

# RM 65 FAMILY

## FLOPPY DISK CONTROLLER (FDC) MODULE

# RM 65 FAMILY

### USER'S MANUAL

#### ELECTRONIC DEVICES DIVISION REGIONAL ROCKWELL SALES OFFICES

##### HOME OFFICE

Electronic Devices Division  
Rockwell International  
4311 Jamboree Road  
Newport Beach, California 92660  
Tel: 800-854-8099, 800-854-8090  
In California 800-422-4230  
TWX: 910 591-1698

##### UNITED STATES

Electronic Devices Division  
Rockwell International  
1842 Reynolds  
Irvine, California 92714  
(714) 632-3710  
TWX 910 505-2518

Electronic Devices Division  
Rockwell International  
921 Bowser Road  
Richardson, Texas 75080  
(214) 996-6500  
Telex: 73-307

Electronic Devices Division  
Rockwell International  
10700 West Higgins Rd., Suite 102  
Rosemont, Illinois 60018  
(312) 297-8862  
TWX: 910 233-0179 (RI MED ROSM)

Electronic Devices Division  
Rockwell International  
5001B Greentree  
Executive Campus, Rt. 73  
Marlton, New Jersey 08053  
(609) 596-0090  
TWX: 710 940-1377

##### EUROPE

Electronic Devices Division  
Rockwell International GmbH  
Fraunhoferstrasse 11  
D-8033 Munchen-Martinsried  
West Germany  
(089) 859-9575  
Telex: 05212850 rimd d

Electronic Devices Division  
Rockwell International  
Healthrow House, Bath Rd  
Cranford, Hounslow  
Middlesex, England  
(01) 759-9911 Ext. 35  
Telex: 851-25463

Electronic Devices  
Rockwell Collins  
Via Boccaccio, 23  
20123 Milano  
Italy  
488 74 79

##### FAR EAST

Electronic Devices Division  
Rockwell International Overseas Corp  
Itotipa Hirakawa-cho Bldg  
7-6, 2-chome, Hirakawa-cho  
Chiyoda-ku, Tokyo 102, Japan  
(03) 265-8806  
Telex J22198

#### YOUR LOCAL REPRESENTATIVE

4/82



**Rockwell International**

...where science gets down to business



**Rockwell International**

Document No.  
29801N02  
Order No. 802  
Rev. 1 June 1982

**RM 65 FAMILY**

**FLOPPY DISK  
CONTROLLER  
(FDC) MODULE**

**USER'S  
MANUAL**

**RM 65 FAMILY**

NOTICE

Rockwell International does not assume any liability arising out of the application or use of any products, circuit, or software described herein, neither does it convey any license under its patent rights nor the patent rights of others. Rockwell International further reserves the right to make changes in any products herein without notice.

TABLE OF CONTENTS

<u>Section</u>		<u>Page</u>
1	INTRODUCTION	
	1.1 Purpose/Function .....	1-1
	1.2 Features .....	1-3
	1.3 Characteristics .....	1-4
	1.4 Reference Documents .....	1-5
2	INSTALLATION AND OPERATION	
	2.1 Unpacking .....	2-1
	2.2 Operating Options .....	2-1
	2.2.1 Base Address Selection .....	2-4
	2.2.2 Bank Selection .....	2-5
	2.2.3 Program ROM and I/O Selection .....	2-6
	2.2.4 DMA Channel Selection .....	2-7
	2.2.5 Standard/Mini-Floppy Selection .....	2-7
	2.2.6 Precompensation .....	2-10
	2.2.7 Dual Head Drive .....	2-10
	2.3 Installing the Module .....	2-11
	2.4 Removing the Module .....	2-13
	2.5 FDC Module Calibration Procedure .....	2-14
3	FUNCTIONAL AND INTERFACE DESCRIPTION	
	3.1 Memory Map .....	3-1
	3.2 Functional Description .....	3-4
	3.3 Interface Description .....	3-7
	3.3.1 Mini-Floppy Disk Drive Interface Cable Assembly .....	3-17
	3.3.2 Standard Floppy Disk Drive Interface Cable Assembly .....	3-18
4	FDC MODULE PRIMITIVE ROUTINES	
	4.1 Primitive Routines Description .....	4-1
	4.2 Calling Considerations .....	4-12
	4.3 Disk Initialization .....	4-13
	4.3.1 Procedure .....	4-13
	4.3.2 Default Conditions .....	4-14
	4.4 Disk Primitive Operation .....	4-15
	4.4.1 Formatting a Disk .....	4-16
	4.4.2 Reading a Sector .....	4-18
	4.4.3 Writing a Sector .....	4-19
	4.5 Primitive Routines Error Handling .....	4-19
	4.6 Primitive Routines Variables .....	4-20
5	USING AIM 65 DOS VERSION 1.0	
	5.1 Initialization .....	5-2
	5.2 Disk Format .....	5-3
	5.3 Primary Operator Commands .....	5-4
	5.3.1 List the Directory .....	5-5
	5.3.2 Backup a Disk .....	5-5
	5.3.3 Format a Disk .....	5-6
	5.3.4 List a File .....	5-8
	5.3.5 Delete a File .....	5-8
	5.3.6 Recover a File .....	5-9
	5.3.7 Extension of UTILITY Functions .....	5-10

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

<u>Section</u>	<u>Page</u>	<u>Figure</u>	<u>Page</u>
5.4 File Handling Under Operator Control .....	5-10	1-1 FDC Module .....	1-2
5.4.1 Using the AIM 65 Monitor .....	5-10	1-2 FDC Module Outline .....	1-5
5.4.2 Using the AIM 65 Editor .....	5-11		
5.4.3 Using the AIM 65 Assembler .....	5-12	2-1 FDC Module Detail .....	2-3
5.4.4 Using AIM 65 BASIC .....	5-15	2-2 Standard/Mini-Floppy Selection Header Locations ..	2-8
5.4.5 Using AIM 65 FORTH .....	5-16		
5.4.6 Using AIM 65 PL/65 .....	5-18	3-1 FDC Module Block Diagram .....	3-5
5.4.7 Using AIM 65 Instant Pascal .....	5-18	3-2 Mini-Floppy Disk Drive Interface Cable .....	3-17
5.5 File Heading Under Program Control .....	5-20	3-3 Standard Floppy Disk Drive Interface Cable .....	3-18
5.5.1 Writing and Reading Data Files ....	5-21		
5.5.2 DOS Compatibility Considerations ..	5-26	5-1 AIM 65 DOS 1.0 Disk Format .....	5-3
5.6 DOS Error Reporting .....	5-27	5-2 Driver to Assemble from Source Code on Disk .....	5-13
5.7 DOS Variables .....	5-31	5-3 Program Control File Handler .....	5-22
		5-4 Example Program Using File Handler .....	5-28
6 USING AIM 65/40 DOS VERSION 1.0			
6.1 Initialization .....	6-2	6-1 AIM 65/40 DOS 1.0 Disk Format .....	6-3
6.2 Disk Format .....	6-3	6-2 Program Control File Handler .....	6-17
6.3 Primary Operator Commands .....	6-3	6-3 Example Program Using File Handler .....	6-23
6.3.1 List the Directory .....	6-5		
6.3.2 Backup a Disk .....	6-5	B-1 AIM 65 FORTH Floppy Disk Example .....	B-3
6.3.3 Format a Disk .....	6-6	B-2 AIM 65/40 FORTH Floppy Disk Example .....	B-4
6.3.4 List a File .....	6-8		
6.3.5 Delete a File .....	6-9	C-1 AIM 65 Floppy Disk System Example .....	C-3
6.3.6 Recover a File .....	6-9		
6.3.7 Extension of UTILITY Functions ....	6-10		
6.4 File Handling Under Operator Control .....	6-10		
6.4.1 Using the AIM 65/40 Monitor .....	6-10		
6.4.2 Using the AIM 65/40 Editor .....	6-11		
6.4.3 Using the AIM 65/40 Assembler .....	6-12		
6.4.4 Using AIM 65/40 BASIC .....	6-14		
6.4.5 Using AIM 65/40 FORTH .....	6-14		
6.4.6 Using AIM 65/40 PL/65 .....	6-15		
6.5 File Handling Under Program Control .....	6-15		
6.5.1 Reading and Writing Data Files ....	6-16		
6.5.2 Compatibility with DOS Functions ..	6-21		
6.6 DOS Error Reporting .....	6-22		
6.7 DOS Variables .....	6-2		
APPENDIX A Disk Drive Configuration .....	A-1		
A.1 5 " Disk Drive Configuratio.....	A-1		
A.1.1 Shugart SA400 .....	A-2		
A.1.2 Shugart SA450 .....	A-3		
A.1.3 Pertec FD200 .....	A-3		
A.2 8" Disk Drive Configuration .....	A-4		
A.2.1 Shugart SA800 .....	A-5		
A.3 DMA Configuration .....	A-6		
APPENDIX B FORTH and the FDC Module .....	B-1		
APPENDIX C FDC Firmware Program Listing .....	C-1		
ENCLOSURE FDC Module Schematic			

LIST OF TABLES

SECTION 1

<u>Table</u>	<u>Page</u>
1-1 FDC Module Physical and Electrical Characteristics .....	1-4
2-1 FDC Module Switches, Jumpers and Connectors .....	2-2
2-2 Base Address Selection PROM Programmed Outputs ...	2-4
2-3 Bank Select Switch Positions .....	2-5
2-4 Program ROM and I/O Selection Jumper Positions ...	2-6
2-5 DMA Channel Selection Jumper Positions .....	2-7
2-6 Standard/Mini-Floppy Selection Header Position ...	2-9
2-7 Precompensation Jumper Positions .....	2-10
2-8 Dual Head Drive Selection Jumper Positions .....	2-10
3-1 FDC Module Memory Map .....	3-2
3-2 FDC Module I/O Memory Map .....	3-3
3-3 Connector P1 (RM 65 Bus) Pin Assignments .....	3-8
3-4 Connector J1 (Disk Drive) Pin Assignments .....	3-11
3-5 Connector P1 (RM 65 Bus) Signal Descriptions .....	3-12
3-6 Connector J1 (Disk Drive) Signal Descriptions .....	3-14
4-1 Primitive Routine Addresses .....	4-3
4-2 FDC Module I/O Register Addresses .....	4-3
4-3 Primitive Routine Description .....	4-4
4-4 Primary User-Alterable Variables Default Values ..	4-14
4-5 Disk Format Parameters .....	4-17
4-6 Primitive Routine Error Definitions .....	4-21
4-7 Primitive Routine Variables .....	4-22
5-1 Commonly Used AIM 65 DOS 1.0 Subroutines .....	5-24
5-2 Commonly Used AIM 65 DOS 1.0 Variables .....	5-25
5-3 AIM 65 DOS 1.0 Error Definitions .....	5-30
5-4 AIM 65 DOS 1.0 Variables .....	5-32
6-1 Commonly Used AIM 65/40 DOS 1.0 Subroutines .....	6-19
6-2 Commonly Used AIM 65/40 DOS 1.0 Variables .....	6-20
6-3 AIM 65/40 DOS 1.0 Error Definitions .....	6-25
6-4 AIM 65/50 DOS 1.0 Variables .....	6-27

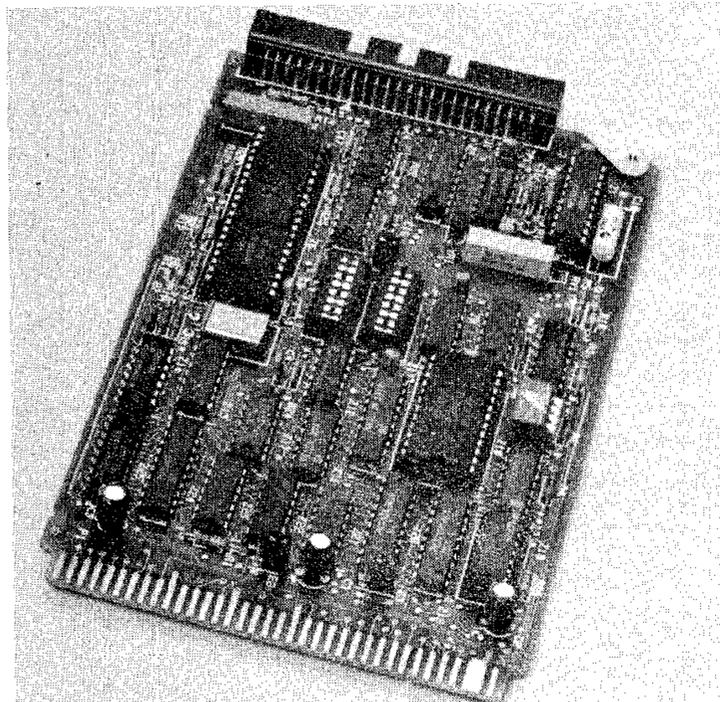
INTRODUCTION

1.1 PURPOSE/FUNCTION

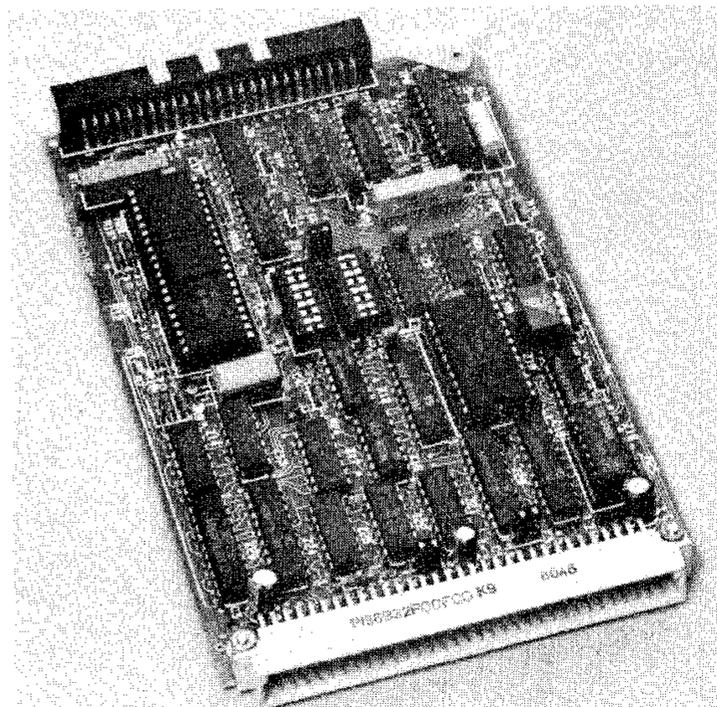
The RM 65 Floppy Disk Controller (FDC) module supports an AIM 65, AIM 65/40 or RM 65 SBC based system with floppy disk mass storage capability. The FDC module controls up to four standard (8") or mini-(5-1/4") floppy disk drives, single- or double-sided, soft sectored with either single-density (FM) or double-density (MFM) format. (8" double-density format requires installation of the RM 65 DMA Controller module--RM65-5104). Software control of media density allows single- or double-density disks to be used in any of the interfacing drives. Two DIP headers configure the FDC module to interface with either standard or mini-floppy disk drives and an on-board jumper selects single- or double-sided operation. The FDC module directly interfaces to most popular drives with only switch and/or header changes.

Bank Select and Bank Select Enable switches dedicate the module to one of two 65K byte memory banks, or assign it common to both banks. Program ROM and I/O Disable switches allow the on-board ROM or FDC module I/O to be enabled or disabled. The module base address is assigned by the Base Address Selection PROM for operation with the Program ROM firmware. When the Program ROM is deselected, the FDC module I/O can be assigned to any page (256 bytes) by replacing the Base Address Selection PROM.

The FDC module is available in a 72-pin Edge Connector version and in a 64-pin Eurocard version. Both versions are shown in Figure 1-1. The pin assignments of the two versions are identical except the edge connector version has four additional pins connected to +5VDC, and four unused pins (see Table 3-3 for pin assignments).



a. Edge Connector Version



b. Eurocard Version

Figure 1-1. FDC Module

The FDC module and optional firmware are identified as follows:

Order No.	Description
RM65-5101	FDC Module (Edge Connector) with on-board ROM*
RM65-5101N	FDC Module (Edge Connector) without on-board ROM
RM65-5101E	FDC Module (Euroconnector) with on-board ROM*
RM65-5101NE	FDC Module (Euroconnector) without on-board ROM
A65-090	AIM 65 DOS 1.0 ROM**
A65/40-7090	AIM 65/40 DOS 1.0 ROM**

NOTES

\* Program ROM contains FDC module primitive subroutines only.  
 \*\*Program ROM contains FDC module primitive subroutines and DOS functions integrated with host computer I/O.

### 1.2 FEATURES

- . RM 65 Bus compatible
- . Compact size--about 4" x 6-1/4" (100 mm x 160 mm)
- . Buffered address, data and control lines
- . Supports single- or double-sided, standard or mini-floppy disk drives
- . Controls up to four disk drives
- . Interfaces directly to Shugart SA850 or SA450 disk drives, and other popular floppy disk drives
- . Supports single-density IBM 3740 (FM) or double-density IBM System 34 (MFM) formats
- . DMA data transfer capability
- . Supports interrupt-driven or polled operation
- . Bipolar PROM Base Address decoding
- . Switches or jumpers for
  - Bank Selection to one or two banks
  - Single- or double-sided operation
  - Select or deselect ROM
  - Select or deselect I/O
- . On-board header configures I/O for 8" or 5-1/4" drives
- . On-board ROM contains FDC module primitive subroutines
- . Optional Disk Operating System (DOS) Version 1.0 ROMs support AIM 65 or AIM 65/40 microcomputers
- . Fully assembled, tested and warranted

### 1.3 CHARACTERISTICS

The physical and electrical characteristics of the the FDC module are listed in Table 1-1.

