

# System Implementation

SA 400/450 minifloppy<sup>®</sup> Diskette  
Storage Drives with an 8080A/FD 1771  
Single Density System

Application Bulletin

# Table of Contents

---

1.0	Introduction	
1.1	General	1
2.0	SA400/450 Minidiskette Drives	3
2.1	General	3
2.2	Drive Performance	3
2.3	Drive Interface	3
3.0	Host Computer System	5
3.1	General	5
3.2	Intel 8255 Programmable Peripheral Interface (PPI)	7
4.0	Western Digital FD1771 Controller/Formatter	9
4.1	General	9
4.2	FD1171 Interface	9
4.3	Controller Command Initiation	9
4.4	Data Separation	10

# List of Illustrations

---

1. FM Recommended Format .....	1
2. System Functional Block Diagram .....	5
3. System Schematic .....	6
4. Initialize Routine .....	11
5. Seek Routine .....	12
6. Read Routine .....	14
7. Write Routine .....	15

## FOREWORD

Shugart Associates Applications Staff designed this 8080/1771 System as an aid to our users. Obviously there are many different methods of implementing the hardware and software of this system, but we feel that this is a general purpose method of implementation. This Applications Bulletin is intended to be a concise statement of implementation, for a detailed description of the system elements refer to the following manuals:

- Intel 8080 Microcomputer Systems User's Manual, P/N 98-153B
- Western Digital FD1771A/B-01, FD1771A/B-02, Floppy Disk Formatter/Controller
- SA400 minifloppy® Diskette Storage Drive OEM Manual, P/N 54102-1
- SA450 minifloppy® Double-Sided Diskette Storage Drive OEM Manual (available first quarter 1978).

The SA450 minifloppy® Double-Sided Diskette Storage Drive is presented for the first time in a systems environment. It should be noted that this drive is capable of MFM double-density encoding and has a track to track access time of 25 milliseconds. However, due to limitations imposed by the Western Digital Controller it functions as a SA400 with two sides.

Shugart Associates does not assume responsibility for the use or implementation of this system nor any infringements of patents or other rights of third parties which may result from its use.

## 1.0 INTRODUCTION

### 1.1 General

This Application Bulletin briefly describes the parameters necessary to interface the SA400/450 minifloppy® Diskette Storage Drives with a Western Digital FD1771 Controller/Formatter, using an Intel 8080A Microcomputer System.

The discussion is based on a 16 sector, 128 byte per sector format as shown in Figure 1. It should be noted that when using a SA450 double-sided drive the ID field format is changed. Byte 2 of the ID now contains side select information instead of zeroes.

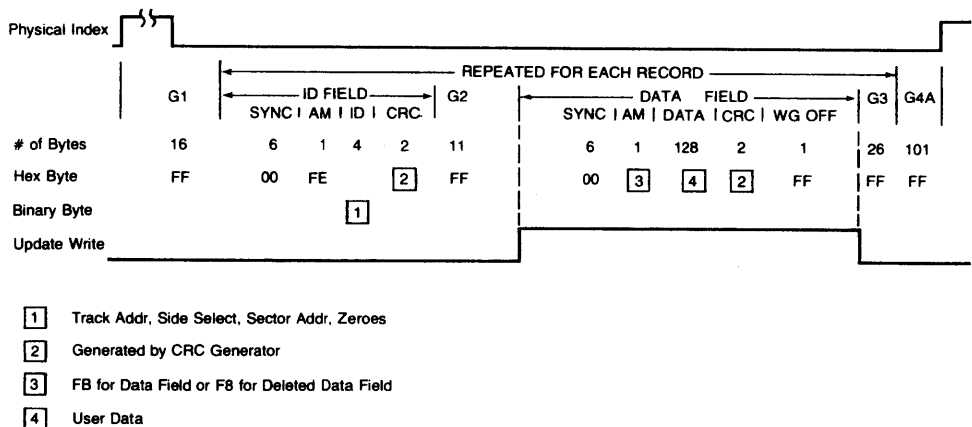


FIGURE 1. FM RECOMMENDED FORMAT — 128 Byte & 16 Records/Track.



