Errata 88-HDSK-ME01
Altair Hard Disk (88-HDSK)
Preliminary Documentation, October 1977

Page 28

1. IV Byte C (Address 35) is Output Mode (not Input Mode, as written.)
2. IV Byte D (Address 36): the DRDST bit (the CRC test result for reads) is only on IV Bit 5 (pin 3). (It is not available on IV bit 3 as the documentation claims.)

Page 29

1. The first line of Page 29 is wrong. The Output Control Signals at IV address 37 are IV Byte E (not IV Byte B).
2. The $\overline{\text{CLR}}$ bit in IV Byte E is Bit 4 (pin 4), not Bit 5 (pin 3). (See Figure 4-10, the Schematic for the 88-HDSK Data Card.)

Errata 88-HDSK-ME02
Altair Hard Disk (88-HDSK)
Preliminary Documentation, October 1977

Page 35-36

1. For all IV Bytes shown in Table 3-D (8T32 Byte Functions), the User Data Bits are reversed from the way they appear to the Altair via the Set Byte and Read Status commands, as follows:

User Data Bit-Pin		Altair Bit
0	8	7
1	7	6
2	6	5
3	5	4
4	4	3
5	3	2
6	2	1
7	1	0

2. IV Bytes H, I, and J (addressed 17, 18, and 19) all invert the data. This means the data must be inverted when written to these IV Bytes. For example, to select Cylinder Address 0 in IV Byte J, you would write 255 (decimal) to IV byte J.
3. Note that the Read Status command rewrites 7 bits in IV Byte H (Address 17). User Bits 4-7 get set according to the Unit bits in the command byte (bits 3:2). User Bits 1:3 always get set to 011b (Extension Select low, Platter and Head select high). User Bit 0 (Start/Stop all drives) is unchanged by the Read Status command.
4. The Read Status command will not complete if the selected Unit is not Ready. This means that the Ready line on P2 of the 88-HDSK Interface Card must be pulled low for testing IV Bytes, when no disk drive is connected.

Errata 88-HDSK-ME03
Altair Hard Disk (88-HDSK)
Preliminary Documentation, October 1977

Page 32

Port 1, Section A, Command Response (IN 161) provides Error Flags, as described in the Addendum Page 7, and reproduced (with 1 erratum) here:

Bit	**1 Means:**[1]	**Error may occur in:**
0	Drive not ready	Any command except INITIALIZE and SET IV BYTE
1	Illegal sector	SEEK, READ SECTOR, WRITE SECTOR, FORMAT, & READ UNFORMATTED commands
2	CRC error in sector read	READ SECTOR, READ UNFORMATTED[2] commands
3	CRC error in header read	SEEK, READ SECTOR, WRITE SECTOR & FORMAT commands
4	Header has wrong sector	Same as in bit 3
5	Header has wrong cylinder	Same as in bit 3[3]
6	Header has wrong head	Same as in bit 3
7	Write protect	Same as in bit 1[4]

NOTES

1. All bits of port 161 are ones on the first read after the controller is turned on.

2. Always occurs on an unformatted read of a formatted sector.

3. Occurs spuriously when one of these commands is issued for a drive different from the one specified in the last seek. This spurious error is ignored by the write logic.

4. Only relevant during a Write Sector command. If a sector is write-protected, data may not be written to it. The Write Sector command is ignored and the error flag is set.

(**Erratum:** The Write Protect bit is also relevant during a FORMAT command. If the drive is write-protected, then the FORMAT command will fail.)

Errata 88-HDSK-ME04
Altair Hard Disk (88-HDSK)
Preliminary Documentation, October 1977

Page 122, first new paragraph

If the 88-HDSK Interface Card is installed, but no disk drive is connected to the 88-HDSK Interface Card, then the four Seeking lines on the 88-HDSK Interface Card must be pulled high (inactive), so that the controller can complete its initialization routine and respond to commands from the computer.

A simple way to do this is to install a temporary jumper from the positive terminal of C3 to the positive terminal of C8. This provides +5V to the termination resistors of the disk interface, and effectively pulls all signals high, including Ready. Be sure to remove this jumper before connecting a disk drive to the 88-HDSK Interface Card!