Table 1-1. FDC Module Physical and Electrical Characteristics

Characteristics	Value
<b>Dimensions (see Figure 1-2)</b>	
<b>Edge Connector Version</b>	
Width	3.9 in. (100 mm)
Length	6.5 in. (164 mm)
Height	0.56 in. (14 mm)
Weight	4.8 oz. (130 g)
<b>Eurocard Version</b>	
Width	3.94 in. (100 mm)
Length	6.30 in. (160 mm)
Height	0.56 in. (14 mm)
Weight	5.2 oz. (140 g)
<b>Environment</b>	
Operating Temperature	0°C to 70°C
Storage Temperature	-40°C to 85°C
Relative Humidity	0% to 85% (without condensation)
<b>Power Requirements</b>	
	+5 Vdc ± 5%
	600 mA (3.2 W) - typical
	900 mA (4.8 W) - maximum
	+12 Vdc ± 5%
	60 mA (0.76W) - typical
	100 mA (1.26W) - maximum
<b>Interface Connector P1</b>	
Edge Connector Version	72-pin edge connector (0.100 in. centers)
Eurocard Version	64-pin plug (0.100 in. centers) per DIN 41612 (Row b not installed)
<b>Module I/O Interface J1</b>	
	50-pin mass terminated (0.100 in. centers). Mates with T&B/Ansley Part No. 609-5001M or equivalent.
<b>NOTES</b>	
1. The height includes the maximum values for component height above the board surface (0.4 in. for populated modules), printed circuit board (PCB) thickness (0.062 in.), and pin extension beyond the PCB (0.1 in.).	
2. The length does not include the module ejector.	
3. The Eurocard dimensions conform to DIN 41612.	

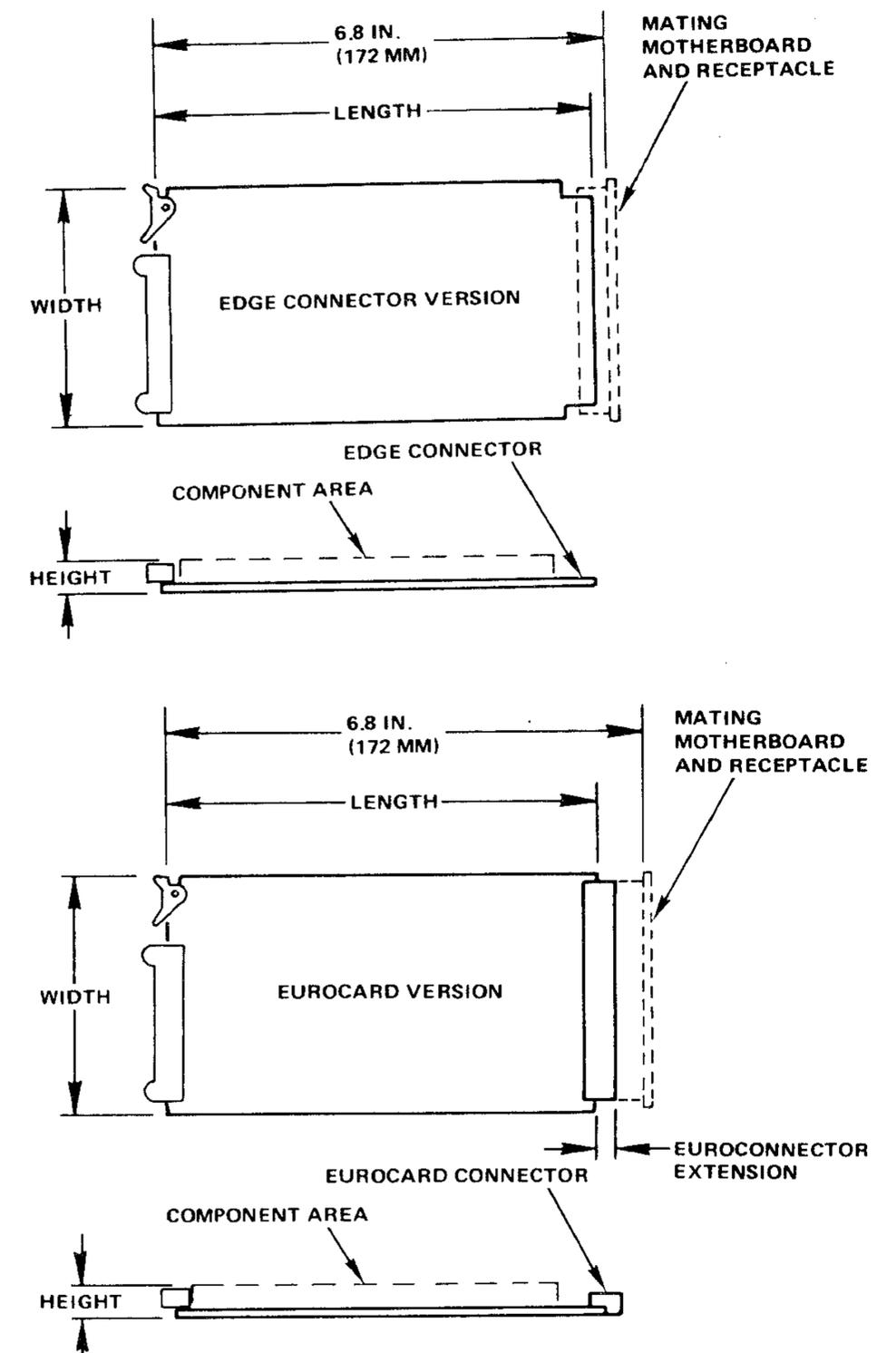


Figure 1-2. FDC Module Outline

#### 1.4 REFERENCE DOCUMENTS

##### Rockwell

###### General

29650N30 R6500 Programming Manual  
Order No. 202

29650N31 R6500 Hardware Manual  
Order No. 201

###### AIM 65 Microcomputer

29650N36 AIM 65 Microcomputer User's Guide  
Order No. 209

29650N72 AIM 65 FORTH User's Manual  
Order No. 265

###### AIM 65/40 Microcomputer

29650N86 AIM 65/40 Microcomputer System User's Manual  
Order No. 280

29651N07 AIM 65/40 FORTH User's Manual  
Order No. 263

###### RM 65 Modules

29801N19 RM 65 Direct Memory Access (DMA) Module  
Order No. 801 User's Manual

##### Western Digital

FD179X-02 Floppy Disk Formatter/Controller Family Data  
Sheet

WD1691 Floppy Support Logic Data

WD2143-01 Four-Phase Clock Generator Data Sheet

#### SECTION 2

##### INSTALLATION AND OPERATION

#### 2.1 UNPACKING

Unpack the FDC module from its shipping carton and refer to the packing sheet to verify that all of the parts are included. Save the packing material for storing the module.

##### CAUTION

This module contains voltage-sensitive items. The module should be stored in an anti-static container when not in use and anyone handling the unit should observe anti-static precautions. Damage to the unit may result if anti-static protection is not maintained.

#### 2.2 OPERATING OPTIONS

Seven operating options are switch, PROM, or jumper selectable:

- . Base Address selection
- . Bank selection
- . Program ROM and I/O selection
- . DMA Channel selection
- . Standard/Mini-floppy selection
- . Precompensation
- . Dual Head Drive

Figure 2-1 identifies the detail on the FDC module. The function of each switch, jumper, and connector is identified in Table 2-1, along with reference to the section and table that describes its use.

Table 2-1. FDC Module Switches, Jumpers, and Connectors

Category	Item	Description	Reference
Switches	S1-1	Bank Select Enable	Section 2.2.2 Table 2-3
	S1-2	Bank Select	Section 2.2.3 Table 2-4
	S1-3	I/O Selection	
	S1-4	Program ROM Selection	
Headers	JB1, JB2	Standard/Mini-Floppy Selection	Section 2.2.5 Table 2-6 Figure 2-2
Jumpers	E1	Precompensation Selection	Section 2.2.6 Table 2-7
	E2	DMA Channel Selection	Section 2.2.4 Table 2-5
	E3	Dual Head Drive	Section 2.2.7 Table 2-8
Connectors	P1	RM 65 Bus	Tables 3-1 & 3-3
	J1	Disk Drive I/O	Table 3-2 & 3-4
PROM	Z18	Base Address Selection	Section 2.2.1 Table 2-2

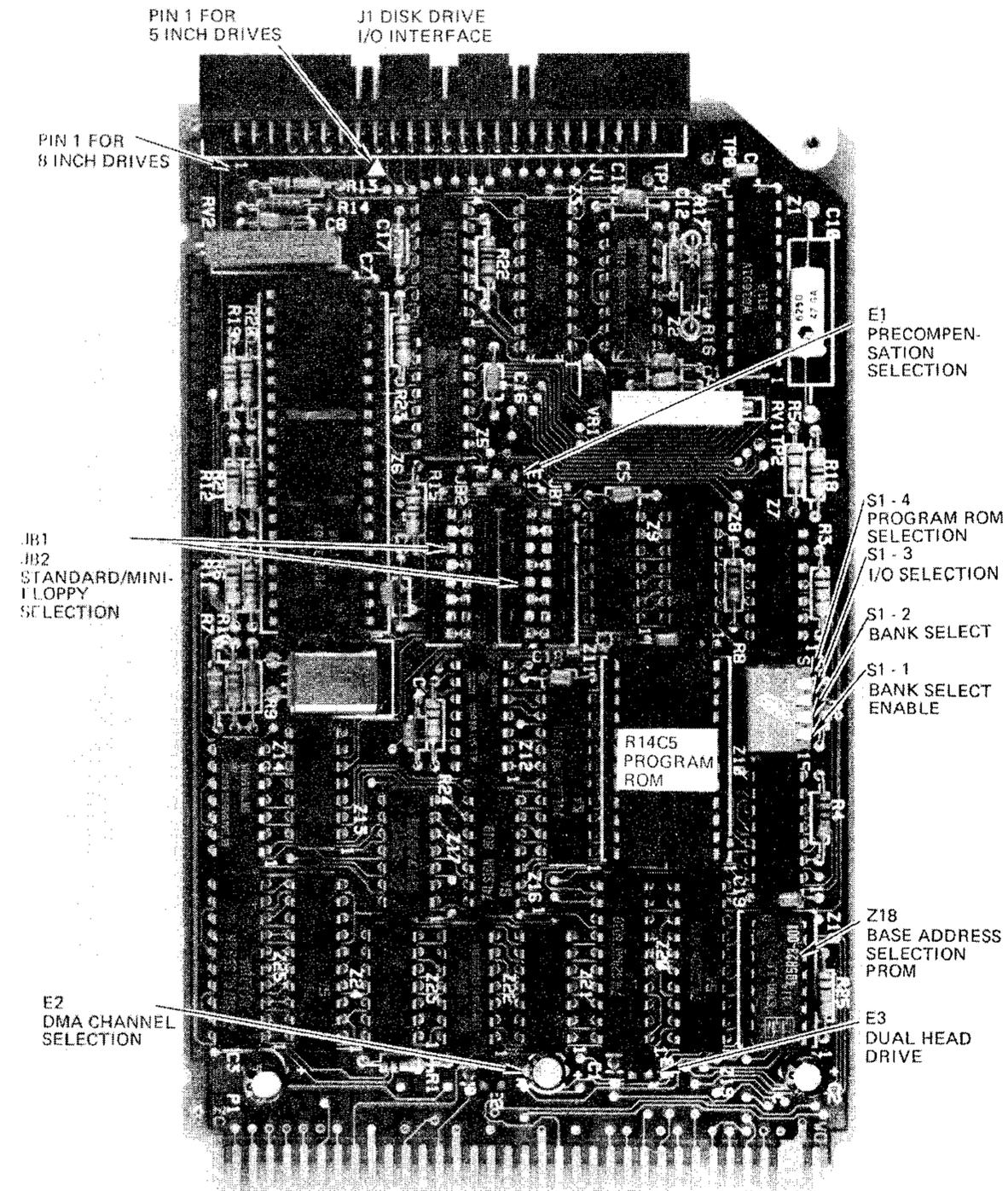


Figure 2-1. FDC Module Detail

### 2.2.1 Base Address Selection

The FDC module has a factory programmed base address. The Base Address Selection PROM (Z18) is programmed with this addressing information and cannot be altered. The programmed outputs for the Addressing PROM are shown (for reference only) in Table 2-2.

Table 2-2. Base Address Selection PROM Programmed Outputs

PROM Address Inputs								PROM Select Outputs			
H	G	F	E	D	C	B	A	D1	D2	D3	D4
0	0	X	X	X	X	X	X	1	1	X	X
0	1	0	X	X	X	X	X	1	1	X	X
0	1	1	0	X	X	X	X	1	1	X	X
0	1	1	1	0	0	0	0	0	1	X	X
0	1	1	1	0	0	0	1	1	0	X	X
0	1	1	1	0	1	X	X	1	0	X	X
0	1	1	1	1	X	X	X	1	0	X	X
1	X	X	X	X	X	X	X	1	1	X	X

NOTES

- X is a don't care level.
- PROM inputs A to H correspond to RM 65 signals BA8/ to BA15/, respectively.
- PROM outputs D1 and D2 correspond to device selects  $\overline{\text{FDC}}$  and  $\overline{\text{ROM}}$ , respectively.
- PROM outputs D3 and D4 are not used.
- This program reflects the following base address:  
 \$8000 to \$8EFF - ROM selection  
 \$8F00 to \$8FFF - I/O selection

For custom applications where the standard FDC firmware is not used, the Addressing PROM can be replaced with a user-provided PROM which has any desired address programmed. The FDC module will be active in any address range for which the  $\overline{\text{ROM}}$  or  $\overline{\text{FDC}}$  selects are programmed active. These two selects must not be programmed active in the same page (that is, for any PROM address input, only one PROM select output may be low).

### 2.2.2 Bank Selection

The Bank Select Enable switch (S1-1), in conjunction with the Bank Select switch (S1-2), allows the module to be assigned common to both memory banks (Bank 0 and Bank 1) or to be dedicated to a selected memory bank (either Bank 0 or Bank 1). When OPEN, the Bank Select Enable switch assigns the I/O logic and Program ROM to both banks regardless of the position of the Bank Select switch. When the Bank Select Enable switch is CLOSED, the assigned bank is determined by the position of the Bank Select switch. See Table 2-3 for the switch positions.

In applications where the module is to be addressed by a microcomputer that does not have bank addressing capabilities, the module must be assigned common to Bank 0 and Bank 1, or dedicated to Bank 0.

Table 2-3. Bank Select Switch Positions

Memory Bank Selected	Switch Position	
	Bank Select Enable Switch S1-1	Bank Select Switch S1-2
Bank 0 and 1	OPEN	EITHER
Bank 0 (Lower 65K)	CLOSED	OPEN
Bank 1 (Upper 65K)	CLOSED	CLOSED

### 2.2.3 Program ROM and I/O Selection

The Program ROM Selection switch (S1-4) and the I/O Selection switch (S1-3) allow the module to operate with the ROM and I/O active, with I/O only active, or with the module disabled from the bus. When only the I/O is selected, the module is active in the page assigned by the Base Address Selection PROM. When the Program ROM and I/O are both selected, the module is active in the 4K-byte block assigned by the Base Address Selection PROM with the Program ROM active in all pages except the I/O page. Both the Program ROM and the I/O can be deselected which removes the FDC module entirely from the RM 65 memory map.

To use the Program ROM firmware, i.e., primitive routines or optional DOS, the Program ROM and I/O must be selected and the module base address assigned to \$8000. The module may also be used with user-provided firmware, with the Program ROM selected installed on-board or deselected (firmware installed off-board). The switch positions are summarized in Table 2-4.

Table 2-4. Program ROM and I/O Selection Switch Positions

Module Operation and Program Location	Switch Positions	
	Program ROM Selection Switch S1-4	I/O Selection Switch S1-3
Supplied Firmware (on-board)	CLOSED	CLOSED
User Firmware (on-board)	CLOSED	CLOSED
User Firmware (off-board)	OPEN	CLOSED
Module Deselected	OPEN	OPEN

### 2.2.4 DMA Channel Selection

For most applications, all data transfers between the module and the RM 65 bus can be directly controlled by the CPU, with no need for a DMA Controller. In this case, the DMA Channel Selection jumper (E2) is not used and can be in either position. For applications that require a DMA Controller (such as 8-inch double-density format), the DMA request can be generated on either of the two RM 65 DMA Request lines (BDRQ1/, BDRQ2/). Install the DMA Channel Selection jumper (E2) in position A for DMA Request 1, or in position B for DMA Request 2 (see Table 2-5).