## 2.0 SA400/450 MINIDISKETTE DRIVES

### 2.1 General

The SA400 minifloppy<sup>®</sup> Diskette Drive is a magnetic media storage device organized as 35 independent tracks with track zero being the outer most track with respect to the center of the disk.

Each track has a capacity of 3125 bytes (unformatted), hence a total disk capacity of 109.4 K bytes.

When formatted using the format shown in Figure 1, each track will have a user data capacity of 2048 bytes for a total user capacity of 71.68 K bytes.

The SA450 is an enhanced double-sided version of the SA400. The basic capacities are listed below:

- 70 tracks (two tracks per cylinder)
- 6250 bytes per cylinder, unformatted
- 4096 bytes per cylinder, formatted\*
- 143.36 K bytes per diskette, formatted\*

### 2.2 Drive Performance

The basic serial data rate of drive is 125 K bits per second which translates to 15.6 K bytes per second or 1 byte transferred every 64 microseconds.

The SA400/450 contains a D.C. spindle drive motor with an interface on/off control. To insure maximum motor life, the motor should be turned off when no further disk commands are anticipated. When turning the motor on, the host computer system should allow for a spindle motor up to speed and settle time of 1 second.

Each track of the disk drive can be accessed in 40 milliseconds with an additional 10 milliseconds of track settle time. The track settle time is non-cumulative, that is, when performing multiple steps it is added only to the last track accessed.

The head load can be activated either from the spindle motor "ON" control signal or by selecting the drive. With either method a delay of 75 milliseconds is necessary after head load.

### 2.3 Drive Interface

The SA400/450 is interfaced by 12 TTL compatible signals. The following interface signals are accompanied by a brief description:

*Read Data (Output)*—This signal is the digitized serial data, read from the diskette.

*Write Data (Input)*—This signal is the digitized serial data to be written on the diskette.

*Write Gate (Input)*—When activated, this signal causes 'write data' to be written on the diskette.

*Write Protect (Output)*—Indicates a write protected diskette has been inserted in the drive.

*Step (Input)*—For each step pulse, the R/W head moves one track.

*Direction Select (Input)*—Selects the direction of the R/W head will move when a pulse occurs on the 'step' line.

*Track 00 (Output)*—This signal indicates when the R/W head is positioned over track 0.

*Drive Select (Input)*—3 Lines to assign logical drive address.

*Index/Sector (Output)*—Pulse indicating that the physical index/sector hole of the diskette has passed over the index sensor.

*Motor On (Input)*—This signal controls spindle motor on/off.

\* When formatted as shown in Figure 1



### 3.0 HOST COMPUTER SYSTEM

#### 3.1 General

The Intel 8080A Microprocessor (MPU) is a monolithic 8-bit microprocessor forming the host system. Peripheral I/O control is treated as a memory address (Memory Mapped I/O) as shown in Figure 2.

It should be noted that due to total system timing constraints, read/write routines must be performed within 56  $\mu$ seconds. In order to accomplish this, bit 7 of the PPI Port C is tied to the Data Request line (DRQ) (Figure 3). By loading the Accumulator with DRQ status, it is possible to perform an inclusive OR of the Accumulator (ORA A) which results in the sign bit being set if there is a data request. Based on the status of the sign bit, a branch to the appropriate routine can be made. This technique eliminates the need to perform a separate check on the status bits using one of the logical instructions and results in a significant time savings.