For further testing, it is convenient to build a simple tester board that plugs into P2 of the 88-HDSK Interface Card, and has 20 DIP switches to ground -- one for each input signal except Read Clock and Read Data.

Errata 88-HDSK-ME05
Altair Hard Disk (88-HDSK)
Preliminary Documentation, October 1977

Page 35

Corrected sample routine for Set Byte command

The sample on page 35 is nonfunctional because the Controller Ready signal is never cleared. Additionally, it is generally better to wait for commands to complete (by waiting for the controller to become Ready) at the end of each command, rather than to check to see of the controller is ready at the beginning of each command. This allows the error bits that are returned at the end of some commands to be checked when that command completes. (See line 140.)

The controller sets the Altair Data Port Available bit 4.5 uSec after receiving this command. This is only 1 or 2 8080 instruction times, so there is no need to spin waiting for this bit at line 135. Similarly, the controller sets the Controller Ready bit 1 uS after receiving the data, so there is no need to spin waiting for this bit at line 140.

This program inverts the data written to IV Byte 17 because IV Byte 17 is inverting. (See lines 25 and 35.)

```
10 REM EXAMPLE- START/STOP DRIVE D (1<=D<=4)
15 E=(D-1)*4     :REM PUT UNIT BITS IN PLACE
20 I=17          :REM SELECT DISK FUNCTION CONTROL IV BYTE
25 J=NOT(E+128)  :REM SELECT DRIVE, SET START/STOP BIT
30 GOSUB 100
35 J=NOT E       :REM CLEAR START/STOP BIT
40 GOSUB 100
45 END


100 REM GENERAL SUBROUTINE TO WRITE J TO IV-PORT I
105 OUT167,I      :REM IV ADDRESS
110 A=INP(167)    :REM RESET ALTAIR DATA PORT HANDSHAKE
115 A=INP(161)    :REM RESET CONTROLLER READY
120 OUT163,128    :REM 'SET BYTE' INITIATES THE COMMAND
125 IF 128 AND INP(166)=0 THEN E$="PORT NOT READY":GOTO 200
135 OUT167,J      :REM WRITE J TO IV-PORT I
140 IF 128 AND INP(160)=0 THEN E$="CTLR NOT READY":GOTO 200
150 RETURN
200 PRINT "ERROR: ";E$:STOP
```

Assembly Language Example:

```
CREADY   equ   160
CSTAT    equ   161
ACMD     equ   163
ADSTA    equ   166
ADATA    equ   167

CSETIV   equ   80h

;===Subroutine=======================================================
;Send IV byte
; On entry:
;   c = IV byte address
;   e = IV byte data
; On Exit:
; carry bit set for controller timeout
; a,d trashed, all other registers preserved
;=====================================================================
SENDIV:  in    ACMD        ;reset CMDACK in ACSTA

         mov   a,c         ;IV Byte address
         out   ADATA       ;low byte of command
         in    ADATA       ;clear ADPA bit in ADSTA

         mvi   a,CSETIV    ;Set IV Byte command
         out   ACMD        ;initiate command now

         mvi   d,0         ;256x37/2 uS=4.7 mS timeout

SIVLP1:  dcr   d           ;(4)timeout?
         stc               ;(4) set carry for error return
         rz                ;(5) error return with carry set
         in    ADSTA       ;(10)wait for ADPA
         rlc               ;(4)test ADPA
         jnc   SIVLP1      ;(10)(37 cycles through loop)

         mov   a,e         ;send IV Byte data
         out   ADATA

         mvi   d,0         ;256x37/2 uS=4.7 mS timeout

SIVLP2:  dcr   d           ;(4)timeout?
         stc               ;(4) set carry for error return
         rz                ;(5) error return with carry set
         in    CREADY      ;(10)Is the controller done?
         rlc               ;(4)look at msb=CRDY
         jnc   SIVLP2      ; (10)(37 cycles through loop)

         in    CSTAT       ;reset CRDY flag
         ora   a           ;and get A=error code
         jnz   UNXERR      ;Go deal with unexplained error
         ret
```

Errata 88-HDSK-ME06
Altair Hard Disk (88-HDSK)
Preliminary Documentation, October 1977

Page 36

Corrected sample routine for Read Status command

The sample on page 36 is nonfunctional because the Controller Ready signal is never cleared. Additionally, it is generally better to wait for commands to complete (by waiting for the controller to become Ready) at the end of each command, rather than to check to see of the controller is ready at the beginning of each command. This allows the error bits that are returned at the end of some commands to be checked when that command completes. (See line 135.)