Table 2-5. DMA Channel Selection Jumper Positions

Module Operation	RM 65 Signal	Jumper Positions
Operation without a DMA Controller	-	E2 = EITHER
Operation using a DMA Controller		
DMA Request Channel 1	BDRQ1/	E2 = A
DMA Request Channel 2	BDRQ2/	E2 = B

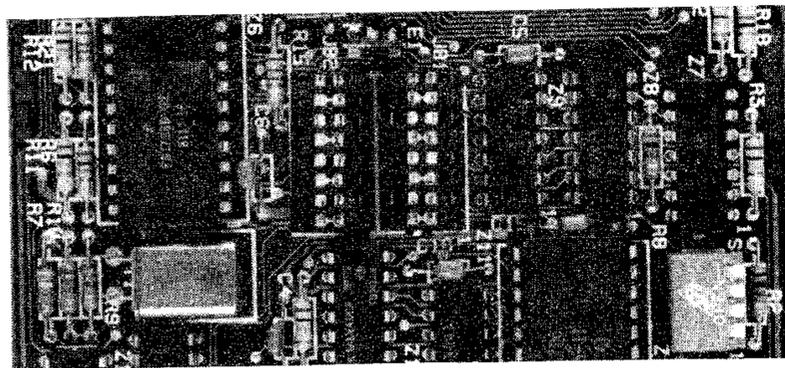
### 2.2.5 Standard/Mini-Floppy Selection

The standard/mini-floppy selection headers (JB1, JB2) configure the module to operate with standard floppy (8") or mini-floppy (5") disk drives. These headers consist of two factory programmed 16-pin shunting bars with alternate shunts removed. By rotating these headers 180° the shorted pins can be changed (refer to Figure 2-2 and Table 2-6).

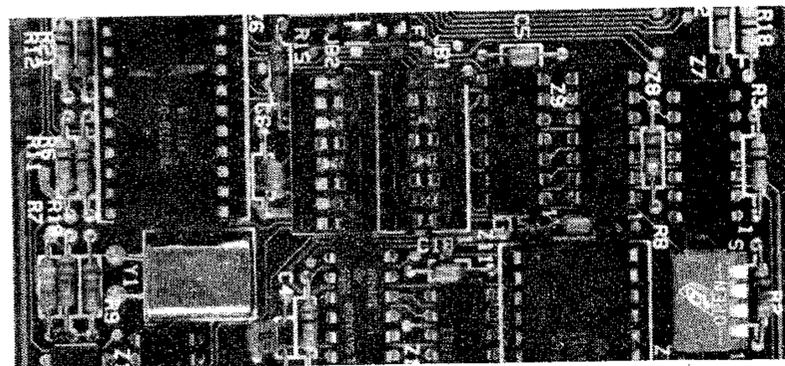
In the 5" Position (see Figure 2-2a), pins 1, 3, 5, and 7 of headers JB1 and JB2 are shunted to the opposite pins 10, 12, 14, 16, with the remaining pins left open. In this mode, the FDC module is configured for mini-floppy disk operation, with only 34 pins on the I/O connector used. Signals re-assigned on the I/O connector include Second Side Select, Drive Four Select, Index Hole, Motor On (used for Head Load in 8"

position), and Drive Ready which is forced always active (logic 1). The FDC controller clock is set to 1 MHz and the VCO reference clock is set to 2 MHz. To indicate the 5" position, bit 6 of the Drive Status Buffer is set low (logic 0).

In the 8" position (see Figure 2-2b) pins 2, 4, 6, and 8 of headers JB1 and JB2 are shunted to the opposite pins 9, 11, 13, 15, with the remaining pins left open. In this mode, the FDC module is configured for standard floppy disk operation with all 50 pins on the I/O connector used. The drive interface signals that differ from the 5" position include the 2nd side select, Drive Four select, Index Hole, Drive Ready, and Head Load (used for Motor On in 5" position). The FDC controller clock is set to 2 MHz and the VCO reference clock is set to 4 MHz. To indicate the 8" position, bit 6 of the Drive Status Buffer is set high (logic 1).



a. 5" Floppy Drive Header Placement



b. 8" Floppy Drive Header Placement

Figure 2-2. Standard/Mini-Floppy Selection Header Locations

Table 2-6. Standard/Mini-Floppy Selection Header Position

	Pins	Mini-Floppy (5") Selected		Standard Floppy (8") Selected	
		SHUNT	OPEN	SHUNT	OPEN
JB1	1, 16	SHUNT	Drive Status Buffer - logic 0 on bit 6	OPEN	Drive Status Buffer - logic 1 on bit 6
	2, 15	OPEN		SHUNT	
	3, 14	SHUNT	FDC Controller Clock - 1 MHz Reference	OPEN	FDC Controller Clock - 2 MHz Reference
	4, 13	OPEN		SHUNT	
	5, 12	SHUNT		OPEN	
	6, 11	OPEN	PLL Clock - 2 MHz Reference	SHUNT	PLL Clock - 4 MHz Reference
JB2	7, 10	SHUNT	2nd Side Select - pin 48 (32) of J1	OPEN	2nd Side Select - pin 14 of J1
	8, 9	OPEN		SHUNT	
	1, 16	SHUNT	Drive Four Select - pin 22 (6) of J1	OPEN	Drive Four Select - pin 32 of J1
	2, 15	OPEN		SHUNT	
	3, 14	SHUNT	Motor On - pin 32 (16) of J1	OPEN	Head Load - pin 18 of J1
	4, 13	OPEN		SHUNT	
	5, 12	SHUNT	Drive Ready - always active	OPEN	Drive Ready - pin 22 of J1
	6, 11	OPEN		SHUNT	
7, 10	SHUNT	Index Hole - pin 24 (8) of J1	OPEN	Index Hole - pin 20 of J1	
8, 9	OPEN		SHUNT		

NOTE

1. Pin numbers in ( ) refer to a 34-pin receptacle for 5" drives.

### 2.2.6 Precompensation

The Precompensation jumper (E1) selects the recording mode of the write circuitry (see Table 2-7). For single-density recording, place jumper E1 in position A so precompensation is not performed. When double-density recording is used, precompensation is performed in two ways. For typical applications, precompensation is not used on the outer cylinders. For this mode, install jumper E1 in the B position to precompensate only on tracks greater than 43. Alternatively, remove jumper E1 so precompensation will be performed on every track.

Table 2-7. Precompensation Jumper Positions

Recording Mode	Jumper Positions
Single-Density - No Precompensation	E1 = A
Double-Density - Precompensation on Tracks >43 - Precompensation on all Tracks	E1 = B E1 = REMOVE

### 2.2.7 Dual Head Drive

The FDC module can use either single head floppy disk drives, double head drives, or combination of both types. If only single head drives are used, set the Dual Head Drive jumper (E3) to position B. If there are any double head drives, set this jumper to position A. These positions are shown in Table 2-8.

Table 2-8. Dual Head Drive Selection Jumper Positions

Types of Floppy Drives	Jumper Positions
Single Head Drives Only	E3 = B
Single and Dual Head Drives	E3 = A

### 2.3 INSTALLING THE MODULE

Before installing the module, ensure that it is not damaged and is free of grease, dirt, liquid or other foreign matter.

#### CAUTION

Prior to module installation turn off power to the RM 65 bus. Also, turn off power to the RM 65 bus when the Floppy Disk Controller module is installed prior to changing switch or jumper positions.

- a. Based on the RM 65 system memory map and requirements (refer to Section 3.1), select the proper module operating options relating to the RM 65 bus interface:
  - (1) Select common or dedicated bank operation for the module by positioning switches S1-1 and S1-2 (refer to Section 2.2.2 and Table 2-3).
  - (2) Install the desired on-board Program ROM:

ROM ID	Description	Address Range
R14C5	FDC Primitive Routines	\$8800-\$8FFF
R324E	AIM 65 DOS 1.0	\$8000-\$8EFF
R325E	AIM 65/40 DOS 1.0	\$8000-\$8EFF
Other	User-provided	

Note that the R14C5 ROM is 2K bytes while the two DOS ROMs are 4K bytes. The factory configured base address PROM assigns address range \$8000-\$8EFFF to the on-board PROM/ROM to allow either 2K- or 4K-byte devices to be installed. Early deliveries of the R14C5 ROM code may be installed in a 4K-byte PROM.

- (3) Select or deselect the program ROM and I/O by positioning switches S1-3 and S1-4 (refer to Section 2.2.3 and Table 2-4). If the program ROM routines will be used, the ROM must be installed and selected.
  - (4) Select the DMA request channel by positioning jumper E2 (refer to Section 2.2.4 and Table 2-5)
- b. Based on the Floppy Disk system requirements, select the remaining operating options:
- (1) Set the module for 8" or 5" drives with the selection headers JB1 and JB2 (refer to Section 2.2.5, Figure 2-2, and Table 2-6).
  - (2) Select the precompensation mode by positioning jumper E1 (refer to Section 2.2.6 and Table 2-7).
  - (3) Select for single or dual headed drives by positioning jumper E3 (refer to Section 2.2.7 and Table 2-8).
- c. Align pin Wa (for Edge Connector version) or pin 1a (for Eurocard version) of the module with the identical pin on the mating RM 65 bus receptacle.

**CAUTION**

RM connectors are keyed to prevent improper module connection. If the module does not insert into the receptacle with moderate pressure applied, check the orientation and the connector alignment of the module. Forcing the module improperly into the receptacle will damage the receptacle and/or the module.

- d. Insert the module into the desired card slot (if a card cage is used) and position it in front of the mating receptacle.
- e. If an 8" disk drive is to be operated in the double-density mode, install an RM 65 DMA Controller Module (RM65-5104).
- f. Insure that all interfacing drives are properly configured (refer to their operating manuals) and apply any necessary power supplies.
- g. Connect the required cable to the Disk Drive connector J1 on the module (see Figure 2-1) and to the interfacing disk drives.
- h. Press in firmly on the end of the module until all pins are securely seated.
- i. Reapply power to the RM 65 bus.

**NOTE**

The FDC Module requires +12 VDC on the RM 65 bus (pin 17a of connector P1).

**2.4 REMOVING THE MODULE**

- a. Turn off power to the RM 65 bus and interfacing disk drives.
- b. Disconnect the cable from the disk drive connector.
- c. If the module is installed in a card cage, lift up on the module ejector tab to release the module from the mating receptacle. Pull the module straight back until it is free from the module guides.

CALIBRATION TEST FOR RM 65 FDC MODULE (CONTINUED)

- d. If the module is installed in a single card adapter, or in a motherboard without a card cage, pull back on the module while moving it slightly from side to side until it is free from the mating receptacle.

2.5 FDC MODULE CALIBRATION PROCEDURE

The FDC module has two trimmer potentiometers (RV1, RV2) which are factory adjusted to nominal values in order for the FDC module to operate with most disk drives. Under normal conditions, these settings will not require changing.

Trim pot RV1 controls the VCO reference frequency. When properly adjusted, there should be a 4.1 MHz waveform between the test point (TP1) and ground (TP0). This frequency is the same regardless of the disk drive units used.

Trim pot RV2 controls the write precompensation pulse width. If the disk drive manufacturer specifies a different value than 150 ns, the calibration procedure is as follows:

- a. Insert a blank or scratch disk into the desired disk drive 1. The drive will be selected for double-density mode, side one.
- b. Adjust RV2 to get a negative-going pulse on TP2 of the specified time period while the track is being repeatedly formatted by the following Precomp Adjust Routine:

CALIBRATION TEST FOR RM 65 FDC MODULE

SOURCE

```

DCONR=#8F04
EXEC03=#8B03
FDAR=#8F03
FORFLG=#4BB
FSTOP=#8F15
*=#0000
ENTRY LDA #FF          ;FLAG FOR IRQ HANDLER
      STA FORFLG
PRECOM LDA #E2         ;DRV. 1, SID. 1, DBL DENS, IRQ ENABLED
      STA DCONR
      JSR WRTRAK       ;WRITE A TRACK
      JMP PRECOM       ;REPEAT UNTIL RESET
    
```

SOURCE

```

WRTRAK LDA #F0         ;FDC WRITE TRACK COMMAND
      JSR EXEC03       ;EXECUTE THE FDC COMMAND
WDATA  BIT FSTOP       ;STOP CPU UNTIL DATA REQUEST
      STA FDAR        ;WRITE DUMMY BYTE FOR REQUEST
      JMP WDATA        ;LOOP UNTIL IRQ EXITS
      END
    
```

- c. When done, press RESET to return control to the Monitor command level.

## SECTION 3

### FUNCTIONAL AND INTERFACE DESCRIPTION

#### 3.1 MEMORY MAP

If the on-board Program ROM is selected (by switch S1-4, see Table 2-4), the FDC module is assigned 4096 bytes (\$8000-\$8FFF). The first 3840 bytes (\$8000-\$8EFF) are available to the Program ROM while the last 256 bytes (\$8F00-\$8FFF) are dedicated to the I/O logic. Of these 256 bytes, only six are unique and are assigned to the controller device or module logic; the other 250 bytes are redundant, i.e., copies of these six bytes. Table 3-1 shows the general FDC Module Memory Map and Table 3-2 lists the FDC module detailed I/O functions.

The primitive subroutines and variables are identified in the FDC Primitive ROM (R323E), the AIM 65 DOS 1.0 ROM (R324E) and the AIM 65/40 DOS 1.0 ROM (R325E). If any of these ROMs are installed, the primitives require 90 bytes of off-board RAM; four bytes on page zero and 86 bytes on page four (see Table 4-7 for the variable address assignments). The primitives also require two RAM buffers for data input/output, each of which must be at least one sector in length (128 bytes for single-density or 256 bytes for double-density). See Section 4 for the description of the primitive routines. If the optional DOS 1.0 ROM is installed, it requires four additional bytes of RAM on page zero and 100 bytes on page five (see Tables 5-4 and 6-4 for AIM 65 and AIM 65/40 variable addresses, respectively). The default I/O buffer locations for the DOS are \$600-\$7FF for the AIM 65 and \$3E00-\$3FFF for the AIM 65/40 (these locations are user-alterable, see Tables 5-4 and 6-4). See Sections 5 and 6 for descriptions of the AIM 65 and AIM 65/40 DOS 1.0, respectively.

If the on-board Program ROM is not selected, the constraints on the base address (i.e., \$8000-\$8FFF) do not apply and the I/O logic may be assigned to any available page (see Section 2.2.1).

Table 3-1. FDC Module Memory Map

Hex Addr.	Function	Hex Addr.	Function
\$0000	Page Zero RAM: \$0D7 - \$0DE Page Four RAM: \$4A0 - \$4FA Page Five RAM: \$500 - \$563 (Page Five for DOS use only)	\$8000	15 Pages ROM: \$8000 - \$8EFF 1 Page I/O: \$8F00 - \$8FFF
\$1000	User Available	\$9000	User Available
\$2000			
\$3000			
\$4000			
\$5000			
\$6000			
\$7000			
NOTES			
1. Base Address Selection PROM is factory programmed to \$8000.			
2. RAM is not on the FDC module.			
3. The DMAC module (optional) can be assigned to any available page.			

Table 3-2. FDC Module I/O Memory Map

Hex Addr.	Buffer/Register	
	Write (R/ $\bar{W}$ = Low)	Read (R/ $\bar{W}$ = High)
XY00	Command Register	Status Register
XY01	Track Register	Track Register
XY02	Sector Register	Sector Register
XY03	Data Register	Data Register
XY04	Drive Control Register	Drive Status Register
XY15	- - -	Force a not ready to stop the CPU until the FDC device is ready.
NOTES		
1. XY corresponds to the assigned Base Address of any two hexadecimal values. For the standard FDC, XY=8F.		
2. The remaining locations in the I/O page are redundant.		
3. The Drive Control Register is as follows:		
	<u>Bit</u>	<u>Meaning</u>
	0	Side Select: 0 = Side 1, 1 = Side 2
	1	Drive 1 Select: 1 = On
	2	Drive 2 Select: 1 = On
	3	Drive 3 Select: 1 = On
	4	Drive 4 Select: 1 = On
	5	5" Motor On/8" Head Load: 1 = Active
	6	STOPEN: 1 = Generate not ready, Clear IRQ
	7	Density: 0 = Double, 1 = Single
4. The Drive Status Register is as follows:		
	<u>Bit</u>	<u>Meaning</u>
	0	Selected Side: 0 = Side 2, 1 = Side 1
	1	- - -
	2	- - -
	3	- - -
	4	- - -
	5	Number of Heads: 0 = One, 1 = Two
	6	Selected Size: 0 = 8", 1 = 5"
	7	Selected Density: 0 = Single, 1 = Double

### 3.2 FUNCTIONAL DESCRIPTION

The block diagram in Figure 3-1 identifies the FDC module functions and interface signals.

The Controller Clock derives a reference frequency for the FDC device from a crystal-controlled oscillator. This reference frequency is 1 MHz or 2 MHz, depending on the Standard/Mini-Floppy Selection Header position.

The FDC device, in conjunction with the Data Separator and Precompensation Circuitry, interfaces the RM 65 bus to the floppy disk medium. The circuitry supports 5-1/4" or 8" disks, with choice of single-density (FM) or double-density (MFM) soft sector formats. The FDC features powerful commands, including read/write head movements, reading and writing of data, and reading and writing of track format, with selectable record lengths. Write precompensation circuitry ensures reliable data recovery in double density formats. The Precompensation jumper (E1) selects precompensation on all tracks, only on tracks greater than 43, or no precompensation at all.

The Standard/Mini-Floppy Selection header (JB1, JB2) selects the Disk Drive connector (J1) and FDC circuitry for either 5" mini-floppy or 8" standard floppy disk formats. The 50-pin I/O receptacle connects the FDC module to a mass terminated cable connected to the installed disk drives. A 34-pin cable and mating connector can be used to connect the 5" mini-floppy drives while a 50-pin cable and mating connector is needed to connect to the 8" floppy drives.

The Drive Status Buffer allows detection of the Standard/Mini-Floppy Selection header and Dual Head Drive jumper (E3) positions, as well as selected density and side information.

The Drive Control Register provides control of the side and drive selection, motor on (5" only) or head load (8" only), the recording density, and the interrupt request disable.

