       PT_WS_SKTRK_ERR2  ;30H,  Seek to track error with SEEK ERROR bit set within WRITE SECTOR Command
DW       PT_DISK_WP_ERR            ;31H, Sector write error, Disk is write protected

DW       PT_CONFIRM_FORMAT ;32H,  Confirm disk format cmd request
DW       PT_FORMAT_HUNG            ;33H, WD2793 Timeout Error after Track Format Command was sent
DW       PT_FORMAT1_ERR            ;34H, Disk format request error
DW       PT_FORMAT2_ERR            ;35H, Track format error (Side A)
DW       PT_FORMAT3_ERR            ;36H, Track format error (Side B)
DW       PT_FORMAT4_ERR            ;37H, Restore error after formatting disk

DW       PT_RT_ERR_HUNG            ;38H, Disk Read Track hung error
DW      PT_RT_ERR              ;39H     Disk Read track error

DW      PT_DRIVE_INACTIVE  ;3AH    Drive is inactive
DW       PT_DRIVE_DOOR          ;3BH    Drive door open
DW      PT_BUFFER_OVERFLOW ;3CH    Too many sectors were requested in a multi sector R/W

DW      PT_ABORT_FLAG             ;3DH   Special error flag to signify the user aborted a command
DW       PT_CMD_RANGE_ERR  ;3EH,  CMD out or range
DW       TIMEOUT_ERROR_MSG ;3FH,  ZFDC Timeout Error


PT_NES_FLAG         DB     CR,LF,      'Error = 00H',CR,LF,'No Errors flag sent back to S-100 BIOS.',0
```

```
PT_BUSY_ERR          DB    CR,LF,BELL, 'Error = 01H',CR,LF,'WD2793 Timeout Error.',0
PT_HUNG_ERR          DB    CR,LF,BELL, 'Error = 02H',CR,LF,'General WD2793 Timeout Error.',0
PT_TABLE_ERR         DB    CR,LF,BELL, 'CMD=04H,  Error = 03H',CR,LF,'Disk parameter table error.',0
PT_DRIVE_ERR         DB    CR,LF,BELL, 'CMD=05H,  Error = 04H',CR,LF,'Invalid Drive. (Must be 0,1,2,or 3)',0
PT_TRACK_RANGE_ERR  DB     CR,LF,BELL, 'CMD=07H,  Error = 05H',CR,LF,'Drive track not valid for this disk.',0
PT_SECTOR_RANGE_ERR DB     CR,LF,BELL, 'CMD=06H,  Error = 06H',CR,LF,'Drive sector not valid for this disk.',0
PT_SIDE_ERR          DB    CR,LF,BELL, 'CMD=08H,  Error = 07H',CR,LF,'No B side on this disk.',0
PT_SIDE_ERR1         DB    CR,LF,BELL, 'CMD=08H,  Error = 08H',CR,LF,'Invalid Side Parameter.',0
PT_SECTOR_SIZE_ERR  DB     CR,LF,BELL, 'CMD=09H,  Error = 09H',CR,LF,'Size of sector > 1024 Bytes.',0

PT_RESTORE_HUNG           DB     CR,LF,BELL, 'CMD=0AH,  Error = 0AH',CR,LF,'WD2793 Timeout Error after RESTORE Command.',0
PT_RESTORE_ERR             DB    CR,LF,BELL, 'CMD=0AH,  Error = 0BH',CR,LF,'Restore to track 0 Error.',0
PT_STEPIN_HUNG            DB     CR,LF,BELL, 'CMD=0BH,  Error = 0CH',CR,LF,'WD2793 Timeout Error after STEP-IN Command.',0
PT_STEPIN_ERR       DB    CR,LF,BELL, 'CMD=0BH  Error = 0DH',CR,LF,'Head Step-In Error, DRIVE NOT READY. ',0

PT_STEPOUT_HUNG           DB     CR,LF,BELL, 'CMD=0CH,  Error = 0EH',CR,LF,'WD2793 Timeout Error after STEP-OUT Command.',0
PT_STEPOUT_ERR            DB     CR,LF,BELL, 'CMD=0CH,  Error = 0FH',CR,LF,'Head Step Out Error, NOT READY ERROR.',0

PT_SEEK_NV_HUNG           DB     CR,LF,BELL, 'CMD=0DH,  Error = 10H',CR,LF,'WD2793 Timeout Error after SEEK-NV Command.',0
PT_SEEK_NV_ERR1           DB     CR,LF,BELL, 'CMD=0DH,  Error = 11H',CR,LF,'Seek (with No Verify) Error, '
                  DB     'DRIVE NOT READY ERROR.',0
PT_SEEK_NV_ERR2           DB      CR,LF,BELL, 'CMD=0DH,  Error = 12H',CR,LF,'Seek (with No Verify), SEEK error bit set.',0