The controller will wait for the specified drive to be Ready before it reads any IV byte and send it to the computer. Therefore, the computer should wait for the controller to be Ready before reading the Status Byte data. (See line 125. Note that the WAIT command is an infinite loop. Better would be a loop with a timeout.)

The controller sets the Controller Ready bit 1 uS after writing the status to the Controller Data Port, so there is no need to spin waiting for this bit at line 130.

```
100 REM GENERAL SUBROUTINE TO READ A FROM IV-PORT I, UNIT D
105 OUT167,I           :REM IV ADDRESS
110 A=INP(165)         :REM RESET DATA PORT HANDSHAKE
115 A=INP(161)         :REM RESET CONTROLLER READY
120 OUT163, (D-1)*4+96:REM INITIATE THE COMMAND
125 WAIT164,128        :REM WAIT FOR CONTROLLER DATA
130 A=INP(165)         :REM A=IV-PORT I DATA
135 IF 128 AND INP(160)=128 THEN RETURN
140 PRINT “ERROR: COMMAND DID NOT COMPLETE”:STOP
```

Assembly Language Example:

```
CREADY   equ   160
CSTAT    equ   161
ACMD     equ   163
CDSTA    equ   164
CDADA    equ   165
ADATA    equ   167

CRSTAT   equ   60h

;===Subroutine=================================================
;Read IV byte
; On entry:
;   c = IV byte address
; On Exit:
;   carry set means no IV data because the drive was not ready
;   a = IV Byte Status
;   b trashed, all other registers preserved
;===============================================================
READIV:  in     CDATA        ;clear CDA
         in     ACMD         ;reset CMDACK in ACSTA

         mov    a,c          ;low byte of command 1st
         out    ADATA
         mvi    a, CRSTAT    ;read status command
         out    ACMD         ;and issue command

         mvi    b,0          ;256x37/2 uS=4.7 mS timeout

RIVLP1:  dcr    b            ;(4) timeout?
         stc    c            ;(4)set carry for error return
         rz                  ;(5)timeout: carry set
         in     CREADY       ;(10)Is the controller done?
         rlc                 ;(4)look at msb=CRDY
         jnc    RIVLP1       ;(10)(37 cycles through loop)

         in     CSTAT        ;reset CRDY flag
         jnz    UNXERR       ;Go deal with unexplained error

         in     CDSTA        ;CDA should already be set
         rlc                 ;test CDA
         cmc                 ;so c means error
         rc                  ;not there? error return

         in     CDATA        ;Read & report IV Byte
         ret                 ;carry clear for normal return
```

Errata 88-HDSK-ME07
Altair Hard Disk (88-HDSK)
Preliminary Documentation, October 1977

Page 37

Corrected sample routine for Write Buffer command

The sample on page 37 is nonfunctional because the Controller Ready signal is never cleared. Additionally, it is generally better to wait for commands to complete (by waiting for the controller to become Ready) at the end of each command, rather than to check to see of the controller is ready at the beginning of each command. This allows the error bits that are returned at the end of some commands to be checked when that command completes. (See line 165.)

The controller sets the Altair Data Port Available bit 7 uSec after receiving this command. This is only 3 or 4 8080 instruction times, so there is no need to spin waiting for this bit at line 140. Similarly, the controller sets the Controller Ready bit 1 uS after receiving the last data byte, so there is no need to spin waiting for this bit at line 165.

```
100 REM GENERAL SUBROUTINE TO WRITE A$ TO BUFFER B
105 L=LEN(A$)
110 IF L>256 OR L=0 THEN E$="ILLEGAL STRING":GOTO 200
115 IF B>3 THEN E$="ILLEGAL BUFFER":GOTO 200
120 OUT167,L MOD 256    :REM COMMAND LOW BYTE IS BYTE COUNT
125 A=INP(167)          :REM RESET ALTAIR DATA PORT HANDSHAKE
130 A=INP(161)          :REM RESET CONTROLLER READY
135 OUT163,64+B         :REM 'WRITE BUFFER' INITIATES CMD
140 IF 128 AND INP(166)=0 THEN E$="PORT NOT READY":GOTO 200
150 FOR I=1 TO L        :REM WRITE L BYTES TO THE CONTROLLER
155 OUT 167,MID$(A$,1)
160 NEXT I
165 IF 128 AND INP(160)=0 THEN E$="CTLR TIMEOUT":GOTO 200
175 RETURN
200 PRINT "ERROR: ";E$:STOP
```