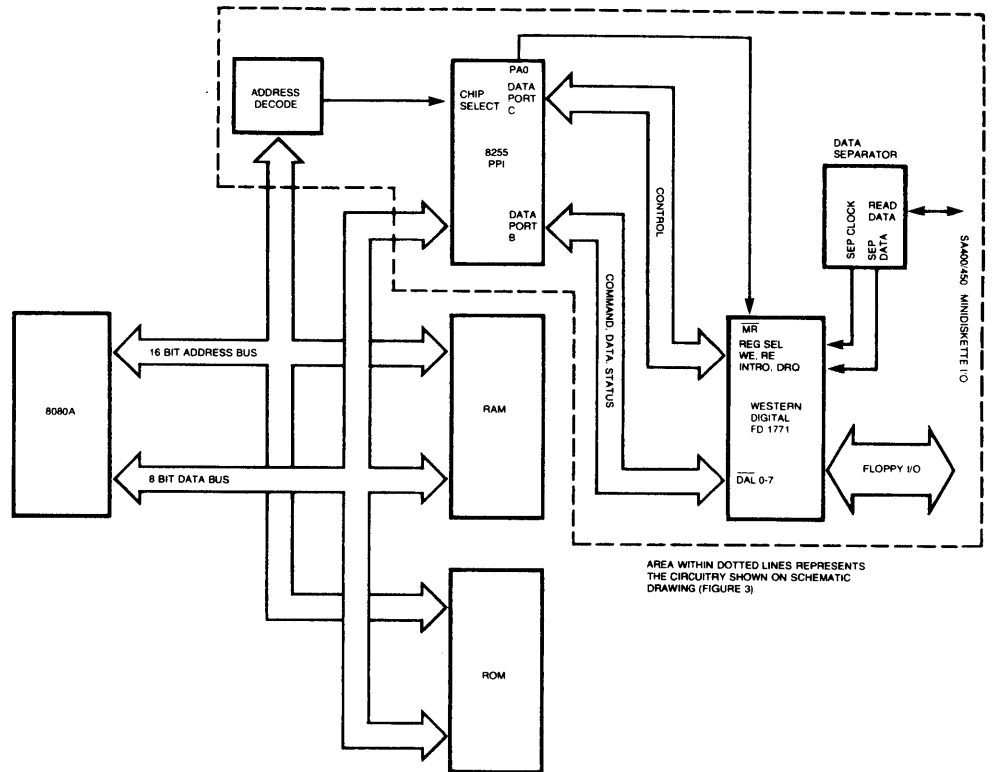


FIGURE 2. System Functional Block Diagram.





### 3.2 Intel 8255 Programmable Peripheral Interface (PPI)

The FD1771 is interfaced to the 8080 through the 8255 Programmable Peripheral Interface (PPI) which is treated as a memory address (Figure 2). All commands, data, status, and controls are handled by the PPI using Mode 0. (Basic Input/Output)

The PPI provides a universal means of adapting peripheral devices to the 8080. The PPI interfaces the MPU data bus and Read and Write channels, through the 8228 System Controller and Bus Driver. Three address lines ( $A/O$ ,  $A1$ , and  $\overline{A15}$ ) of the 8080 are connected directly to the 8255.  $\overline{A15}$  is connected to CS of the 8255 giving the PPI a memory address of 8000 (hex).  $A0$  and  $A1$  directly access registers in the 8255.

In this application Port B of the PPI will interface with the FD1771 data lines, Port C will handle the control lines and Port A will connect to FD1771 and the drive.

Six of the outputs of the PPI are inverted ( $PA0$ – $PA3$ ,  $PC2$  and  $PC3$ ). The inverters are necessary because when commanding the PPI Port B to change direction the outputs of all of the ports go low. This causes false signals to go to the FD1771 which in turn deselects the drive and turns off the motor.

As shown in Figure 3 the 8080 has an external clock. The 8224 Clock Generator/Driver provides the necessary timing and strobe signals for the 8080.



## **4.0 WESTERN DIGITAL FD1771 CONTROLLER/FORMATTER**

### **4.1 General**

The Western Digital FD1771 is a MOS/LSI device that performs the functions of a general purpose Floppy Disk Controller/Formatter. The FD1771 is compatible with the IBM 3740 Data Entry System Format, but can be programmed for variable formats.