PT_SEEK_TRK_HUNG    DB     CR,LF,BELL, 'CMD=0EH,  Error = 13H',CR,LF,'WD2793 Timeout Error after SEEK '
                    DB     'with Verify Command.',0
PT_SEEK_TRK_ERR1    DB      CR,LF,BELL, 'CMD=0EH,  Error = 14H',CR,LF,'Seek to track (with Verify) error, '
                    DB     'NOT READY bit set.',0
PT_SEEK_TRK_ERR2    DB     CR,LF,BELL, 'CMD=0EH,  Error = 15H',CR,LF,'Seek to track (with Verify) error '
                    DB     'with SEEK ERROR bit set.',0
PT_SEEK_REST_HUNG   DB     CR,LF,BELL, 'CMD=0EH,  Error = 16H',CR,LF,'WD2793 Timeout Error after RESTORE'
                    DB     'within a SEEK (With Verify) Command.',0
PT_SEEK_REST_ERR    DB      CR,LF,BELL, 'CMD=0EH,  Error = 17H',CR,LF,'Restore to track 0, DRIVE NOT '
                    DB     'READY ERROR within a SEEK (With Verify) Command.',0

PT_ID_ERR_HUNG             DB      CR,LF,BELL, 'CMD=0FH,  Error = 18H',CR,LF,'WD2793 Timeout Error after '
                  DB     'READ TRACK ID Command.',0
PT_ID_ERR1          DB      CR,LF,BELL, 'CMD=0FH,  Error = 19H',CR,LF,'Track ID Error, DRIVE NOT READY ERROR.',0
PT_ID_ERR2          DB      CR,LF,BELL, 'CMD=0FH,  Error = 1AH',CR,LF,'Track ID Error, RNF ERROR.',0
PT_ID_ERR3          DB      CR,LF,BELL, 'CMD=0FH,  Error = 1BH',CR,LF,'Track ID Error, LOST DATA ERROR.',0

PT_ID_ERR4          DB      CR,LF,BELL, 'CMD=0FH,  Error = 1CH',CR,LF,'Track ID Error, CRC ERROR.',0

PT_RS_HUNG          DB     CR,LF,BELL, 'CMD=10H,  Error = 1DH',CR,LF,'WD2793 Timeout Error after '
                    DB     'READ SECTOR Command.',0
PT_RS_ERR1          DB     CR,LF,BELL, 'CMD=10H,  Error = 1EH',CR,LF,'READ SECTOR error, DRIVE NOT READY ERROR.',0
PT_RS_ERR2          DB      CR,LF,BELL, 'CMD=10H,  Error = 1FH',CR,LF,'READ SECTOR error, RNF ERROR.',0
PT_RS_ERR3          DB      CR,LF,BELL, 'CMD=10H,  Error = 20H',CR,LF,'READ SECTOR error, LOST DATA ERROR.',0
PT_RS_ERR4          DB      CR,LF,BELL, 'CMD=10H,  Error = 21H',CR,LF,'READ SECTOR error, CRC ERROR.',0
PT_RS_SK_TRK_HUNG   DB     CR,LF,BELL, 'CMD=10H,  Error = 22H',CR,LF,'WD2793 Timeout Error after SEEK ',0
                    DB     'within a READ SECTOR Command.',0
```

```
PT_RS_RES_HUNG          DB     CR,LF,BELL, 'CMD=10H,  Error = 23H',CR,LF,'WD2793 Timeout Error after a RESTORE command '
                DB      'within a READ SECTOR Command.',0
PT_RS_RES_ERR   DB       CR,LF,BELL, 'CMD=10H,  Error = 24H',CR,LF,'Restore to Track 0, DRIVE NOT '
                DB      'READY ERROR within a READ SECTOR Command.',0
PT_RS_SKTRK_ERR1 DB      CR,LF,BELL, 'CMD=10H,  Error = 25H',CR,LF,'Seek to Track error, DRIVE NOT '
                DB      'READY ERROR bit set within a READ SECTOR Command.',0
PT_RS_SKTRK_ERR2 DB      CR,LF,BELL, 'CMD=10H,  Error = 26H',CR,LF,'Seek to Track error with SEEK ERROR '
                DB      'bit set within a READ SECTOR Command.',0