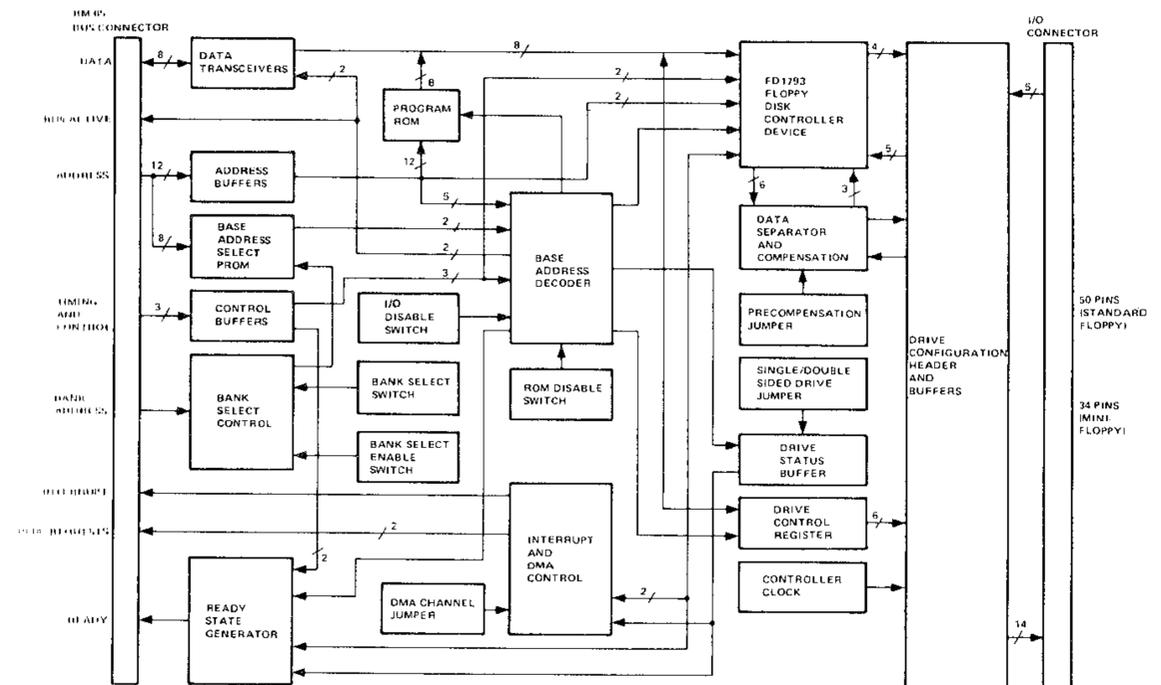


Figure 3-1. FDC Module Block Diagram

The Interrupt and DMA control circuit allows operation in either an interrupt-driven mode or under DMA control. An interrupt request latch holds all interrupt requests until serviced and cleared (STOPEN). DMA requests from the FDC module can be driven on either of two DMA request channels (BDRQ1/, BDRQ2/), as selected by the DMA Channel Selection jumper (E2), or disabled by removing the jumper.

The Ready State Generator provides wait states as required by the FDC device.

The Base Address Decoder, with the Base Address Select PROM (Z18), the Bank Select Control circuit, the ROM and I/O Disable switches (S1-2, S1-4), and the phase 2 and read/write signals control device selection on the module. The Base Address Select PROM compares the eight most significant address lines to the programmed addresses to generate device select signals to the Program ROM and the I/O devices. The ROM Disable switch assigns the module to be active either in a 256-byte page (disabled) or in a 4K-byte block (enabled). The I/O Disable switch allows the FDC I/O to be disabled.

When the ROM is disabled, only the I/O devices are active in the 256-byte page that matches all eight Base Address Select bits. For the I/O devices, the three least significant address lines, along with the phase 2 clock and read/write control signals, drive register select lines to the FDC device, and device select lines to the Drive Status Buffer and Drive Control Register.

When the ROM is enabled, the module is active in the 4K byte block that matches the four most significant Base Address Select bits. The program ROM is selected except when the address matches the four least significant Base Address Select bits, in which case the I/O device select lines are selected.

The Bank Select Control circuit detects when the module's assigned memory bank is addressed by comparing the bank address signal (BADR/) from the RM 65 bus to the Bank Select (S1-2) and Bank Select Enable (S1-1) switches. The Bank Select Enable switch allows the board to reside in common memory (both Bank 0 and Bank 1) or only in the bank set by the Bank Select switch (either Bank 0 or Bank 1).

The Control Buffers invert and transfer phase 2 clock (B02/), reset (BRES/), and read/write (BR/W/) control signals from the RM 65 bus onto the module. The interrupt request (BIRQ/) is buffered and driven onto the RM 65 bus.

The Data Transceivers invert and buffer 8-bits of parallel data (BD0/-BD7/) between the module and the RM 65 bus based on control signals from the Base Address Decoder and the Control Buffers. Data to the module is latched (02) by the write data latch, extending the data hold time for the FDC device. Data from the module is buffered by the read data buffers. The transceivers are enabled by the Base Address Decoder when the module is addressed.

The Address Buffers invert and transfer 12 of the 16 parallel address bits (BA0/-BA11/) from the RM 65 bus to the Base Address Decoder (A0-A4), to the Program ROM (A0-A11), and to the FDC device (A0, A1).

### 3.3 INTERFACE DESCRIPTION

Tables 3-3 and 3-4 list the interface signals and pin assignments for connector P1 (RM 65 Bus) and connector J1 (Disk Drive), respectively.

Tables 3-5 and 3-6 define the interface signals for connectors P1 and J1, respectively.

Table 3-3. Connector P1 (RM 65 Bus) Pin Assignments

Pin	Signal Mnemonic	Signal Name	Input/Output
Wa		Not Connected (See Note)	
Wc		Not Connected (See Note)	
Xa	+5V	+5 VDC (See Note)	
Xc	+5V	+5 VDC (See Note)	
1a	GND	Ground	
1c	+5V	+5 VDC	
2a	BADR/	Buffered Bank Address	I
2c	BA15/	Buffered Address Bit 15	I
3a	GND	Ground	
3c	BA14/	Buffered Address Bit 14	I
4a	BA13/	Buffered Address Bit 13	I
4c	BA12/	Buffered Address Bit 12	I
5a	BA11/	Buffered Address Bit 11	I
5c	GND	Ground	
6a	BA10/	Buffered Address Bit 10	I
6c	BA9/	Buffered Address Bit 9	I
7a	BA8/	Buffered Address Bit 8	I
7c	BA7/	Buffered Address Bit 7	I
8a	GND	Ground	
8c	BA6/	Buffered Address Bit 6	I
9a	BA5/	Buffered Address Bit 5	I
9c	BA4/	Buffered Address Bit 4	I
10a	BA3/	Buffered Address Bit 3	I
10c	GND	Ground	
11a	BA2/	Buffered Address Bit 2	I
11c	BA1/	Buffered Address Bit 1	I
12a	BA0/	Buffered Address Bit 0	I
12c	B01	Not Used	
13a	GND	Ground	
13c	BSYNC	Not Used	
14a	BSD	Not Used	
14c	BDRQ1	Buffered DMA Request 1	O
15a	BRDY	Buffered Ready	O

Table 3-3. Connector P1 (RM 65 Bus) Pin Assignments (Cont'd)

Pin	Signal Mnemonic	Signal Name	Input/Output
15c	GND	Ground	
16a		Not Used	
16c		Not Used	
17a	+12V	+12V	
17c		Not Used	
18a	GND	Ground	
18c		Not Used	
19a	BFLT/	Not Used	
19c	B00	Not Used	
20a		Not Used	
20c	GND	Ground	
21a	BR/W/	Buffered Read/Write "Not"	I
21c	BDRQ2/	Buffered DMA Request 2	O
22a		Not Used	
22c	BR/W	Not Used	
23a	GND	Ground	
23c	BACT/	Buffered Bus Active	O
24a	BIRQ/	Buffered Interrupt Request	O
24c	BNMI/	Not Used	
25a	B02/	Buffered Phase 2 "Not" Clock	I
25c	GND	Ground	
26a	B02	Not Used	
26c	BRES/	Buffered Reset	I
27a	BD7/	Buffered Data Bit 7	I/O
27c	BD6/	Buffered Data Bit 6	I/O
28a	GND	Ground	
28c	BD5/	Buffered Data Bit 5	I/O
29a	BD4/	Buffered Data Bit 4	I/O
29c	BD3/	Buffered Data Bit 3	I/O
30a	BD2/	Buffered Data Bit 2	I/O
30c	GND	Ground	
31a	BD1/	Buffered Data Bit 1	I/O

Table 3-3. Connector P1 (RM 65 Bus) Pin Assignments (Cont'd)

Pin	Signal Mnemonic	Signal Name	Input/Output
31c	BD0/	Buffered Data Bit 0	I/O
32a	+5V	+5 VDC	
32c	GND	Ground	
Ya	+5V	+5 VDC (See Note)	
Yc	+5V	+5 VDC (See Note)	
Za		Not Connected (See Note)	
Zc		Not Connected (See Note)	

NOTES

1. Pins Wa, Wc, Xa, Xc, Ya, Yc, Za and Zc are available on Edge Connector version only.
2. "/" suffix denotes signal active at negative or low voltage level.

Table 3-4. Connector J1 (Disk Drive) Pin Assignments

FDC Module I/O Connector Pin	Standard Floppy Disk Drive Interface Cable Connector		Mini-Floppy Disk Drive Interface Cable Connector (2)	
	Pin	Signal Name	Pin	Signal Name
2	2	Track >43 (Remex & MFE or equivalents)		
4	4	N.C.		
6	6	N.C.		
8	8	Track >43 (C-disk or equivalents)		
10	10	N.C.		
12	12	N.C.		
14	14	2-J Side Select		
16	16	N.C.		
18	18	Head Load	2	N.C.
20	20	Index	4	N.C.
22	22	Drive Ready	6	Drive Select #4
24	24	N.C.	8	Index
26	26	Drive Select #1	10	Drive Select #1
28	28	Drive Select #2	12	Drive Select #2
30	30	Drive Select #3	14	Drive Select #3
32	32	Drive Select #4	16	Motor On
34	34	Direction In	18	Direction In
36	36	Step Pulse	20	Step Pulse
38	38	Write Data	22	Write Data
40	40	Write Gate	24	Write Gate
42	42	Track Zero	26	Track Zero
44	44	Write Protected	28	Write Protected
46	46	Read Data	30	Read Data
48	48	N.C.	32	2nd Side Select
50	50	N.C.	34	N.C.

NOTES:

1. All odd numbered pins are GND.
2. Pin 1 of the 34-pin mini-floppy disk drive interface cable connector should be keyed to pin 17 of the FDC module I/O connector.

Table 3-5. Connector P1 (RM 65 Bus) Signal Descriptions

Mnemonic	Signal Name and Signal Description
+5V	+5 VDC supplied to the module from the RM 65 Bus.
+12V	+12 VDC supplied to the module from the Bus.
GND	<u>Ground</u> System ground.
BA0/-BA15/	<u>Buffered Address Bits 0-15</u> Sixteen address lines transfer an inverted 16-bit parallel address from the Bus to the module.
BADR/	<u>Buffered Bank Address</u> A high BADR/ signal addresses the lower 65K (Bank 0) memory bank; a low BADR/ addresses the upper 65K (Bank 1) memory bank.
BD0/-BD7/	<u>Buffered Data Bits 0-7</u> Eight bidirectional inverted data lines transfer 8-bit data bytes between the Data Transceivers in the module and the Bus.
BACT/	<u>Buffered Bus Active</u> A low BACT/ indicates that the module has been addressed and the Data Transceivers are enabled in either the receive (write operation) or transmit (read operation) direction.
B02/	<u>Buffered Phase 2 Clock "NOT"</u> The B02/ signal synchronizes data transfers on the Bus. The address and read/write lines are setup in the positive portion of B02/. The data lines are set-up in the negative portion of B02/.

Table 3-5. Connector P1 (RM 65 Bus) Signal Descriptions (Cont'd)

Mnemonic	Signal Name and Signal Description
BR/ $\bar{W}$ /	<u>Buffered Read/Write "Not"</u> The BR/ $\bar{W}$ / signal controls the direction of data transfer on the Bus. A low BR/ $\bar{W}$ / indicates a read operation. A high BR/ $\bar{W}$ / indicates a write operation.
BRDY	<u>Buffered Ready</u> The BRDY signal is generated by the module. When the R6502 CPU receives a low BRDY, the CPU will stop execution in the next read cycle. Execution will resume when BRDY returns high.
BIRQ/	<u>Buffered Interrupt Request</u> The BIRQ/ signal is generated by the module for the Bus to request interrupt service. BIRQ/ is forced low by any interrupt condition in the FDC device.
BDRQ1/ BDRQ2/	<u>Buffered DMA Requests 1 and 2</u> Either BDRQ1/ or BDRQ2/ can be assigned to the module to request DMA service. When the FDC is ready for a data byte transfer, a DMA request is generated.
BRES/	<u>Buffered Reset</u> The BRES/ signal is received by the module from the Bus. A low BRES/ clears the Drive Status Register and the Wait State Generator, and resets the FDC device.
NOTE	
All signals interfaced to and from the module are driven at TTL voltage levels.	

Table 3-6. Connector J1 (Disk Drive) Signal Descriptions

Signal Name and Signal Description	Signal Source
<p><u>Drive Select No. 1 to No. 4</u> These signals allow up to four disk drives to be individually selected by the FDC when active.</p>	FDC
<p><u>Second Side Select</u> This signal is used by the FDC to access side 1 when inactive and side 2 when active.</p>	FDC
<p><u>Index</u> This signal is pulsed (logic 0) by the selected disk drive when the one disk medium is at the start of a revolution.</p>	Drive
<p><u>Read Data</u> This signal is the serial NRZ data from the selected disk drive medium (FM if single-density, MFM if double-density).</p>	Drive
<p><u>Write Data</u> This signal is the serial NRZ data from the FDC to be written to the selected disk drive medium (FM is single-density, MFM if double-density).</p>	FDC
<p><u>Write Gate</u> This signal, when made active by the FDC, indicates that the Write Data line is valid. When inactive, the FDC is in the read mode.</p>	FDC
<p><u>Track Zero</u> This signal is made active by the selected disk drive when the read/write head is on the outermost track.</p>	Drive

Table 3-6. Connector J1 (Disk Drive) Signal Descriptions (Cont'd)

Signal Name and Signal Description	Signal Source
<p><u>Direction In</u> This signal is made active by the FDC when the read/write head of the selected disk drive is to be moved toward an inner (higher number) track. When inactive, movement will be toward an outer (lower number) track.</p>	FDC
<p><u>Step Pulse</u> This signal is pulsed (logic 0) by the FDC to move the read/write head of the selected floppy drive in the direction set by the Direction In signal. The head moves one track position on each negative transition.</p>	FDC
<p><u>Write Protected</u> This signal is made active by the selected disk drive to indicate that the medium is write protected. For 5" drives, this means the write protect tab is on the disk; for 8" drives, this means the write protect notch is not covered.</p>	Drive
<p><u>Motor On</u> (5" Drive Only) This signal, when made active by the FDC, turns on the motor of the selected disk drive.</p>	FDC
<p><u>Head Load</u> (8" Drive Only) This signal, when made active by the FDC, lowers the read/write head of the selected disk drive onto the medium.</p>	FDC

Table 3-6. Connector J1 (Disk Drive) Signal Descriptions (Cont'd)

Signal Name and Signal Description	Signal Source
<p><u>Drive Ready</u> (8" Drive Only)</p> <p>This signal is made active by the selected disk drive to indicate that it is ready for data transfers.</p>	Drive
<p><u>Track &gt;43</u> (8" Drive Only)</p> <p>This signal is made active by the selected disk drive when the read/write head is positioned on an inner track (track 44 and higher).</p>	Drive
<p>NOTE</p> <p>All signals interfaced to and from the module are driven at standard TTL voltage levels. An active signal (logic 0) is an electrical low while an inactive (logic 1) is an electrical high.</p>	

### 3.3.1 Mini-Floppy Disk Drive Interface Cable Assembly

The mating ribbon cable connector to the Disk Drive connector (J1) must be a 50-pin mass terminating receptacle (3M No. 3425-6000 or equivalent). For interfacing with industry standard 5" mini-floppy disk drives, this connector must be attached to a 34 wire ribbon cable, displaced such that wire 1 mates to pin 17, and wire 34 mates to pin 50.

The mini-floppy disk drives (Shugart SA-450 or equivalent) have a 34-pin edge connector for the signal interface, with a pin assignment shown in Table 3-6. Up to four 34-pin mass terminating card edge receptacles (3M No. 3463-0001 or equivalent) may be attached to the ribbon cable, one connector for each drive used. The cable length should be less than four feet, however, spacing between connectors is not critical. Figure 3-2 shows the typical connector placement.