Assembly Language Example:

```
CREADY   equ   160
CSTAT    equ   161
ACMD     equ   163
ADSTA    equ   166
ADATA    equ   167

CWRBUF   equ   40h

;===Subroutine=====================================
; Write to controller buffer
; On Entry:
;   b = number of bytes to write (0 means 256)
;   c = controller buffer number
;  hl = address of write data
; On Exit:
;   carry set if timeout error on controller
;   a,bc,hl trashed, de preserved
;==================================================
WRBUF:   in      ADATA       ;reset ADPA in ADSTA
         in      ACMD        ;reset CMDACK in ACSTA

         mov     a,b         ;transfer byte count
         out     ADATA       ;low byte of command 1st

         mov     a,c         ;Controller buffer number
         ori     CWRBUF      ;combine with write buffer command
         out     ACMD        ;and issue command

         mvi     c,0         ; 256x37/2 uS=4.7 mS timeout

WRBLP1:  dcr     c           ;(4)timeout?
         stc                 ;(4)carry set for error
         rz                  ;(5)exit with carry set for timeout

         in      ADSTA       ;(10)Wait for data port to be ready
         rlc                 ;(4)msb=ADPA
         jnc     WRBLP1      ;(10) (37 cycles through loop)

; loop to write b bytes from (hl) to controller

WRBLP2:  mov     a,m         ;get RAM buffer data
         out     ADATA       ;send to controller
         inx     h           ;next
         dcr     b
         jnz     WRBLP2

         in      CREADY      ;CRDY should already be set
         rlc                 ;test CRDY
         cmc                 ;so carry means error
         ret                 ;success/fail in carry
```

Errata 88-HDSK-ME08
Altair Hard Disk (88-HDSK)
Preliminary Documentation, October 1977

Page 38

Corrected sample routine for Read Buffer command

The sample on page 38 is nonfunctional because the Controller Ready signal is never cleared. Additionally, it is generally better to wait for commands to complete (by waiting for the controller to become Ready) at the end of each command, rather than to check to see of the controller is ready at the beginning of each command. This allows the error bits that are returned at the end of some commands to be checked when that command completes. (See line 165.)

The controller sets the Altair Data Port Available bit 7 uSec after receiving this command. This is only 3 or 4 8080 instruction times, so there is no need to spin waiting for this bit at line 135. Similarly, the controller sets the Controller Ready bit 1 uS after receiving the last data byte, so there is no need to spin waiting for this bit at line 165.

```
100 REM GENERAL SUBR TO READ L BYTES FROM BUFFER B INTO A$
105 IF L>256 OR L=0 THEN E$="ILLEGAL LENGTH":GOTO 200
110 IF B>3 THEN E$="ILLEGAL BUFFER":GOTO 200
115 OUT167,L MOD 256 :REM COMMAND LOW BYTE IS BYTE COUNT
120 A=INP(165)       :REM RESET CONTROLLER DATA PORT HANDSHAKE
125 A=INP(161)       :REM RESET CONTROLLER READY
130 OUT163,80+B      :REM 'READ BUFFER'INITIATES CMD
135 IF 128 AND INP(164)=0 THEN E$="PORT NOT READY":GOTO 200
145 A$=''
150 FOR I=1 TO L     :REM READ L BYTES FROM THE CONTROLLER
155 A$=''=A$+INP(165)
160 NEXT I
165 IF 128 AND INP(160)=0 THEN E$="CTLR TIMEOUT":GOTO 200
175 RETURN
200 PRINT "ERROR: ";E$:STOP
```