PT_WS_HUNG      DB       CR,LF,BELL, 'CMD=11H,  Error = 27H',CR,LF,'WD2793 Timeout Error '
                DB      'after WRITE SECTOR Command.',0
PT_WS_ERR1      DB       CR,LF,BELL, 'CMD=11H,  Error = 28H',CR,LF,'WRITE SECTOR error, DRIVE NOT READY ERROR.',0
PT_WS_ERR2      DB       CR,LF,BELL, 'CMD=11H,  Error = 29H',CR,LF,'WRITE SECTOR error, RNF ERROR.',0
PT_WS_ERR3      DB       CR,LF,BELL, 'CMD=11H,  Error = 2AH',CR,LF,'WRITE SECTOR error, LOST DATA ERROR.',0
PT_WS_ERR4      DB       CR,LF,BELL, 'CMD=11H,  Error = 2BH',CR,LF,'WRITE SECTOR error, CRC ERROR.',0
PT_WS_SK_TRK_HUNG DB     CR,LF,BELL, 'CMD=11H,  Error = 2CH',CR,LF,'WD2793 Timeout Error after SEEK '
                DB      'within WRITE SECTOR Command.',0
PT_WS_RES_HUNG          DB      CR,LF,BELL, 'CMD=11H,  Error = 2DH',CR,LF,'WD2793 Timeout Error after '
                DB      'RESTOR command within a WRITE SECTOR Command.',0
PT_WS_RES_ERR   DB       CR,LF,BELL, 'CMD=11H,  Error = 2EH',CR,LF,'Restore to track 0, DRIVE NOT '
                DB      'READY ERROR within a WRITE SECTOR Command.',0
PT_WS_SKTRK_ERR1 DB      CR,LF,BELL, 'CMD=11H,  Error = 2FH',CR,LF,'Seek to track error, DRIVE NOT '
                DB      'READY ERROR bit set within a WRITE SECTOR Command.',0
PT_WS_SKTRK_ERR2 DB      CR,LF,BELL, 'CMD=11H,  Error = 30H',CR,LF,'Seek to track error with SEEK ERROR '
                DB      'bit set within a WRITE SECTOR Command.',0
PT_DISK_WP_ERR           DB      CR,LF,BELL, 'CMD=11H,  Error = 31H',CR,LF,'WRITE SECTOR error, '
                DB      'Disk is write protected.',0

PT_CONFIRM_FORMAT DB     CR,LF,BELL, 'CMD=16H,  Error = 32H',CR,LF,'Confirm disk format command request.',0
PT_FORMAT_HUNG           DB      CR,LF,BELL, 'CMD=16H,  Error = 33H',CR,LF,'WD2793 Timeout Error after Track '
                DB      'Format Command.',0
PT_FORMAT1_ERR           DB      CR,LF,BELL, 'CMD=16H,  Error = 34H',CR,LF,'Disk format request error.',0
PT_FORMAT2_ERR           DB      CR,LF,BELL, 'CMD=16H,  Error = 35H',CR,LF,'Track format error (Side A).',0
PT_FORMAT3_ERR           DB      CR,LF,BELL, 'CMD=16H,  Error = 36H',CR,LF,'Track format error (Side B).',0
PT_FORMAT4_ERR           DB      CR,LF,BELL, 'CMD=16H,  Error = 37H',CR,LF,'Restore error after formatting disk.',0

PT_RT_ERR_HUNG          DB     CR,LF,BELL, 'CMD=15H,  Error = 39H,',CR,LF,'Disk Read Track error,DRIVE NOT READY.',0
PT_RT_ERR       DB      CR,LF,BELL, 'CMD=15H',CR,LF,'Error = 39H,',CR,LF,'Disk Read Track (unknown) error.',0

PT_DRIVE_INACTIVE       DB CR,LF,BELL, 'CMD=3AH',CR,LF,'No detected disk in drive.',0
PT_DRIVE_DOOR           DB CR,LF,BELL, 'CMD=3BH',CR,LF,'Drive door open?',0
PT_BUFFER_OVERFLOW DB    CR,LF,BELL, 'Two many sectors were requested in a single multi sector R/W request.',0

PT_ABORT_FLAG   DB       CR,LF,BELL, 'User aborted current command.',0
PT_CMD_RANGE_ERR DB      CR,LF,BELL, 'Error = 3DH,  CMD out or range.',0
TIMEOUT_ERROR_MSG DB     CR,LF,BELL, 'Timeout Error. ZFDC Board did not reply back in time.',0
```