There are two constraints for formatting with the FD1771. The first is the ID field, which must be 4 bytes long with byte 1 containing the Track Address and byte 3 containing the sector Address. The other is GAP 2 (the gap between the ID field and the Data Address Mark), which must contain 11 bytes of hex "FF" and a sync area of 6 bytes of zeros. Other gaps and data field lengths may be varied to suit individual format requirements.

### **4.2 FD1771 Interface**

The FD1771 interface to the host system processor is through the 8 data lines and associated control signals.

By reading and writing selected registers within the FD1771 command, data, and status bytes are transferred between the host computer and the FD1771. This is accomplished by programming the register select pins A0, A1. For further information refer to the FD1771 data sheet.

### **4.3 Controller Command Initiation**

The FD1771 will accept eleven macro commands which perform the various disk drive functions. These commands are divided into four types and are briefly described as follow:

#### **Type I Commands.**

Restore—Causes the addressed drive to seek to track zero.

Seek—Causes the addressed drive to position the R/W head over the track specified by the host computer.

Step—Causes the drive to step one track in the direction previously selected.

Step In—Causes the drive to step one track toward track 35.

Step Out—Causes the drive to step one track toward track zero.

#### **Type II Commands.**

Read Command—Transfers a full sector of data, a byte at a time, to the host computer.

Write Command—Transfers a full sector of data, a byte at a time, from the host computer to the disk drive.

#### **Type III Commands.**

Read Track—Transfers all bytes of data on a track to the host computer. Read begins with the first index pulse encountered.

Read Address—Transfers the next encountered ID field to the host computer (Refer to Figure 1), places the Sector Address into the sector register, and checks the two byte CRC field.

Write Track—In effect, this command is the format command. The host computer must supply all gap, ID field, and data bytes with the exception of the address marks and CRC bytes.

## **Type IV Command.**

Force Interrupt—Any command may be terminated and an interrupt generated by the use of this command.

In order to initiate the FD1771 commands, the host computer must load the desired command byte into the command register. Prior to this, the data register or sector register must be loaded to provide the information required by the command. Refer to Figure 4 PPI initialize. During a data transfer between the FD1771 and the host computer, the 'DRQ', 'RE', and 'WE' signals will comprise the handshake lines. The data transfer handshake is shown in Figures 6 and 7.

At the end of every operation a status handshake occurs where a status byte is available to the host computer. The 'INTRQ' and 'RE' lines are used in the status handshake. Refer to Figures 5, 6 and 7, for the status handshake.

### **4.4 Data Separation**

The FD1771 is equipped with an internal data separator. But due to the fact that the internal data window is not synchronous with the serial data and can cause errors in worst case data patterns, an external separator of the type shown in Figure 3 is recommended.

The external data separator is of the type known as a 'hard' data separator. A one shot is triggered on the leading edge of the clock pulse. This 'window' extends into the middle of the bit cell. If a pulse is present in the area of the window, it is decoded as a '1' bit. Otherwise it is decoded as a zero bit.

It is possible for any data separator to get out of phase (decode clock pulses as data pulses). Therefore, a 4 bit counter is present to detect more than 3 missing clock pulses. In FM encoding the clock stream will never have more than 3 missing clock pulses in a row. Therefore, if 4 missing clock pulses occur, the data window is made to rephase on the next pulse in the stream (a clock pulse).

During address marks, the clock stream will have 3 missing clock pulses in a row. During this time data pulses are present, but due to the absence of clock pulses the data window would be terminated. Therefore, a 'false clock' circuit is present to generate a data window in the absence of clock pulses. This window is generated from the leading edge of each data pulse. If no clock pulse occurs, the trailing edge of the 'false clock' window will generate a data window to provide data decoding in the absence of clock pulses.