Assembly Language Example:

```
CREADY   equ   160
CSTAT    equ   161
ACMD     equ   163
CDSTA    equ   164
CDADA    equ   165
ADATA    equ   167

CRDBUF   equ   50h

;===Subroutine=====================================
; Read controller buffer
; On Entry:
;   b = number of bytes to write (0 means 256)
;   c = controller buffer number
;  hl = address for data
; On Exit:
;   a,bc,hl trashed, de preserved
;===========================================
RDBUF:  in      CDATA       ;reset CDA in CDSTA
        in      ACMD        ;reset CMDACK in ACSTA

        mov     a,b         ;transfer byte count
        out     ADATA       ;low byte of command 1st

        mov     a,c         ;buffer number
        ori     CRDBUF      ;combine with read buffer command
        out     ACMD        ;and issue command

        mvi     c,0         ; 256x37/2 uS=4.7 mS timeout

RDBLP1: dcr     c           ;(4)timeout?
        stc                 ;(4)carry set for error
        rz                  ;(5)exit with carry set for timeout

        in      CDSTA       ;(10)Wait for data port to be ready
        rlc                 ;(4)msb=CDA
        jnc     RDBLP1      ;(10) (37 cycles through loop)

; Loop to read b bytes from controller into memory at hl

RDBLP2: in      CDATA       ;get a data byte
        mov     m,a         ;stash in buffer
        inx     h           ;next
        dcr     b
        jnz     RDBLP2

        in      CREADY      ;CRDY should already be set
        rlc                 ;test CRDY
        cmc                 ;so carry means error
        ret                 ;success/fail in carry
```

Errata 88-HDSK-ME09
Altair Hard Disk (88-HDSK)
Preliminary Documentation, October 1977

Page 39

Corrected sample routine for Write Sector command

The sample on page 39 is nonfunctional because the Controller Ready signal is never cleared. Additionally, it is generally better to wait for commands to complete (by waiting for the controller to become Ready) at the end of each command, rather than to check to see of the controller is ready at the beginning of each command. This allows the error bits that are returned at the end of this command to be checked when the command completes. (See lines 125-135.)

```
100 REM SUBR TO WRITE BUFFER B TO HEAD H, SECTOR A ON UNIT D
105 IF B>3 THEN E$=”ILLEGAL BUFFER”:GOTO 200
110 IF S>23 THEN E$=”ILLEGAL SECTOR”:GOTO 200
115 OUT167,H*64+A       :REM HEAD AND SECTOR
120 OUT163,(D-1)*4+B+32 :REM UNIT, BUFF, WRITE SECTOR COMMAND
125 WAIT INP(160),128   :REM WAIT FOR SECTOR-WRITE TO COMPLETE
130 C=INP(161)          :REM RESET CTLR READY, READ STATUS
135 IF C=0 THEN RETURN
140 REM.....................DECODE ERROR FLAGS
145 IF C AND 1=1 THEN PRINT “DRIVE NOT READY”
150 IF C AND 2=2 THEN PRINT “ILLEGAL SECTOR”
155 IF C AND 8=8 THEN PRINT “CRC ERROR IN SECTOR HEADER”
160 IF C AND 16=16 THEN PRINT “HEADER HAS WRONG SECTOR”
165 IF C AND 32=32 THEN PRINT “HEADER HAS WRONG CYLINDER”
170 IF C AND 64=64 THEN PRINT “HEADER HAS WRONG HEAD”
175 IF C AND 128=128 THE PRINT “WRITE PROTECTED”
200 PRINT “ERROR: “;E$:STOP
```