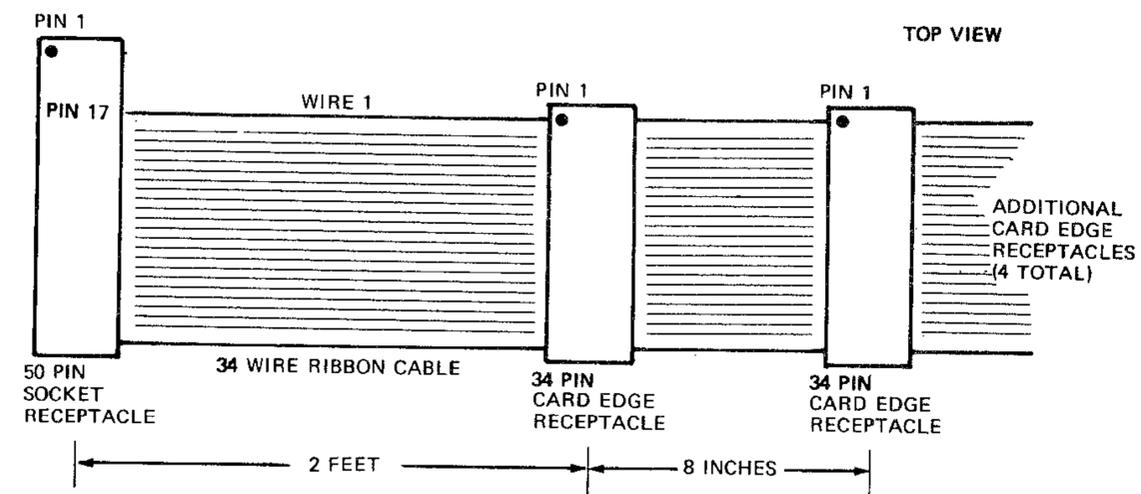


Figure 3-2. Mini-Floppy Disk Drive Interface Cable

### 3.3.2 Standard Floppy Disk Drive Interface Cable Assembly

The mating ribbon cable connector to the Disk Drive connector (J1) must be a 50-pin mass terminating receptacle (3M No. 3425-6000 or equivalent). For interfacing with industry standard 8" floppy disk drives, this connector must be attached to a 50 wire ribbon cable with wire 1 mating to pin 1.

The standard floppy disk drives (Shugart SA-850 or equivalent) have a 50-pin card edge connector for the signal interface, with a pin assignment shown in Table 3-6. One 50-pin mass terminating card edge receptacle (3M No. 3415-0001 or equivalent) must be attached to the ribbon cable for each drive used (maximum of four). The cable length should be less than four feet, however, spacing between connectors is not critical. Figure 3-3 shows the typical connector placement.

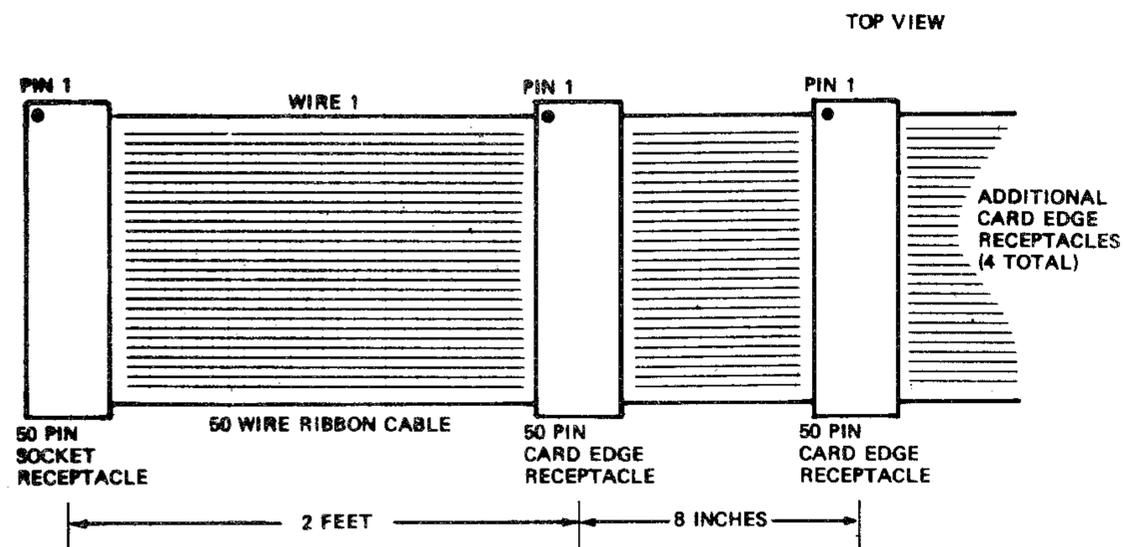


Figure 3-3. Standard Floppy Disk Drive Interface Cable

## SECTION 4

### FDC MODULE PRIMITIVE ROUTINES

#### 4.1 PRIMITIVE ROUTINE DESCRIPTIONS

The FDC primitive routines simplify the operation of the FDC module by:

- Handling the detail protocol to interface with the FD 1793 Floppy Disk Controller device.
- Providing a set of closed subroutines that initialize and format a disk, position drive heads, write data on a disk, and read data from a disk.

These primitives allow additional higher level functions, such as a disk operating system or specialized data recording, to be easily implemented without being concerned with the detailed operation of the controller device. The optional AIM 65 DOS 1.0 and AIM 65/40 DOS 1.0 functions described in Section 5 and 6, respectively, use these primitive routines. If you are using one of these DOS versions, skip to the appropriate section. If you are designing a DOS or other specialized disk interface/file handling function using the primitive routines, the detailed description of these routines included in this section will be helpful.

The primitive routines perform the following major functions to support operation of up to four disk drives (single- or double-sided; single- or double-density):

- Format a Disk
- Read or Write a Sector
- Seek or Verify Seek of a Track
- Reset or Re-zero the Head
- Read or Write Multiple Sectors
- Read or Write a Track
- Turn Motors On or Off
- Select or De-select any Drive

### 3.3.2 Standard Floppy Disk Drive Interface Cable Assembly

The mating ribbon cable connector to the Disk Drive connector (J1) must be a 50-pin mass terminating receptacle (3M No. 3425-6000 or equivalent). For interfacing with industry standard 8" floppy disk drives, this connector must be attached to a 50 wire ribbon cable with wire 1 mating to pin 1.

The standard floppy disk drives (Shugart SA-850 or equivalent) have a 50-pin card edge connector for the signal interface, with a pin assignment shown in Table 3-6. One 50-pin mass terminating card edge receptacle (3M No. 3415-0001 or equivalent) must be attached to the ribbon cable for each drive used (maximum of four). The cable length should be less than four feet, however, spacing between connectors is not critical. Figure 3-3 shows the typical connector placement.

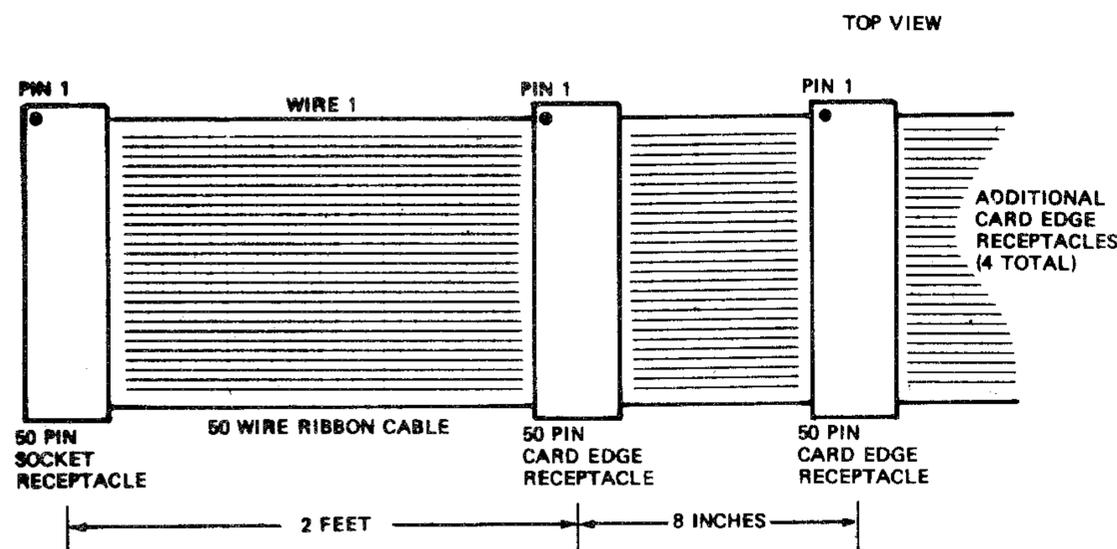


Figure 3-3. Standard Floppy Disk Drive Interface Cable

## SECTION 4

### FDC MODULE PRIMITIVE ROUTINES

#### 4.1 PRIMITIVE ROUTINE DESCRIPTIONS

The FDC primitive routines simplify the operation of the FDC module by:

- . Handling the detail protocol to interface with the FD 1793 Floppy Disk Controller device.
- . Providing a set of closed subroutines that initialize and format a disk, position drive heads, write data on a disk, and read data from a disk.

These primitives allow additional higher level functions, such as a disk operating system or specialized data recording, to be easily implemented without being concerned with the detailed operation of the controller device. The optional AIM 65 DOS 1.0 and AIM 65/40 DOS 1.0 functions described in Section 5 and 6, respectively, use these primitive routines. If you are using one of these DOS versions, skip to the appropriate section. If you are designing a DOS or other specialized disk interface/file handling function using the primitive routines, the detailed description of these routines included in this section will be helpful.

The primitive routines perform the following major functions to support operation of up to four disk drives (single- or double-sided; single- or double-density):

- Format a Disk
- Read or Write a Sector
- Seek or Verify Seek of a Track
- Reset or Re-zero the Head
- Read or Write Multiple Sectors
- Read or Write a Track
- Turn Motors On or Off
- Select or De-select any Drive

These functions are implemented in 15 major primitive routines. Each of these routines can be classified into one of four general categories:

- INITIALIZE - These initialize the FDC module or prepare a disk for use.
- DRIVE - These control the selection and read/write head movement of the floppy disk drives.
- READ - These support reading data from the floppy disk medium.
- WRITE - These support writing data onto the floppy disk medium.

The primitive routines are identified along with their entry points in Table 4-1 (in alphabetical order of their labels). The labels and addresses of the I/O registers are listed in Table 4-2.

The primitive routines are described in Table 4-3. The entry and exit conditions are specified along with the processing performed.

For the primitive routines, the following conditions apply:

- Drive number: 0 to 3.
- Side number: 0 for side one, 1 for side two.
- Density: 1 for single, 0 for double.
- Track number: 0 to 34 (5") or 0 to 76 (8").
- Sector number: 1 to 16 (5") or 1 to 26 (8").

**NOTE**

There is no error checking done on the entry parameters of the primitive routines. This is the responsibility of the user in the disk operating software.

Table 4-1. Primitive Routine Addresses

Label	Address	Function
DESEL	\$8CE2	Deselects drive
FORCOF	\$8DB9	Cancel any command in progress
FORMAT	\$890D	Formats a side (with interleaving)
INIT	\$886C	Initializes variables and buffers
IRQHAN	\$8BED	FDC Interrupt Handler
MOTOFF	\$8CF7	Turns motor off
MOTON	\$8C53	Sets motor on, selects drive, side and density
RDMSC	\$8D8D	Reads multiple sectors within a track
RDSEC	\$8D29	Reads a sector of data
RDTRK	\$8CFB	Reads entire track (sector headers also)
SEEK	\$8938	Seek with verify to a track
SELECT	\$8CBC	Selects drive, side and density (only one at a time)
WRTMSC	\$8DEA	Writes multiple sectors within a track
WRTRK	\$8994	Writes an IBM formatted track
WRTSEC	\$8D61	Writes a sector of data

Table 4-2. FDC Module I/O Register Addresses

Label	Address	Description
FCOMR	\$8F00	Command register (Write only)
FSTAR	\$8F00	Status register (Read only)
FCYLR	\$8F01	Track register (Read or Write)
FSECR	\$8F02	Sector register (Read or Write)
FDAR	\$8F03	Data register (Read or Write)
DSTAR	\$8F04	Drive Status register (Read only)
DCONR	\$8F05	Drive Control register (Write only)
FSTOP	\$8F15	Stop CPU (Write only)

Table 4-3. Primitive Routine Descriptions

Routine	Description	Category
DESEL		DRIVE
	Deselects all disk drives. Any selected drive is deselected, with Drive Select No. 1 to No. 4 made inactive. The Motor On/Head Load line is left active.  On entry, nothing is required.  On exit, nothing is saved.	
FORCOF		INITIALIZE
	Stops execution of any FDC command in progress by forcing an interrupt command to the FDC device. Also clears the DMA command register (if DMAFLG is enabled) and disables the IRQ latch. This routine should always be called after a hardware reset. Typically, this will be part of the warm RESET processing.  On entry, nothing is required. No drive needs to be active.  On exit, any errors (type I, if no command was in progress) are indicated in the A register or STFLG.	
FORMAT		INITIALIZE
	Formats the disk in the selected drive. In single-density mode, FM encoding (IBM 3740) is used with 128 data bytes per sector. For double-density, MFM encoding (System 34) is used with 256 data bytes per sector. In both modes, every sector data field is filled with the character in PADB (default is \$E5). For 5" mode, the disk is formatted with CYL5 tracks, SECT5 sectors per track, and sector interleaving as specified in TABLE5 area. For 8" mode, the disk is formatted with CYL8	

Table 4-3. Primitive Routine Descriptions (Cont'd)

Routine	Description	Category
FORMAT (Cont'd)		INITIALIZE
	tracks, SECTK8 sectors per track, and sector interleaving as specified in TABLE8 area. The read/write head is left on the inner-most track. Refer to Section 4.4.1 for default disk formats.  On entry, the drive must be active (MOTON/SELECT).  On exit, any errors (the error type depends on the FDC command that was executing when the error occurred) are indicated in the A register or STFLG.	
INIT		INITIALIZE
	Initializes the RM 65 system for the FDC module. Sets up all default values for the FDC variables (see Table 4-7). This must always precede any other FDC firmware routines. Typically, this will be part of the cold RESET processing.  On entry, nothing is required.  On exit, nothing is saved.	
IRQHAN		INITIALIZE
	Interrupt handling routine that services all RM 65 FDC interrupts. The system interrupt vector must be set up with IRQHAN before the FDC firmware routines are used. Typically, this would be set-up in the cold RESET processing. If an additional interrupt service routine is to follow, IRQOUT must point to it. Since this is an interrupt handler, it is <u>never called directly</u> by a user program, but is used by most of the firmware routines.	

Table 4-3. Primitive Routine Descriptions (Cont'd)

Routine	Description	Category
MOTOFF		DRIVE
	Turns off all drive motors (5")/unloads the read/write heads (8") and deselects all disk drives. Any selected drive is deselected, with Drive Select No. 1 to No. 4 made inactive. The drive motor flag MOFLG is reset and the Motor On/Head Load line is made inactive.	
	On entry, nothing is required.	
	On exit, nothing is saved.	
MOTON		DRIVE
	Turns on all drive motors(5")/loads the read/write heads (8") and selects the desired disk drive. Any selected drive is deselected then the desired Drive No. 1 to No. 4 is made active (uses SELECT), the drive motor flag MOFLG is set, and the Motor On/Head Load line is made active. The number of heads selected by the Single/Double Sided Drive jumper is set in NHEAD.	
	On Entry, the drive (0-3) is in A, the side (0,1) is in Y, and the density (1 for single, 0 for double) is in X.	
	On exit, nothing is saved.	

Table 4-3. Primitive Routine Descriptions (Cont'd)

Routine	Description	Category
RDMSC		READ
	Reads multiple sectors of data from the present track of selected disk drive. Any number of sequential sectors on a given cylinder may be read. The read buffer (RDBUF) must be as large as the number of sectors being read.	
	RDMSC will use the DMA module if DMAFLG is enabled; the Source Bank must correspond to the FDC module, while the Destination Bank is the RDBUF RAM buffer.	
	On entry, the drive must be active (MOTON/SELECT) with the read/write head positioned on the proper cylinder (SEEK). The starting sector is in A and the last sector is in X.	
	On exit, the data read is pointed to by RDBUF. Any errors (type II) will be indicated in the A register or STFLG.	
RDSEC		READ
	Reads the desired sector of data from the present track of the selected disk drive. The read buffer (RDBUF) must be as large as the sector size.	
	RDSEC will use the DMA module if DMAFLG is enabled; the Source Bank must correspond to the FDC module, while the Destination Bank is the RDBUF RAM buffer.	
	On entry, the drive must be active (MOTON/SELECT) with the read/write head positioned on the proper track (SEEK). The sector number is in A.	
	On exit, the data read is pointed to by RDBUF. Any errors (type II) will be indicated in A register or STFLG.	