The required data pulse width for the FD1771 (external separator mode) is 300 to 700 nanoseconds. Since the SA400 Drive generates a 1.2 microsecond nominal data pulse, this pulse must be reduced in width. In the circuit shown in Figure 3, the pulse width has been reduced to approximately 400 nanoseconds.

```

; INIT
;
;THIS PROGRAM IS DESIGNED TO PERFORM SEEKS,
;READS, WRITES AND FORMATS USING THE
;WESTERN DIGITAL FD1771 FLOPPY CONTROLLER
;CHIP INTERFACED TO AN SA400 AND A 8080 MPU
0091 CTRL EQU 91H
;CONTROL WORD FORMAT:
;MODE=0
;PORTA=DATA PORT (INPUTS)
;PORTB=STATUS AND COMMAND LINES (OUTPUTS)
;PORTC=INPUTS, 0-3
;PORTC=OUTPUTS, 4-7
8000 PORTA EQU 8000H ;PORT A ADDRESS
8001 PORTB EQU 8001H ;PORT B ADDRESS
8002 PORTC EQU 8002H ;PORT C ADDRESS
8003 CWR EQU 8003H ;CONTROL WORD ADDRESS
0520 CHARS EQU 520H ;COMMAND CHARACTERS
; ENTERED VIA CONSOLE
07D0 SEEK EQU 7D0H ;SEEK TRACK ROUTINE
0B10 READ EQU 0B10H ;READ SECTOR ROUTINE
0B80 WRITE EQU 0B80H ;WRITE SECTOR ROUTINE
0798 PIN EQU 798H ;PORT B SET AS INPUTS
; ROUTINE

0600 ORG 600H
;
;INITIALIZE THE PPI (POWER UP)
;
;MR ALWAYS=1 AFTER INITIALIZATION
;
0600 31FF0B INIT LXI SP,0BFFH SET THE STACK
0603 CD9807 CALL PIN ;PORTB INPUTS
0606 210080 LXI H,PORTA ;PORTA ADRS
0609 3E01 MVI A,01H ;MR=0,DR SEL,MOT ON
060B 77 MOV M,A ;WRITE PORTA
060C 110280 LXI D,PORTC ;LOAD PORTC ADRS
060F 3E00 MVI A,0 ;RE,WE=1
0611 12 STAX D ;WRITE PORTC
;
;PPI INITIALIZATION DONE
;RESTORE TO TRACK 0 IS AUTOMATIC
;
0612 3E00 TRK0 MVI A,00H ;MR=1
0614 77 MOV M,A ;WRITE PORTA
0615 3E08 MVI A,08H ;READ ENAB-STAT REG
0617 12 STAX D ;(RE=0) WRITE PORTC
0618 3A0180 LDA PORTB ;READ PORTB—GET
; STATUS
061B 4F MOV C,A ;SAVE STAT REG
061C 3E00 MVI A,00H ;RE=1 (MR,WE=1)
061E 12 STAX D ;WRITE PORTC
061F 79 MOV A,C ;MOV STATUS TO A
0620 E604 ANI 4H ;GET TRAK 0 STATUS
0622 FE04 CPI 4H ;TRAK 0?
0624 CA1206 JZ TRKO ;NO

```

FIGURE 4. Initialize Routine.

```

;
;DRIVE NOW AT TRACK 0 — TEST READY
;
0627 79      MOV A,C      ;RESTOR STATUS IN A
0628 E680    ANI 80H     ;GET READY STATUS
062A FE80    CPI 80H     ;READY?
062C CA3506  JZ INTERP    ;YES GO TO IDLE LOOP
062F 010000  LXI B,0H     ;ERROR CODE
0632 CD0007  CALL ERROR