Assembly Language Example:

```
CREADY   equ   160
CSTAT    equ   161
ACMD     equ   163
ADATA    equ   167

CWRSEC   equ   20h

;===SUBROUTINE=======================================================
; write Sector
; (Note: This could easily be combined with Read Sector.)
; On Entry:
; b = sector number
; c = controller buffer number
; h = head number
; On Exit:
;  requested controller buffer has been written to disk sector
;  carry set if timeout for controller
;  Z set if no errors
;  a = error flags
;  bc trashed, others preserved
;=============================================================
WRSEC:   in    ACMD        ;(10)reset CMDACK in ACSTA

         mov   a,h         ;head number
         rrc
         rrc
         rrc               ;...to bits 7:5
         ora   b           ;combine with sector
         out   ADATA

         mov   a,c         ;buffer number
         ora   CWRSEC      ;create write sector command
         out   ACMD        ;issue command

         lxi   b,5102      ;5102*49/2=125 mS (5 rotations)

WRSLP1:  dcx   b           ;(7)timeout?
         mov   a,b         ;(5)
         ora   c           ;(4)16-bit test
         stc               ;(4)
         rz                ;(5)carry set for error return
         in    CREADY      ;(10)Is the controller done?
         rlc               ;(4)look at msb=CRDY
         jnc   WRSLP1      ;(10)(49 cycles through loop)

         in    CSTAT       ;reset CRDY flag, read error flags
         ora   a           ;test for errors
         ret               ;Z set if okay
```

Errata 88-HDSK-ME10
Altair Hard Disk (88-HDSK)
Preliminary Documentation, October 1977

Page 40

Corrected sample routine for Read Sector command

The sample on page 40 is nonfunctional because the Controller Ready signal is never cleared. Additionally, it is generally better to wait for commands to complete (by waiting for the controller to become Ready) at the end of each command, rather than to check to see of the controller is ready at the beginning of each command. This allows the error bits that are returned at the end of this command to be checked when the command completes. (See lines 130-140.)

```
100 REM SUBR TO READ HEAD H, SECTOR A ON UNIT D INTO BUFFER B
105 IF B>3 THEN E$=”ILLEGAL BUFFER”:GOTO 200
110 IF S>23 THEN E$=”ILLEGAL SECTOR”:GOTO 200
120 OUT167,H*64+A:REM........HEAD AND SECTOR
125 OUT163,(D-1)*4+B+48 :REM UNIT, BUFF, WRITE SECTOR COMMAND
130 WAIT INP(160),128   :REM WAIT FOR SECTOR-WRITE TO COMPLETE
135 C=INP(161)          :REM RESET CTLR READY, READ STATUS
140 IF C=0 THEN RETURN
145 REM.....................DECODE ERROR FLAGS
150 IF C AND 1=1 THEN PRINT “DRIVE NOT READY”
155 IF C AND 2=2 THEN PRINT “ILLEGAL SECTOR”
160 IF C AND 4=4 THEN PRINT “CRC ERROR IN SECTOR DATA”
165 IF C AND 8=8 THEN PRINT “CRC ERROR IN SECTOR HEADER”
170 IF C AND 16=16 THEN PRINT “HEADER HAS WRONG SECTOR”
175 IF C AND 32=32 THEN PRINT “HEADER HAS WRONG CYLINDER”
180 IF C AND 64=64 THEN PRINT “HEADER HAS WRONG HEAD”
185 IF C AND 128=128 THE PRINT “WRITE PROTECTED”
190 STOP
200 PRINT “ERROR: “;E$:STOP
```

Assembly Language Example:

```
CREADY   equ   160
CSTAT    equ   161
ACMD     equ   163
ADATA    equ   167

CRDSEC   equ   30h

;===SUBROUTINE=======================================================
; Read Sector
; (Note: This could easily be combined with Write Sector.)
; On Entry:
; b = sector number
; c = controller buffer number
; h = head number
; On Exit:
;  disk sector data is in requested controller buffer
;  carry set if timeout for controller
;  Z set if no errors
;  a = error flags
;  bc trashed, others preserved
;=====================================================================
RDSEC:   in    ACMD        ;(10)reset CMDACK in ACSTA

         mov   a,h         ;head number
         rrc
         rrc
         rrc               ;...to bits 7:5
         ora   b           ;combine with sector
         out   ADATA

         mov   a,c         ;buffer number
         ora   CRDSEC      ;create read sector command
         out   ACMD        ;issue command

         lxi   b,5102      ;5102*49/2=125 mS (5 rotation times)

RDSLP1:  dcx   b           ;(7)timeout?
         mov   a,b         ;(5)
         ora   c           ;(4)16-bit test
         stc               ;(4)
         rz                ;(5)carry set for error return
         in    CREADY      ;(10)Is the controller done?
         rlc               ;(4)look at msb=CRDY
         jnc   RDSLP1      ;(10)(49 cycles through loop)

         in    CSTAT       ;reset CRDY flag, read error flags
         ori   7Fh         ;test for errors (Write protect is not an error)
         ret               ;Z set if okay
```