Table 4-3. Primitive Routines Description (Cont'd)

Routine	Description	Category
RDTRK	<p>Reads the present track from the selected disk drive. This reads the entire track -- including data, clock, address marks and gaps--starting and ending at the index hole. The read buffer (RDBUF) must be large enough for the entire track.</p> <p>RDTRK will use the DMA module if DMAFLG is enabled; the Source Bank must correspond to the FDC module, while the Destination Bank is the RDBUF RAM buffer.</p> <p>On entry, the drive must be active (MOTON/SELECT) with the read/write head positioned on the proper cylinder (SEEK).</p> <p>On exit, everything read from the track is left in the buffer pointed to by RDBUF. Any errors (type III) will be indicated in the A register or STFLG.</p>	READ
SEEK	<p>Seeks to the specified track of the selected disk drive and verifies that the track was reached. The selected disk drive is restored (or re-zeroed) by a SEEK to track 0. This recalibrates the disk drive read/write head back out to track 00.</p> <p>On entry, the drive must be active (MOTON/SELECT). The track number to seek is in the A register.</p> <p>On exit, any errors (type I) are indicated in the A register or STFLG.</p>	DRIVE

Table 4-3. Primitive Routines Description (Cont'd)

Routine	Description	Category
SELECT	<p>Selects the desired disk drive. The drive is made active and the read/write head is loaded (8" drives only). Only one drive may be active (by MOTON or SELECT) at a time, so other drives will first be deselected (DESEL) automatically.</p> <p>On entry, the drive (0-3) is in A, the side (0 = one, 1 = two) is in Y, and the density (0 = double, 1 = single) is in X. The drive motors must be on (MOTON).</p> <p>On exit, if the motors are not on, an error is indicated by an \$80 in the A register or STFLG.</p>	DRIVE
WRTMSC	<p>Writes multiple sectors of data to the present track of the selected disk drive. Any number of sequential sectors on a given track may be written. The write buffer (WRBUF) must be filled with the data for all the sectors.</p> <p>WRTMSC will use the DMA module if DMAFLG is enabled. In this case, Source Bank must correspond to the WRBUF RAM buffer, while the Destination Bank is the FDC module.</p> <p>On entry, the drive must be active (MOTON/SELECT) with the read/write head positioned on the proper track (SEEK). The start sector number is in A and the last sector number is in X. The data written is pointed to by WRBUF.</p> <p>On exit, any errors (type II) are indicated in the A register or STFLG.</p>	WRITE

Table 4-3. Primitive Routine Descriptions (Cont'd)

Routine	Description	Category
WRTRK		WRITE
	<p>Writes an IBM compatible track onto the selected disk drive. This writes the entire track--including data, clock, address marks and gaps--starting and ending at the index hole. The track format used depends on the density and the disk size. In single-density mode, FM encoding (IBM 3740) is used with 128 data bytes per sector. For double-density, MFM encoding (System 34) is used with 256 data bytes per sector. The data fields are filled with PADB characters. For 5" disks, the number of sectors written is from SECT5 with interleaving from TABLE5 area. For 8" disks, the number of sectors written is from SECTK8 with interleaving from TABLE8. A disk can be formatted by performing WRTRK for every track. Refer to Section 4.4.1 for the default formats.</p> <p>WRTRK does not use the DMA module.</p> <p>On entry, the drive must be active (MOTON/SELECT) with the read/write head positioned on the proper track (SEEK).</p> <p>On exit, any errors (type III) are indicated in the A register or STFLG.</p>	

Table 4-3. Primitive Routine Descriptions (Cont'd)

Routine	Description	Category
WRTSEC		WRITE
	<p>Writes the desired sector of the data to the present track of the selected disk drive. The write buffer (WRBUF) must be filled with the data.</p> <p>WRTSEC will use the DMA module if DMAFLG is enabled. In this case, the Source Bank must correspond to the WRBUF RAM buffer, while the Destination Bank is the FDC module.</p> <p>On entry, the drive must be active (MOTON/SELECT) with the read/write head positioned on the proper cylinder (SEEK). The sector number is in the A register. The data written is pointed to by WRTBUF.</p> <p>On exit, any errors (type II) are indicated in the A register or STFLG.</p>	

## 4.2 CALLING CONSIDERATIONS

The primitives are designed to work with any RM 65 system, with the memory constraints described in Section 3.1. This software can be interfaced to work with any system since it is I/O independent. A necessary vector to be set is the IRQ interrupt. The address of the routine IRQHAN must be put into the IRQ vector.

Due to the physical timing constraints of the floppy disk medium (i.e., the disk rotating at a fixed speed), the FDC routines that read or write data from the disk are time-critical and must not be interrupted (by either IRQ or NMI interrupts) except by the FDC device. If an interruption does occur, the FDC operation will be invalid and the error will be flagged in the status register.

To avoid such errors, calls to a primitive read or write routine should always be preceded by a disabling of all interrupt sources except the FDC module (this cannot be an SEI instruction). Care must be taken with the non-maskable interrupt to ensure proper error handling. After the returning from the primitive routine, the interrupts can then be re-enabled.

There are two data buffers needed for input and output of data to/from the disk. The addresses of these buffers should be put into RDBUF and WRBUF. These buffers should be 128 bytes for single-density and 256 bytes for double-density.

When calling a primitive routine, the action of the CPU depends on the type of the FDC command being performed (every primitive routine typically corresponds to one FDC command). After the FDC device (and DMA controller if applicable) is set up with the appropriate control data, the FDC command is then issued to the FDC device. If a type I command is issued, the CPU will wait until an interrupt is received, indicating command completion. For the type II and III commands - in which data is moved between the disk and memory--the CPU controls the transfer of the data a byte at a time (the FDC device stops the

CPU with the Ready line), with an interrupt indicating completion of the command. When the DMA module is used with type II and III commands, the DMA controls the transfer of data while the CPU will wait for the command completion interrupt.

After receiving an interrupt indicating command completion, the status register is checked to report if the primitive routine was successful. A return is now done to the calling program, which should always examine the status in A or STFLG to ensure that a disk error did not occur before continuing (see Section 4.5).

## 4.3 DISK INITIALIZATION

### 4.3.1 Procedure

The standard initialization procedure for the FDC Module consists of calling INIT. This routine sets up the default variables by initializing the RAM locations used by the operating software and clears the FDC I/O logic. Before initialization, insure that all switches, jumpers, and headers are properly set. If any of the default values must be changed, it must be done after the initialization. Refer to Table 4-1 for the primitive routine addresses.

The typical sequence for initialization (which would be part of the cold RESET procedure) is as follows:

- a) Load the IRQ interrupt request vector for the system interrupt handler with the address of IRQHAN (beginning of the FDC interrupt service routine).
- b) Load the address of the IRQ service routine to follow the FDC interrupt handler into IRQOUT. If there are no additional routines, this IRQOUT can point to an RTI instruction.
- c) Call the INIT subroutine.

- d) Change any default variables to their new value (refer to Sections 4.3.2 and 4.6).
- e) If the DMA module is being used, load the page number (high order byte of the assigned base address) into DMAADR and the source bank, destination bank, and enable information into DMAFLG.

The system is now prepared to operate with any of the primitive routines (refer to Section 4.4).

Whenever a reset occurs while an FDC operation is in progress, it is necessary to force a software interrupt to the FDC device. This is done by the FORCOF routine. Thus, the FORCOF routine will typically be included as part of the warm RESET processing.

#### 4.3.2 Default Conditions

When the FDC module is initialized, the default parameters that are set up will be used for a majority of applications of the module. Any of these default values can be changed to meet the user's requirements. A list of default conditions that might be changed is given in Table 4-4. Refer to Table 4-7 for the variable addresses.

Table 4-4. Primary User-Alterable Variable Default Values

Variable	Default	Description
CYL5	35	Number of tracks on 5" disk
CYL8	77	Number of tracks on 8" disk
DMAADR	0	Pointer to page in which the DMA module resides
DMAFLG	0	DMA module enable and bank selection flags
IRQOUT	None	Pointer to additional interrupt service routines
PADB	\$E5	Pad character used to fill data field
RDBUF	\$1000	Read buffer pointer

Table 4-4. Primary User-Alterable Variable Default Values (Cont'd)

Variable	Default	Description
SECTK5	16	Number of sectors/track on 5" disk
SECTK8	26	Number of sectors/track on 8" disk
TABLE5	Section 4.4.1	Sector interleaving table for 5" disk
TABLE8	Section 4.4.1	Sector interleaving table for 8" disk
TUNDEL	100	Write tunnel delay of 1mS (needed by some drives)
WRTRBUF	\$2000	Write buffer pointer

#### 4.4 DISK PRIMITIVE OPERATION

After the RM 65 FDC Module has been initialized (refer to Section 4.3) the module is prepared for all disk operations. The procedures for some of the more common primitive operations are described in this section.

The primitive routines all fall into four major categories-- Initialize, Drive, Read, and Write. These categories provide a functional grouping of commands. In building application programs with the primitives, these categories should be kept in mind. For example, in a typical disk read access:

```

Set up or recover system - INIT           (Initialize)
Select drive and track   - MOTON and SEEK (Drive)
Read the data            - RDSEC          (Read)
Deselect the drive       - MOTOFF        (Drive)

```

The system or disk must always be set up first (Initialize). After this, the drive selection and head positioning must take place (Drive). The disk access operation can now be performed (Read or Write), followed by drive deselection (Drive). The routines are not fully independent, requiring proper sequencing for meaningful results.

#### 4.4.1 Formatting a Disk

Before any disk can be used for read/write data, the disk must be initialized or formatted. This consists of writing on all tracks of the disk: clock information; track, sector, and side identification (ID Field); data marker and data (Data Field). This disk initialization is compatible with the IBM 3740 format in single-density mode and IBM System 34 format in double-density mode.

In typical IBM compatible disks, sectors will be written sequentially on each track--1, 2, 3, ..., 15, 16 for 5" disks and 1, 2, 3, ..., 25, 26 for 8" disks. When doing sequential disk accesses (which is most of the time), all data movement and calculations preparing for the next sector must be done within the short intersector gap time, otherwise the sector is missed and an entire disk revolution must pass before the sector is available again. This makes the typical access time between sectors very long.

To shorten the typical access time between adjacent sectors, the disks are formatted with sequential sectors not adjacent, i.e., they are interleaved. Sequential sectors are separated by twelve sectors for 5" disks and 20 sectors for 8" disks, which gives at least 12 full sector time periods for data movement and calculations between sectors. This interleaving actually shortens the typical access time between sectors. For 5" disks, the physical sector format recorded on the disk is 1, 6, 11, 16, 5, 10, 15, 4, 9, 14, 3, 8, 13, 2, 7, 12. For 8" disks, the sector order is 1, 6, 11, 16, 21, 26, 5, 10, 15, 20, 25, 4, 9, 14, 19, 24, 3, 8, 13, 18, 23, 2, 7, 12, 17, 22.

The sector interleaving information is kept as a table of data starting at TABLE5 for 5" disks and TABLE8 for 8" disks. Because this is RAM based, the interleaving order can be modified--either to support additional sectors per track or to optimize the inter-sector time period for a specific application.

#### NOTES

1. Disks formatted by the FDC module using the FORMAT routine are IBM compatible in recording technique (FM for single density, MFM for double density) and field structure, therefore they can be read/written on other IBM compatible equipment. This does not imply compatibility of the file structure used or the meaning of the data stored on the disk with other IBM compatible systems.
2. Diskettes created on the Rockwell System 65 cannot be used with the firmware routines because they are not IBM compatible.

There are four disk formats as summarized in Table 4-5.

Table 4-5. Disk Format Parameters

Disk Format	Bytes per Sector	Sectors per Track	Total No. Tracks
Single-density, 5" (FM)	128	16	35
Single-density, 8" (FM)	128	26	77
Double-density, 5" (MFM)	256	16	35
Double-density, 8" (MFM)	256	26	77

To format a disk, the following steps are taken:

- a. Initialize the FDC module (refer to Section 4.3).
- b. Call MOTON (for initial selection), or SELECT (on subsequent selections) to select the drive, side, and density of the disk to be formatted. Check the A register or STFLG and process any errors.

- c. Call FORMAT. The disk in the selected drive will now be formatted for the selected density (refer to Table 4-5). This will take a few seconds. Upon return from FORMAT, check the A register for error conditions (such as disk write protected).
- d. The read/write head is left at the inner-most track. Call SEEK to bring the head back to the desired track (typically, track 0).
- e. Call DESEL or MOTOFF to deselect the drive.

The disk is now prepared for read/write operations. All data fields on the disk are filled with the character in PADB (default is \$E5).

#### 4.4.2 Reading a Sector

To read a sector of data from a formatted disk, the following steps may be followed:

- a. Initialize the FDC module (refer to Section 4.3).
- b. Call MOTON (for initial selection) or SELECT (on subsequent selections) to select the drive, side, and density of the disk being read. Check the A register or STFLG and process any errors.
- c. Call SEEK to move the read/write head to the desired track to be read. The head will remain at this track until another head movement command is performed. Check the A register for errors.
- d. Call RDSEC to read the desired sector of data. The data left is pointed to by RDBUF. Check the A register for errors.
- e. Call DESEL or MOTOFF to deselect the drive.

The read buffer now contains the sector of data read from the disk.

#### 4.4.3 Writing a Sector

To write a sector of data to a formatted disk, the following steps may be followed:

- a. Initialize the FDC module (refer to Section 4.3).
- b. Call MOTON or SELECT to select the drive, side and density of the disk to be written. Check the A register or STFLG and process any errors.
- c. Call SEEK to move the read/write head to the desired track. The head will remain at this track until another head movement command is performed. Check the A register for errors.
- d. Call WRTSEC to write data to the desired sector. The data must be in the buffer pointed to by WRBUF. Check the A register for errors.
- e. Call DESEL or MOTOFF to deselect the drive.

The write buffer data is now written on the disk.

#### 4.5 PRIMITIVE ROUTINES

Whenever a disk operation is performed, error conditions can keep the operation from being completed. Most of the FDC primitive routines return with a status of the operation-- this status reports on incorrect or incomplete operation. After primitive disk functions, the status is left in the A register and STFLG. This status is typically the FDC device status register contents. The meaning of the status bits depends on the type of the routine. Table 4-6 lists the error definitions.

The Type I routines involve the selection of drives or movement of the read/write head. The Type I routines are: FORCOF and SEEK.

The Type II routines control reading and writing sectors of data to and from the disk. The Type II routines are: RDMSC, RDSEC, WRTMSC, and WRTSEC.

The Type III routines are for reading and writing of data and clock information, a track at a time, as well as reading the ID Field. The Type III routines are: FORMAT, RDTRK, and WRTRK.

All the above routines (types I, II, and III) can also return a motors not on (the SELECT routine does also) or no drive selected error (\$80).

The remaining primitive routines do not return any error information: DESEL, INIT, MOTOFF, MOTON.

Dependent on the routine called, certain error checks should always be performed on return. Some of the error conditions (such as write protection) are checked within the routines, while others rely entirely on the application software for correction.

#### 4.6 PRIMITIVE ROUTINES VARIABLES

The variables used by the primitive routines are located and identified in Table 4-7.

Table 4-6. Primitive Routine Error Definitions

Type	STFLG(A)	Description
I	Bit 1	Drive not ready (8" Drive only).
	Bit 6	Disk is write protected
	Bit 5	- - -
	Bit 4	Seek error
	Bit 3	CRC error in ID Field
	Bit 2	Track 00
	Bit 1	- - -
	Bit 0	FDC device is busy
II and III	Bit 7	Drive not ready (8" Drive only).
	Bit 6	Disk is write protected
	Bit 5	Read Record Type error (1 = Deleted Data Mark) Write error
	Bit 4	Record not found
	Bit 3	CRC error in ID Field
	Bit 2	Lost Data error
	Bit 1	- - -
	Bit 0	FDC device busy.
NOTE		
An \$80 indicates a Motors Not On or No Drives Selected error for Type I, II, and III routines.		

Table 4-7. Primitive Routine Variables

Addr (Hex)	Label	No. Bytes	Init Value (Hex)	Definition
0DB	PTR	2	-	Source/Dest. Buffer Pointer
0DD	DMAPTR	2	-	Pointer to DMA Module Addr.
4A0	IRQOUT	2	-	IRQ Exit Address*
4A2	FLAG	1	00	Last Command Performed
4A3	CNTR	1	00	Counter
4A4	TEMP	2	00 00	Temporary Storage
4A6	TEMP2	1	00	Temporary Storage
4A7	MSC	1	00	DMA Most Significant Count
4A8	LSC	1	00	DMA Least Significant Count
4A9	NCYL	1	00	No. of Tracks
4AA	NSEC	1	00	No. of Sectors
4AB	CURSEC	1	00	Current Sector No.
4AC	LSECR	1	00	Current Sector To Read Or Write
4AD	STFLG	1	00	Status Flag
4AE	CURCMD	1	00	Current Command Being Executed
4AF	WRNFLG	1	00	Write Command Flag
4B0	SELFLG	1	00	Select Flag
4B1	MOFLG	1	00	Motor Flag
4B2	USIDE	1	00	Side No.
4B3	UCYL	1	00	Track No.
4B4	UDRV	1	00	Drive No.
4B5	CURCYL	4	00 -> 00	Current Track For Each Drive
4B9	DMAADR	1	00	DMA MSB I/O Address*
4BA	DMAFLG	1	00	DMA Flag*
4BB	FORFLG	1	00	Write Track Flag
4BE	DBLCNT	1	FF	Double-Density Counter
4BF	SINGLCT	1	7F	Single-Density Counter

NOTES

\*User-alterable.

Table 4-7. Primitive Routine Variables (Cont'd)

Addr (Hex)	Label	No. Bytes	Init Value (Hex)	Definition
4C0	NBYTE	1	80	Bytes Per Sector
4C1	NHEAD	1	01	No. of Heads*
4C2	CYL5	1	23	Tracks Per 5" Disk*
4C3	CYL8	1	4D	Tracks Per 8" Disk*
4C4	SECTK5	1	10	Sectors Per 5" Disk*
4C5	SECTK8	1	1A	Sectors Per 8" Disk*
4C6	HDEL	1	1E	Delay for Head Load*
4C7	SFLAG	1	06	Side CMP & Data Mark (C&E)
4C8	TUNDEL	1	64	Delay for Non-Shugart Tunnel
4C9	RDBUF	2	00 3E	FDC Source Buffer Vector*
4CB	WRBUF	2	00 3F	FDC Destination Buffer Vector*
4CD	RATE	1	07	Stepping Rate
4CF	RETRY	1	05	Number Of Retries
4CF	PADB	1	E5	Format Byte Pattern*
4D0	TABLE5	16	(Note 1)	5" Sector Interleaving Table*
4E0	TABLE8	26	(Note 2)	8" Sector Interleaving Table*

NOTES

\*User-alterable.