```

FIGURE 4. Initialize Routine (Continued).

```

;THIS ROUTINE IS DESIGNED TO SEEK THE
;DRIVE TO A TRACK SPECIFIED BY THE 3RD & 4TH
;CHARACTERS ENTERED BY AN OPERATOR.
8000 PORTA  EQU 8000H   ;PORT A ADDRESS
8001 PORTB  EQU 8001H   ;PORT B ADDRESS
8002 PORTC  EQU 8002H   ;PORT C ADDRESS
0793 POUT   EQU 793H   ;PORT B SET AS OUTPUTS ROUTINE
0798 PIN    EQU 798H   ;PORT B SET AS INPUTS ROUTINE
0769 STATUS EQU 769H   ;ROUTINE, CONVERTS STATUS TO
                        ASCII PRINTABLE DATA
0520 CHARS  EQU 0520H   ;COMMAND CHARACTERS ENTERED
                        VIA CONSOLE.
;
07D0      ORG 7D0H     ;
;
;FORM THE TRACK ADDRESS FROM THE 3RD & 4TH
;CHARACTERS. 4TH CHARACTER MIGHT BE NEGATIVE
;IF ONLY ONE CHAR WAS ENTERED.
;
07D0 2A2205 SEEK  LHLD CHARS+2 ;GET BOTH CHARS
07D3 7C      MOV A,H      ;XFR LS CHAR
07D4 B7      ORA A        ;TERM ?
07D5 F2DC07 JP TWO       ;NO
07D8 7D      MOV A,L      ;LOAD SINGLE CHAR
07D9 C3E207 JMP NEWTRK    ;YES
07DC 7D      TWO  MOV A,L      ;XFR MS CHAR
07DD 07      RLC          ;SHIFT TO MS POSITION
07DE 07      RLC
07DF 07      RLC
07E0 07      RLC
07E1 84      ADD H        ;MERGE CHARS

```

FIGURE 5. Seek Routine.

```

;NOW PUT NEW TRACK ADDRESS IN
FDC DATA REGISTER
;
07E2 322008 NEWTRK STA TRACK ;SAVE TRACK ADDRESS
07E5 CD9307 CALL POUT ;PORTB=OUTPUTS
07E8 210280 LXI H,PORTC ;GET PORT C ADRS
07EB 0603 MVI B,03H ;A0,A1=1
07ED 70 MOV M,B ;WRITE PORTC
07EE 3A2008 LDA TRACK ;TRAK ADRS
07F1 2F CMA ;INVERT FOR WD BUS
07F2 320180 STA PORTB ;WRITE PORTB
07F5 0607 MVI B,07H ;WRITE TO DATA REG
07F7 70 MOV M,B ;WRITE PORTC
07F8 0600 B,00H
07FA 70 MOV M,B ;WRITE PORTC
;
;INITIATE SEEK COMMAND
;
07FB 3E1F MVI A,1FH ;SEEK 40 MS STEP
07FD 2F CMA
07FE 320180 STA PORTB ;WRITE PORTB
0801 0604 MVI B,04H ;WRITE TO CMD REG
0803 70 MOV M,B ;WRITE PORTC
0804 0600 MVI B,00H ;RE,WE = 1
0806 70 MOV M,B ;WRITE PORTC
;
;WAIT FOR END OF SEEK —
THEN REPORT STATUS
;
0807 CD9807 CALL PIN
080A 7E WAIT MOV A,M ;WAIT FOR END OF SEEK
080B E640 ANI 40H
080D CA0A08 JZ WAIT
0810 3E08 MVI A,08H ;STAT REG READ
0812 77 MOV M,A ;WRITE PORTC
0813 3A0180 LDA PORTB ;BRING STATUS
0816 0600 MVI B,00H ;RE,WE=1
0818 70 MOV M,B ;WRITE PORT B
0819 2F CMA ;INVERT
081A E618 ANI 18H ;SEEK AND CRC BITS
;
081C CD6907 CALL ;REPORT TO CONSOLE
STATUS
081F C9 RET
;
0820 00 TRACK BYTE 0
07D0 07D0 END SEEK