Errata 88-HDSK-ME11
Altair Hard Disk (88-HDSK)
Preliminary Documentation, October 1977

Page 41

Corrected sample routine for Seek command

The sample on page 41 is nonfunctional because the Controller Ready signal is never cleared. Additionally, it is generally better to wait for commands to complete (by waiting for the controller to become Ready) at the end of each command, rather than to check to see of the controller is ready at the beginning of each command. This allows the error bits that are returned at the end of this command to be checked when the command completes. (See lines 135-145.)

```
100 REM SUBROUTINE TO SEEK CYLINDER C ON UNIT D
105 IF C>405 THEN E$="ILLEGAL CYLINDER":GOTO 200
115 A=0:IF C>255 THEN C=C-256:A=1
125 OUT167,C             :REM LOW 7 BITS OF CYLINDER
130 OUT 163,(D-1)*4+A+0 :REM UNIT, HIGH BIT OF CYL, SEEK CMD
135 WAIT INP(160),128    :REM WAIT FOR SECTOR-READ TO COMPLETE
140 C=INP(161)           :REM RESET CTLR READY, READ STATUS
145 IF C=0 THEN RETURN
150 REM.....................DECODE ERROR FLAGS
155 IF C AND 1=1 THEN PRINT "DRIVE NOT READY"
160 IF C AND 4=4 THEN PRINT "CRC ERROR IN SECTOR DATA"
165 IF C AND 8=8 THEN PRINT "CRC ERROR IN SECTOR HEADER"
170 IF C AND 16=16 THEN PRINT "HEADER HAS WRONG SECTOR"
175 IF C AND 32=32 THEN PRINT "HEADER HAS WRONG CYLINDER"
180 IF C AND 64=64 THEN PRINT "HEADER HAS WRONG HEAD"
185 IF C AND 128=128 THE PRINT "WRITE PROTECTED"
190 STOP
200 PRINT "ERROR: ";E$:STOP
```

Assembly Language Example:

```
CREADY   equ   160
CSTAT    equ   161
ACMD     equ   163
ADATA    equ   167

CSEEK    equ   00h

;===SUBROUTINE=======================================================
; Seek
; (Note: This could easily be combined with Read & Write Sector.)
; On Entry:
; hl = cylinder number
; On Exit:
;  disk sector data is in requested controller buffer
;  carry set if timeout for controller
;  Z set if no errors
;  a = error flags
;  bc trashed, others preserved
;=====================================================================
RDSEC:   in    ACMD        ;(10)reset CMDACK in ACSTA

         mov   a,l         ;cylinder number bits 7:0
         out   ADATA

         mov   a,h         ;cylinder number bit 8
;        ora   CSEEK       ;create read sector command (CSEEK=0)
         out   ACMD        ;issue command

         lxi   b,5102      ;5106*49/2=130 mS (2* max seek time)

RDSLP1:  dcx   b           ;(7)timeout?
         mov   a,b         ;(5)
         ora   c           ;(4)16-bit test
         stc               ;(4)
         rz                ;(5)carry set for error return
         in    CREADY      ;(10)Is the controller done?
         rlc               ;(4)look at msb=CRDY
         jnc   RDSLP1      ;(10)(49 cycles through loop)

         in    CSTAT       ;reset CRDY flag, read error flags
         ori   7Fh         ;test for errors (Write protect is not an error)
         ret               ;Z set if okay
```

Errata 88-HDSK-ME12
Altair Hard Disk (88-HDSK)
Preliminary Documentation, October 1977

Page 25

IV Byte A8 (Address 2) is used for communicating error conditions to the computer, as part of 88-HDSK communications. (The General comment for this IV Byte is incorrect.)

Page 26

IV Byte A13 (Address 7) provides signals from the computer to the controller, either Write Data, Set Byte data, or the lower 8 bits of a command instruction. (The General comment for this IV Byte has the data direction incorrect.)