- 6,7,8,9,10,11,12,13,14,15,16,0,2,3,4,5 (decimal).
- 6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,0,2,3,4,5 (decimal).

## SECTION 5

### USING AIM 65 DOS VERSION 1.0

The AIM 65 Disk Operating System (DOS) Version 1.0 (A65-090) integrates the RM 65 FDC primitive subroutines with fundamental file management functions for use on the AIM 65 Microcomputer. These ROM-based functions, contained in a 4K-byte R2332 ROM, are accessible from the operator through the Debug Monitor/Text Editor ROMs as well as language ROMs and from an application program. Being ROM-based, DOS operation may proceed immediately upon power turn-on without waiting for separate loading of the DOS into RAM.

Text and program source code may be written to, and read from, floppy disk with the Editor List and Read commands, respectively. Similarly, binary data and program object code may be written to, and loaded from, disk using the Monitor Dump and Load commands, respectively. Files containing source and object code for application programs written in AIM 65 Assembler, BASIC, FORTH, PL/65 and Pascal languages are, therefore, supported.

DOS primary commands are selected by the operator at the Monitor command level. These commands invoke the following functions:

- Format a Disk
- List the Directory
- Backup a Disk
- List a File
- Delete a File
- Recover a File

Files are created automatically upon writing a file to disk. A file name and the disk drive number are operator entered in response to system prompts.

Disk read or write errors, both at the DOS and FDC device level, are reported upon detection. User-alterable variables allow changing of default values to application unique values.

## 5.1 INITIALIZATION

After installing the AIM 65 DOS ROM on the RM 65 FDC module, the DOS is ready for use. The DOS must first be initialized, however, by executing the startup routine, e.g.:

```
<*>=8000
<G>/.
```

The FDC module primitive subroutine variables are initialized, the DOS variables are set to default values (see Section 5.7), the IRQ vectors are initialized, the F1 and F2 user function keys are linked to DOS, the user-defined I/O vectors are loaded for floppy disk I/O, and other temporary variables are initialized. Control then returns to the I/O ROM and subsequently to the Monitor command level. DOS commands may be entered when the Monitor prompt is displayed. (The DOS may not be operated without the Monitor/Editor ROMs installed.)

AIM 65 DOS 1.0 is designed primarily to support AIM 65 Monitor/Editor and language functions in response to commands entered from the keyboard with response directed to the display/printer. When used in this manner, the Monitor/Editor ROMs must be installed. The DOS may also be used to write and read a data file under program control (see Section 5.5). In this case, the Monitor/Editor ROMs are not required unless a Monitor subroutine is used.

### NOTE

The DOS uses two contiguous 256-byte buffers, the output buffer and the input buffer, to transfer data between RAM and a floppy disk (as well as for other intermediate data storage). The high byte of the output (or source) buffer is specified by the variable BUFFER (see Section 5.7). The input (or destination) buffer is located immediately above the output buffer. BUFFER is initialized by the startup routine (\$8000) to \$06 which locates the buffers in low RAM (i.e., \$600-7FF).

### NOTE (Cont'd)

Application programs must be located above the FDC module primitive subroutine variables (\$040A-\$4FF-- see Section 4.6), DOS 1.0 variables (\$0500-\$0564-- see Section 5.7) and above the DOS buffers (if left at their default locations, i.e., \$600-\$7FF).

## 5.2 DISK FORMAT

A floppy disk must be formatted in the AIM 65 DOS 1.0 format before it can be used (see Section 5.3.3). The DOS disk format (illustrated in Figure 5-1) reserves track 0 for future use, uses track 1 and one-half of track 2 for the directory, and the rest of the tracks for sector header and data storage. The sectors are formatted sequentially on the disk.

Trk No.	Sect No.	Function	Single-Density		Double-Density			
			Byte No.	File No.		Byte No.	File No.	
				5"	8"		5"	8"
0	All	Reserved	1-128	-	-	1-256	-	-
1	1	File Entry	1- 16	1	1	1- 16	1	1
		File Status	1			1		
		00=Directory End						
		01=Active File						
		02=Deleted File						
		File Name	2- 11			2- 11		
		Start Track	12			12		
		Start Sector	13			13		
		Length-Low Byte	14			14		
		Length-High Byte	15			15		
		Null (00)	16			16		
		File Entries	17-128	2- 8	2- 8	17-256	2- 16	2- 16
2		File Entries	1-128	9-16	9- 16	1-256	17- 32	17- 32
3- 8		File Entries	1-128	17-64	17- 64	1-256	33-128	33-128
9-13		File Entries	1-128	-	65-104	1-256	-	129-208
14-16		Free	1-128	-	-	1-256	-	-
2	All	Free	1-128	-	-	1-256	-	-

Figure 5-1. AIM 65 DOS 1.0 Disk Format

### 5.3 PRIMARY OPERATOR COMMANDS

The primary operator commands are selected by using one of two keys: F1 and F2. The F1 key directly commands the List Directory Function, (see Section 5.3.1) while the F2 key displays a UTILITY = prompt. One of five utility functions may then be commanded (see Sections 5.3.2 through 5.3.3). If an invalid key is pressed for a utility function, the utility menu is displayed, e.g.:

```
<]> UTILITY=<RETURN>
(F)ORMAT, (B)ACKUP, (L)IST, (D)EL, (R)EC
```

The <ESC> key is vectored through variable FESCIV (see Section 5.7) to jump to the Monitor command level. The <ESC> key vector is initialized upon cold reset and may be user-altered. Note that this vector is also used to return control to the Monitor after reporting an error condition (see Section 5.6).

#### NOTE

Upon cold reset, the density selection corresponding to each disk drive is initially set to double. If a disk cannot be read at the selected density, DOS automatically switches to the other density, and attempts to read the disk again. If the read is successful, the selected density is saved for future reading from that disk drive (until a cold reset is performed). This process is essentially transparent to the operator except for a slight delay.

If DOS cannot read the disk at either density, the operator will be prompted to enter the disk density (e.g., DENSITY= ) upon the next disk command. The DOS will then attempt to read the disk again at the commanded density (and, if necessary, the other density).

### 5.3.1 List the Directory

The List Directory function displays the file name of all files recorded on the disk, the sector length of the files and the active/deleted status of each file. The number of free sectors, i.e., the amount of space left on the disk, is also reported.

To list the directory, press the F1 key, then enter the disk drive number in response to the prompt.

Example:

To list the contents of the directory on disk drive No. 1:

```
<[>
DISK=1
FILE NAME   LEN S
LOGGER1.OBJ 269 A
TEXT1       193 A
TEXT2        26 A
SEC LEFT= 178
```

### 5.3.2 Backup a Disk

The Backup Disk function copies all the active files from one disk to another disk. Since deleted files on the input disk are not copied to the output disk, the newly created disk is essentially compressed (i.e., deleted files removed) to provide additional free space.

To backup a disk, press the F2 key to request the utilities prompt, then press the B key. Enter the disk drive number of the disk to copy from, then the disk drive number of the disk to copy to, in response to prompts.

Example:

To copy a disk on disk drive No. 1 to a disk on disk drive No. 2:

```
<|>
UTILITY=B DISK=1
DISK=2
```

### 5.3.3 Format a Disk

The Format Disk function initializes a disk to prepare it for subsequent use (see Section 5.2). A new disk, i.e., an unformatted disk, must be formatted before a file can be written upon it. It may also be desired to initialize a previously formatted disk to remove all existing files before re-use.

The Format Disk function also verifies proper operation of the disk media before file storage use. The disk is first initialized by writing sector headers, gaps and CRC data, then a \$E5 byte pattern to all sectors. The directory is next initialized to all \$00 bytes. If the NOVRFY flag (see Section 5.7) is non-zero (default value), the sector headers are verified, otherwise control returns to the Monitor command level. If the NOVRFY flag is zero, the sector headers are not checked, allowing a faster formatting of the disk.

To format a disk, press the F2 key to request the utilities prompt, then press the F key. Enter the desired disk number and density character when requested. The disk number may vary from 1 to 4 for single-sided disks and from 1 to 8 for double-sided disks (i.e., each side is treated as a separate drive number with odd numbers being the front side and even numbers being the back side). The density character may be either S (for single-density) or D (for double-density).

Since the initialization completely overwrites the disk, a command verification prompt, "ARE YOU SURE?", is displayed before continuing, to prevent inadvertent reformatting. Press Y to continue, or any other key to terminate the command.

The WAIT message is displayed during disk initialization, followed by the Monitor command prompt upon completion. The number of free sectors and bytes is dependent upon disk size and density while the initialization time is dependent upon the state of the NOVRFY flag:

Disk Size	Density Code	Approx. Format Time (Min:Sec)		Free Sectors	Free Bytes
		Verify NOVRFY≠0	No Verify NOVRFY=0		
5"	S	1:45	0:20	535	68,480
5"	D	1:45	0:20	535	136,960
8"	S	5:00	0:30	1962	251,136
8"	D	5:00	0:30	1962	502,272

Example:

Initialize a 5" disk in disk drive No. 1 to single-density then list the directory to check it.

```
<|> UTILITY=F
DISK=1 DENSITY=S
ARE YOU SURE?Y
WAIT
```

```
<[>
DISK=1
FILE NAME LEN S
SEC LEFT= 535
```

#### NOTES

1. An RM 65 DMA module (RM65-5104) must be used to write and read double-density on 8" disks.
2. If an RM 65 DMA module is used (with either 5" or 8" disk drives), refer to Section A.3 for operating instructions.

#### 5.3.4 List a File

The List File function lists the contents of a file from a disk to another peripheral device. If the peripheral is another disk, a file may be copied from one disk to another.

To list a file, press the F2 key to request the utilities prompt, then press the L key. Enter the file name, disk drive number, and output device code as requested. Respond to output device code subprompts.

Example 1:

To list a text file to the display/printer:

```
<]> UTILITY=L F=TEXT3
DISK=1
OUT=<RETURN>
```

Example 2:

To list an object code file from disk drive No. 1 to disk drive No. 2:

```
<]>
UTILITY=L F=PROG1.OBJ
DISK=1
OUT=U F=PROG1.OBJ
DISK=2
```

The listing may be suspended by pressing the <SPACE> bar, then resumed by pressing the <SPACE> bar again (or one of the other keys).

#### 5.3.5 Delete a File

The Delete File Function changes a file from the active (A) state to the deleted (D) state. In the deleted state, the file cannot be accessed by file name by any function except Recover File (see Section 5.3.5).

To delete a file, press the F2 key to request the utilities prompt, then press the D key. Enter the file name and disk drive number in response to prompts. The Monitor prompt will be displayed upon completion. List the directory to verify the file deletion.

Example:

To delete file TEXT3 (currently active) from disk drive No. 1:

```
<]>
UTILITY=D F=TEXT3
DISK=1
{[]
DISK=1
FILE NAME  LEN S
TEXT1      193 A
TEXT2       94 A
TEXT3       63 D
SEC LEFT= 238
```

A deleted file may be returned to the active state by using the Recover File function.

#### 5.3.6 Recover a File

The Recover File function changes a file from the deleted (D) state to the active (A) state. In the active state, the file can be accessed by file name by other DOS functions.

To recover a file, press F2 key to request the utilities prompt, then press the R key. Enter the file name and disk drive number in response to prompts. The Monitor prompt will be displayed upon completion. List the directory to verify file recovery.

Example:

To recover file TEXT3 (currently deleted) from a disk on disk drive No. 1:

```

<]>
UTILITY=R F=TEXT3
DISK=1
<[>
DISK=1
FILE NAME  LEN S
TEXT1      193 A
TEXT2       44 A
TEXT3       63 A
SEC LEFT= 238

```

### 5.3.7 Extension of UTILITY Functions

The DOS links to the UTILITY functions through two indirect jump vectors. These two vectors, MENUVC and MENULS, are user-alterable and may be changed to provide alternative linkage and functions. Upon depression of the ] key, the DOS jumps through the before menu vector, MENUVC, to check for the five default command characters. If the command character is an F, B, L, D or R, the appropriate processing is performed (see Section 5.3.2 through 5.3.6). If the command character is not one of these letters, the DOS jumps through the after menu vector, MENULS, to display the command menu then redisplay the "UTILITY=" prompt.

Either one or both of these vectors may be altered to meet application dependent requirements.

## 5.4 FILE HANDLING UNDER OPERATOR CONTROL

A file may be written to, or read from, a disk under operator control by pressing U in response to the OUT=, or IN=, prompt displayed by other major functions, e.g., Monitor, Editor or language. Enter the file name and disk number in response to subprompts.

### 5.4.1 Using the AIM 65 Monitor

The Monitor Dump (D) function can be used to output machine code from memory to a disk file in ASCII format, while the Monitor Load (L) function can be used to read the file from a disk into RAM (see Sections 4.8.2 and 4.8.1, respectively, in the AIM 65 User's Guide).

### a. Writing a File

Example:

Dump machine code, in ASCII, from addresses 0800 through 08FF to file PGML\*A on disk drive No. 1:

```

<D>
FROM=0800 TO=08FF
OUT=U F=PGML*A
DISK=1
MORE?N

```

### b. Reading a File

Example:

Load file PGML\*A from disk drive No. 1 into RAM:

```

<L>IN=U F=PGML*A
DISK=1

```

### 5.4.2 Using the AIM 65 Editor

The Editor List (L) function can be used to output ASCII text from the Editor Text Buffer in RAM to a disk file while the Read (R) function can be used to read the file into the Text Buffer (see Sections 5.4.9 and 5.4.1, respectively, in the AIM 65 User's Guide).

Be sure that the Text Buffer is located above the DOS variables and input/output buffers, e.g., \$800. If the buffers are relocated, the Text Buffer may start at \$600.

### a. Writing a File

Example:

Write all the text in the Text Buffer to file PGML on disk drive No. 1:

```
=<T>
=<L>
/.
OUT=U F=PGM1
DISK=1
```

#### b. Reading a File

##### Example:

Read file PGM1 from disk drive no. 1 into the Text Buffer:

```
=<R>
IN=U F=PGM1
DISK=1
```

END

#### 5.4.3 Using the AIM 65 Assembler

Source code may be read from a disk file into the AIM 65 assembler and/or object code may be written to a disk file from the assembler during the assembly process.

Be sure that the symbol table and, if located in RAM, the source and/or object code are located above the DOS variables and buffers, e.g., \$800. If the DOS buffers are relocated, RAM can be used starting at \$600.

To assemble a program from source code on the disk, the program shown in Figure 5-2 must first be loaded. Entry to the assembler is now made using the <F3> rather than the usual <N> command. Only use this <F3> entry when assembly is from source code on disk. When using this program, the motors must be turned off after the assembly is complete by depressing the RESET switch. Use of this program is demonstrated in Examples 2 and 3.

##### NOTE

When assembling from disk using the driver in Figure 5-2, the first character of the source file is ignored. Begin the first line with a space to avoid an error from occurring.

```
ADDR OBJECT SOURCE
                                FOPIN=$8110
                                NOPENA=$8155
                                ASMVEC=$53F
                                ASMFLG=$536
                                *=$112 ;SET UP F3 KEY FOR ENTRY
0012 4C 41 05 JMP START
                                *=$ASMVEC+2 ;FREE MEMORY
0041 A9 01 START LDA #01 ;FLAG FOR SOURCE ON DISK
0043 8D 36 05 STA ASMFLG
0046 A9 53 LDA #<PASSCK ;CHECK ON EVERY BYTE OF SOURCE
0048 A0 05 LDY #>PASSCK ;THAT IT IS STILL PASS 1
004A 8D 3F 05 STA ASMVEC
004D 8C 40 05 STY ASMVEC+1
0050 4C 00 D0 JMP $D000 ;ENTER THE ASSEMBLER

0053 A5 23 PASSCK LDA #23 ;CHECK FOR PASS 2
0055 2D 36 05 AND ASMFLG
0058 D0 03 BNE PASS2 ;STILL PASS 1 -- CONTINUE
005A 4C 55 81 JMP NOPENA

005D A9 00 PASS2 LDA #00 ;ZERO THE ASSEMBLER FLAG
005F 8D 36 05 STA ASMFLG ;TO SKIP PASSCK
0062 20 10 81 JSR FOPIN ;REOPEN FILE
0065 4C 55 81 JMP NOPENA ;GET THE CHAR
                                .END
```

Figure 5-2. Driver to Assemble from Source Code on Disk

##### Example 1:

Assemble source code from memory (i.e., the Text Buffer) and write the generated object code in ASCII to file PGM1\*A on disk drive No. 2:

```
<N>
ASSEMBLER
FROM=1800 TO=1FFF
IN=M
LIST?N
LIST-OUT=<RETURN>

OBJ?Y
OBJ-OUT=U F=PGM1*A
DISK1

PASS 1

PASS 2

ERRORS= 0000
```

Example 2:

Assemble source code from file PGMI on disk drive No. 1 and do not generate object code.

```
<F3>
ASSEMBLER
FROM=1800 TO=1FFF
IN=U F=PGMI
DISK=1
LIST ?N
LIST-OUT=<RETURN>

OBJ?Y
OBJ-OUT=X

PASS 1

PASS 2

ERRORS= 0000
```

Example 3:

Assemble source code from file PGMI on disk drive No. 1 and write the generated object code to file PGMI\* on disk drive No. 1:

```
<F3>
ASSEMBLER
FROM=1800 TO=1FFF
IN=U F=PGMI
DISK=1

LIST?N
LIST-OUT=<RETURN>

OBJ?Y
OBJ-OUT=U F=PGMI*
DISK=1

PASS 1

PASS 2

ERRORS= 0000
```

#### 5.4.4 Using AIM 65 BASIC

The starting address of RAM used by BASIC to store the application program and variables must be changed from its default value of \$211 to a value above the DOS 1.0 variables and input/output buffers, e.g., \$0800.