```

COMMAND  
HANDSHAKE

STATUS  
HANDSHAKE

FIGURE 5. Seek Routine (Continued).



```

;READ READ SECTOR ROUTINE
;
;READ SECTOR ROUTINE INITIATES THE READ
;COMMAND AND TRANSFERS ALL THE DATA FOR A
;SELECTED SECTOR TO THE BOTTOM OF MEMORY
;BEGINNING AT LOCATION 5FF
;R,XX READ SECTOR, SECTOR ADRS
8000   PORTA   EQU 8000H   ;PORT A ADDRESS
8001   PORTB   EQU 8001H   ;PORT B ADDRESS
8002   PORTC   EQU 8002H   ;PORT C ADDRESS
0793   POUT    EQU 793H    ;PORT B SET AS OUTPUTS ROUTINE
0798   PIN     EQU 798H    ;PORT B SET AS INPUTS ROUTINE
0769   STATUS  EQU 769H    ;ROUTINE CONVERTS STATUS TO
                                ASCII PRINTABLE DATA
05FF   DATBUF  EQU 5FFH    ;BEGINNING ADRS OF "READ"
                                DATA BUFFER
0520   CHARS   EQU 520H    ;COMMAND CHARACTERS ENTERED
                                VIA CONSOLE
;
0B10   ;                ORG 0B10H
;
0B10   2A2205  READ    LHLD CHARS+2;GET BOTH CHARS
0B13   7C      ;                MOV A,H      ;XFER LS CHAR
0B14   B7      ;                ORA A        ;TERM?
0B15   F21C0B  ;                JP TWO      ;NO
0B18   7D      ;                MOV A,L     ;LOAD SINGLE CHAR
0B19   C3220B  ;                JMP SECTOR
0B1C   7D      TWO    MOV A,L     ;XFER MS CHAR
0B1D   07      ;                RLC        ;SHIFT TO MS POSITION
0B1E   07      ;                RLC
0B1F   07      ;                RLC
0B20   07      ;                RLC
0B21   84      ;                ADD H      ;MERGE CHARS
0B22   327C0B  SECTOR STA SECSTR ;STOR SECTOR
0B25   CD9307  ;                CALL POUT  ;PORTB OUTPUTS
0B28   210280  ;                LXI H,PORTC ;GET PORTC ADRS
0B2B   0602    ;                MVI B,02H  ;SECTOR REGISTER
0B2D   70      ;                MOV M,B   ;WRITE PORTC
0B2E   3A7C0B  ;                LDA SECSTR  ;SECTOR ADRS
0B31   2F      ;                CMA        ;INVERT FOR WD BUS
0B32   320180  ;                STA PORT B  ;WRITE PORTB
0B35   0606    ;                MVI B,06H  ;WRITE TO SECTOR
                                REG
0B37   70      ;                MOV M,B   ;WRITE PORTC
;
;INITIATE THE READ COMMAND
;
0B38   0600    ;                MV B,0     ;SEL CMD REG
0B3A   70      ;                MOV M,B   ;WRITE PORTC
0B3B   3E88    ;                MVI A,88H  ;READ CMD
0B3D   2F      ;                CMA        ;INVRT FOR WD BUS
0B3E   320180  ;                STA PORTB  ;WRITE PORTB
0B41   0604    ;                MVI B,04H  ;ISSUE READ TO
                                CMD REG
0B43   70      ;                MOV M,B   ;WRITE PORTC

```

COMMAND  
HANDSHAKE

FIGURE 6. Read Routine.

```

;
;WAIT FOR END OF READ — THEN REPORT
;
0B44 0603          MVI B,03H      ;SEL DATA REG
0B46 70           MOV M,B        ;WRITE PORTC
0B47 CD9807       CALL PIN        ;PORTB = INPUTS
0B4A 11FF05       LXI D,DATBUF    ;FWA OF DATA
0B4D 0603          MVI B,03H      ;RE, WE=1
0B4F C35C0B       JMP DLOOP
0B52 3E0B GD      MVI A,0BH      ;RE=0
0B54 77           MOV M,A        ;WRITE PORTC
0B55 3A0180       LDA PORTB      ;GET DATA
0B58 2F           CMA            ;INVERT DATA
0B59 12           STAX D         ;SAVE IT
0B5A 1B           DCX D          ;BUMP INDEX
0B5B 70           MOV M,B        ;RE=1,PORTC
0B5C 7E DLOOP     MOV A,M        ;GET STATUS
                                PORTC
0B5D B7           ORA A          ;DRQ=1?
0B5E FA520B       JM GD          ;YES
0B61 E640         ANI 40H        ;INTRQ SET?
0B63 CA5C0B       JX DLOOP       ;NO
;
;READ DONE — GET STATUS
;
0B66 3E00         MVI A,0        ;ADRS STAT REG
0B68 77           MOV M,A        ;WRITE PORTC
0B69 3E08         MVI A,08H       ;STROBE RE=0
0B6B 77           MOV M,A        ;WRITE PORTC
0B6C EB          XCHG
0B6D 227E0B       SHLD ISAVE     ;SAVE INDEX TO
                                DATA
0B70 EB          XCHG            ;RESTOR PORTC
                                ADRS
0B71 3A0180       LDA PORTB      ;GET STAT BYTE
0B74 0600         MVI B,0        ;STAT HANDSHAKE
0B76 70           MOV M,B        ;WRITE PORTC
0B77 2F           CMA            ;INVERT STAT
                                BYTE
0B78 CD6907       CALL STATUS   ;REPORT STATUS
0B7B C9          RET
;
0B7C 0000 SECSTR  WORD 0        ;SECTOR ADRS
0B7E 0000 ISAVE   WORD 0        ;DATA INDEX
                                STORAGE AREA
0B10             END READ

```

FIGURE 6. Read Routine (Continued).

```

;WRITE WRITE SECTOR ROUTINE
;
;WRITE SECTOR ROUTINE INITIATES THE WRITE
;COMMAND AND TRANSFERS ALL THE DATA FOR A
;SELECTED SECTOR
;
;W,XX = WRITE SECTOR,SECTOR ADRS

```

FIGURE 7. Write Routine.



WAIT FOR END OF READ — THEN REPORT

0BB4	0603		MVI B,03H	;SEL DATA REG	}	DATA TRANSFER HANDSHAKE
0BB6	70		MOV M,B	;WRITE PORTC		
0BB7	110180		LXI D,PORTB	;PORTB ADRS		
0BBA	0603		MVI B,03H	;RE,WE=1		
0BBC	C3C70B		JMP DLOOP			
0BBF	3E8D	GD	MVI A,8DH	;LOAD DATA		
0BC1	2F		CMA	;INVRT DATA		
0BC2	12		STAX D	;WRITE PORT B		
0BC3	3E07		MVI A,07H	;WE=0		
0BC5	77		MOV M,A	;WRITE PORTC		
0BC6	70		MOV M,B	;RE=1,PORTC		
0BC7	7E	DLOOP	MOV A,M	;GET STATUS PORTC		
0BC8	B7		ORA A	;DRQ=1		
0BC9	FABF0B		JM GD	;YES		
0BCC	E640		ANI 40H	;INTERQ SET		
0BCE	CAC70B		JZ DLOOP	;NO		
READ DONE — GET STATUS						
0BD1	CD9807		CALL PIN	;PORTB INPUTS	}	STATUS HANDSHAKE
0BD4	3E00		MVI A,0	;ADRS STAT REG		
0BD6	77		MOV M,A	;WRITE PORTC		
0BD7	3E08		MVI A,08H	;STROBE RE=0		
0BD9	77		MOV M,A	;WRITE PORTC		
0BDA	3A0180		LDA PORTB	;GET STAT BYTE		
0BDD	0600		MVI B,0	;STAT HANDSHAKE		
0BDF	70		MOV M,B	;WRITE PORTC		
0BE0	2F		CMA	;INVERT STAT BYTE		
0BE1	CD6907		CALL STATUS	;REPORT STATUS		
0BE4	C9		RET			
0BE5	0000	SECSTR	WORD 0	;SECTOR ADRS		
	0B80		END READ			

FIGURE 7. Write Routine (Continued).



415 Oakmead Parkway, Sunnyvale, California 94086  
Phone: (408) 733-0100 TWX: 910 339 9355 SHUGART SUVL

11/77 39003

Copyright 1977 Copyright 1977

PRINTED IN U.S.A.