##### a. BASIC Program Relocation

The following can be used to change the start of the BASIC program:

1. Enter BASIC from the Monitor and enter the memory size and terminal width in response to prompts.

```
<5>
MEMORY SIZE?
WIDTH?
11758 BYTES FREE
AIM 65 BASIC V1.1
```

2. Press <ESC> to return to the Monitor.
3. Move the BASIC program and variables start address from \$211 to greater than \$7FF, e.g., \$800. Also, load zeros into three bytes at the start of the BASIC program.

```
<M>=0073 12 02 14 02
</> 0073 01 08 03 08
<M>=0800 AA AA AA AA
</> 0800 00 00 00 <RETURN>
```

4. Initialize DOS 1.0:

```
<*>=8000
<G>/.
```

5. Re-enter BASIC.

```
<6>
```

6. Enter BASIC commands.

The AIM 65 BASIC SAVE and LOAD functions may be used to write and read a BASIC application program, to and from a disk file, respectively (see Section 303 in the AIM 65 BASIC Language Reference Manual).

b. Saving a File

Example:

Save the BASIC program in RAM to file PGM3 on disk drive  
No. 1:

```
SAVE
OUT=U F=PGM3
DISK=1
```

c. Loading a File

Example:

Load a BASIC program into RAM file PGM3 on disk drive  
No. 1:

```
LOAD
IN=U F=PGM3
DISK=1
```

5.4.5 Using AIM 65 FORTH

The source code for an application program written in AIM 65 FORTH may be formatted to edit in the AM 65 Text Editor or to input in the FORTH screen format. Refer to Sections 5.4.9 and 5.4.1 in the AIM 65 User's Guide to write source code from the Text Editor to disk or to read source code from disk into the Text Editor. Refer to Section 12 in the AIM 65 FORTH User's Manual for FORTH screen format usage.

The start-up address of RAM used by AIM 65 FORTH to store the application words must be changed from its default value of \$300 to a value above the DOS variables and input/output buffers, e.g., \$0800.

a. FORTH Application Dictionary Relocation

The following procedure can be used to change the start of the application words:

1. Enter FORTH from the Monitor with the 5 key.

```
<5>
AIM 65 FORTH V1.2
```

2. Move the FORTH starting address from \$300 to greater than \$7FF, e.g., \$800.

```
FORGET TASK
1280 ALLOT
:TASK ;
```

NOTE

Any time FORTH is entered with the 5 key, the above steps must be repeated.

3. Press <ESC> to return to the Monitor.

4. Initialize DOS:

```
<*>=8000
<G>/.
```

5. Re-enter FORTH with the 6 key.

```
<6>
```

6. Enter FORTH commands.

#### 5.4.6 Using AIM 65 PL/65

The source code for an application program written in AIM 65 PL/65 may be read from disk into the PL/65 compiler and/or the generated object code (i.e., in 6502 assembly language) may be written to disk (see Section 2.6 in the AIM 65 PL/65 User's Manual).

##### a. Writing a File

Compile source code from memory (i.e., the Text Buffer) and write the output assembler code to file PGM2 or disk drive No. 2:

```
<5>
AIM 65 PL/65 V1.0
IN=M OUT=U F=PGM2
DISK=1
PASS(1 OR 2)?2

ERRORS=00
```

##### b. Reading a File

Compile source code from file PGM2L on disk drive No. 1 and output the generated assembly language to memory:

```
<5>
AIM 65 PL/65 V1.0
IN=U F=PGM2L
OUT=M
PASS(1 OR 2)?2

ERRORS=00
```

#### 5.4.7 Using AIM 65 Instant Pascal

An application program written in Pascal can be compiled (i.e., to an internal language code for interpretation) into RAM, from a disk file, by AIM 65 Instant Pascal. The source code for an application program in RAM may also be translated from internal format to structured Pascal statements and written to a disk file. Refer to Sections 3.1 and 3.2.3, respectively, in the AIM 65 Instant Pascal User's Manual for user instructions.

The start-up address of RAM used by AIM 65 Instant Pascal to start the application program internal code must be changed from its default value of \$02FC to a value above the DOS 1.0 variables and input/output buffers, e.g., \$0800.

##### a. Instant Pascal Program Relocation

The following procedure can be used to move the start-up address of the internal code:

1. Enter Instant Pascal from the Monitor and enter the memory size and terminal width in response to prompts.

```
<5>MEMORY WIDTH?
WIDTH?
```

2. Press <ESC> to return to the Monitor.

3. Change the program start-up address to a value greater than \$7FF, e.g., \$800:

```
<M>=001A FC 02 XX XX
</> 001A FC 07 XX XX
```

4. Initialize AIM 65 DOS 1.0:

```
<*>=8000
<G>/.
```

5. Re-enter Instant Pascal:

```
<5>
```

6. Enter Pascal statements.

#### NOTE

Any time a Cold Reset (Pascal Z command) is performed, location \$1A-\$1B will be reset to \$02FC.

b. Writing a File

Example:

Write the Pascal statements for a program from AIM 65 Instant Pascal to file PGM4P on disk drive No. 1:

```
+<L>/.  
OUT=U F=PGM4P  
DISK=1
```

c. Reading a File

Example:

Read the Pascal statements into AIM 65 Instant Pascal from file PGM4P on disk drive No. 1:

```
+<R>IN=U F=PGM4P  
DISK=1
```

c. Read a File

Compile source code from file PGM2L on disk drive No. 1 and output the generated assembly language to memory:

```
<5>AIM 65 PL/65 V1.0  
IN=F F=PGM2L  
DISK=1  
OUT=M  
PASS(1 OR 2)?2  
ERRORS=00
```

5.5 FILE HANDLING UNDER PROGRAM CONTROL

Often it is desirable to store and/or retrieve data from a floppy disk entirely under program control, e.g., when collecting data in an unattended installation. While this can be easily done by writing a program which calls the primitive routines (see Section 4), use of the primitive routines alone do not support the directory format used by AIM 65 DOS 1.0. DOS compatible data files written under program control allow easy examination and handling under operator control using DOS

utility functions. This section describes a method to write data to and read data from a floppy disk in a format that is compatible with the AIM 65 DOS 1.0 file structure.

5.5.1 Writing and Reading Data Files

The AIM 65 DOS 1.0 file structure allows the writing and reading of data to and from disk independent of the data meaning (e.g., program source code, program object code, or symbol table) or data format (e.g., ASCII coding or machine code). The assembly listing in Figure 5-3 details a DOS compatible file handler. This handler contains both an output driver (to write a file to disk) and an input driver (to read a file from disk). The DOS subroutines and variables used by these drivers are listed in Tables 5-1 and 5-2, respectively.

To write a file, first set up the file name, disk number and side number, then call the write open subroutine (OPENWR). Next, pass data a byte at a time to the disk file by loading each byte into the Accumulator and calling the Monitor OUTALL subroutine. When all data bytes have been output, output an End of File (EOF) character (\$1A) and call the write close subroutine (CLOSEW) to close the file. The CLOSEW subroutine automatically outputs the additional fill bytes (value=\$00) to complete the last sector.

To read a file, first set up the file name, disk number and side number, then call the read open subroutine (OPENRD). Next, read data from the disk file, a byte at a time, by calling the Monitor INALL subroutine and storing the read byte in the Accumulator into memory. When an EOF is received, turn off the disk drives, call the read close file (CLOSER) to close the file.

NOTE

This input driver uses \$1A as the EOF indicator. When \$1A is detected by the INALL subroutine, the disk drive motors are turned off. The user program must close the file however.

PAGE 0001 ROUTINES TO READ AND WRITE FILES FOR AIM 65

```

ADDR OBJECT SOURCE
;
; FDC ADDRESSES ( 5-26-82 )
DISK2=#885B
DISK1=#884A
TRKSK=#84E6
CLOPEN=#81A7
FOPIN=#8110
CLSOUT=#82A0
CLSIN=#8195
LISTRT=#8353
DOSSET=#8006
WRITIT=#8248
; FDC VARIABLES
FINOUT=#522
DSTBUF=#D9
SRCBUF=#D7
BUFFER=#531
CLOUTV=#52E
EOF=#537
;
; AIM 65 VARIABLES
OUTFLG=#A413
INFLG=#A412
;
; **$F00
;
; AFTER TURNING ON THE SYSTEM , THE FDC FIRMWARE MUST BE
; INITIALIZED ( TYPICALLY AS A RESET PROCEDURE ).
; THIS CAN BE UNDER OPERATOR CONTROL BY EXECUTING #8000
; OR UNDER PROGRAM CONTROL BY :
; INIT JSR DOSSET ; INITIALIZE FDC DOS
;
; TO OPEN A WRITE FILE , SET UP DISK, SIDE AND
; NAME BEFORE CALLING THIS OPEN ROUTINE:
; DISK = #501 - NUMBER 0-7
; SIDE = #503 - 0 FOR FIRST SIDE
; FILENAME = #518 - 10 CHARACTERS PADDED WITH #20
; DENSITY = #505 - 0 FOR DOUBLE (OPTIONAL)
;
0F00 A2 01 OPENW LDX #1 ; USE WRITE BUFFER
0F02 8E 22 05 STX FINOUT
0F05 A9 53 LDA #CLISTR ; DISABLE CHARACTER CHECKS
0F07 A0 83 LDY #CLISTR
0F09 8D 2E 05 STA CLOUTV
0F0C 8C 2F 05 STY CLOUTV+1
0F0F 20 5B 88 JSR DISK2 ; TURN ON WRITE DRIVE
0F12 A9 00 LDA #0 ; SET UP DESTINATION BUFFER
0F14 85 D9 STA DSTBUF
0F16 AC 31 05 LDY BUFFER
0F19 C8 INY ; WRITES USE BUFFER+1
0F1A 84 DA STY DSTBUF+1
0F1C 20 E6 84 JSR TRKSK ; RESTORE WITH DENSITY CHECK
0F1F 20 A7 81 JSR CLOPEN ; OPEN THE FILE
0F22 A9 55 LDA #10 ; POINT OUTALL TO FLOPPY
0F24 8D 13 A4 STA OUTFLG
0F27 60 RTS

```

Figure 5-3. Program Control File Handler

PAGE 0002 ROUTINES TO READ AND WRITE FILES FOR AIM 65

```

ADDR OBJECT SOURCE
;
; TO WRITE DATA , CALL OUTALL WITH BYTE THE IN A .
; THE USER DRIVER WILL WRITE 128/256 BYTES AT A TIME .
; PASS AN EOF AS THE LAST CHARACTER .
;
; TO CLOSE THE WRITE FILE
0F28 AD 37 05 CLOSEW LDA EOF ; CLOSE OUT LAST SECTOR
0F2B 20 48 82 JSR WRITIT
0F2E A9 20 LDA #20 ; RESTORE AOD TO AIM 65
0F30 8D 13 A4 STA OUTFLG
0F33 4C AD 82 JMP CLSOUT ; CLOSE DISK FILE
;
; TO OPEN A READ FILE , SET UP DISK, SIDE AND
; NAME BEFORE CALLING THIS OPEN ROUTINE:
; DISK = #500 - DRIVE NO. 0-7
; SIDE = #502 - 0 FOR FIRST SIDE
; FILENAME = #50E - 10 CHARACTERS PADDED WITH #20
; DENSITY = #504 - 0 FOR DOUBLE (OPTIONAL)
;
0F36 A2 00 OPENR LDX #0 ; USE READ BUFFER
0F38 8E 22 05 STX FINOUT
0F3B 20 4A 88 JSR DISK1 ; TURN ON READ DRIVE
0F3E A9 00 LDA #0 ; SET UP SOURCE BUFFER
0F40 85 D7 STA SRCBUF
0F42 AC 31 05 LDY BUFFER
0F45 84 D8 STY SRCBUF+1
0F47 20 E6 84 JSR TRKSK ; RESTORE WITH DENSITY CHECK
0F4A 20 10 81 JSR FOPIN ; OPEN THE READ FILE
0F4D 60 RTS
;
; TO READ DATA , CALL INALL - BYTE RETURNED IN A .
; CHECK FOR AN EOF CHARACTER IF NEEDED .
;
; TO CLOSE THE READ FILE
0F4E 4C 95 81 CLOSER JMP CLSIN ; CLOSE DISK READ FILE
END
ERRORS=0000

```

Figure 5-3. Program Control File Handler (Cont'd)

Table 5-1. Commonly Used AIM 65 DOS 1.0 Subroutines

Label	Address (Hex)	Function
CLOPEN	81A7	Open the Write Disk File.
CLSIN	8195	Close the Read Disk File and Turn Off the Drives.
CLSOUT	82AD	Close the Write Disk File and Turn Off the Drives.
DISK1	884A	Set Up and Turn On the Read Drive.
DISK2	885B	Set Up and Turn On the Write Drive.
DOSSET	8006	Initialize FDC Primitive and DOS Variables.
ERRLCL	84D6	Error Handler for the AIM 65 DOS (includes FESC).
FESC	84DA	Turn Off Motors and Jump Through FESCIV.
FOPIN	8110	Open the Read Disk File.
LISTRT	8353	Return Instructions.
TRKSK	84E6	Seek to the Track in UCYL with Verify and Density Check.
WRTIT	8248	Write out one Byte from A (called through OUTALL).

Table 5-2. Commonly Used AIM 65 DOS 1.0 Variables

Label	Address (Hex)	Function
BUFFER	531	MSB of Pointer to DOS Buffer.
CLOUTV	52E	Pointer to the close check for write files.
CURCMD	4AE	Last Command Executed by FDC device.
EOF	537	Character used as an End-of-File marker.
DSTBUF	0D9	Pointer to DOS write buffer.
ERSTAT	524	DOS error status.
ERRVEC	532	Pointer to the DOS error processing routine
FBYTE	50A	Buffer 1 current byte (Read).
	50B	Buffer 2 current byte (Write).
FESCIV	534	Pointer to the DOS escape processing routine
FDISK	500	Buffer 1 disk number (Read).
	501	Buffer 2 disk number (Write).
FSIDE	502	Buffer 1 side number (Read).
	503	Buffer 2 side number (Write).
FDEN	504	Buffer 1 density (Read).
	505	Buffer 2 density (Write).
FNAMES	50E-517	Buffer 1 file name (Read - 10 characters).
	518-521	Buffer 2 file name (Write) - 10 characters.
FINOUT	522	Read/Write indicator (0=Read).
LEN	53D	Number of remaining sectors (Read).
STFLG	4AD	Image of FDC device status register.
SRCBUF	0D7	Pointer to DOS Read buffer.

If an error occurs during writing or reading using this handler, the IRQ interrupts are reenabled, the disk drive motors are turned off and control is returned to the AIM 65 Monitor command level. The source of the error can be determined by examining the command type (WRCMD), the FDC Status Register (STFLG), and the DOS error status (ERSTAT).

Note that the address of the error handler is placed in ERRVEC and the address to jump to after the error has been serviced is placed in FESCIV. If errors are to be handled under program control, these vectors must be loaded appropriately. The error handler must isolate and service the error as required. If the disk operation is to be terminated, the IRQ interrupts must be reenabled (JSR ION) the disk drive motors turned off (JMP FESC) and I/O restored to the system terminal.

#### 5.5.2 DOS Compatibility Considerations

The user is responsible for the meaning and coding of all data in files created under program control, including the definition and inclusion of control bytes. Sometimes the files are handled by knowing exactly how many bytes are written to a file, and therefore how many to expect upon reading it (which can be detected by a logical zero result in LEN, LEN+1 and FBYTE). Any \$00 pad bytes used to fill the last sector can be ignored.

Another approach is to include one or more special characters to indicate the end of a file. The end of file character(s) must not conflict with valid data values using their approach, however.

In order for a data file to be listed using the DOS list utility function (UTILITY=L, see Section 5.3.4), the data must be coded in ASCII and terminated with an ASCII End of File (EOF) character (byte value = \$1A).

An example program using the handler shown in Figure 5-3 is listed in Figure 5-4. This example writes a series of ASCII characters to disk then reads the file. Since an \$1A EOF terminator is used, the written file can be examined using the UTILITY=L DOS function. Constants are used for the file name, disk number and side number in this example.

#### 5.6 DOS ERROR REPORTING

The DOS reports an error condition and value indicating the cause of the error whenever improper disk operation is detected. These errors may occur due to reasons such as improper disk drive operation, incorrect disk drive/FDC module connection, bad disk media, and operator error (e.g., wrong disk installed, wrong file name specified, wrong disk drive specified, attempted use of unformatted disk, etc.).

The error reporting format is:

DISK ERROR=

where:

XX = DOS detected errors plus seek error from the FDC device on the FDC module.

YY = The contents of the FDC device Status Register (less SEEK error) on the FDC module (see Section 4.5).

The error definitions are described in Table 5-3.

The error reporting linkage is through two vectors, ERRVEC (vector before error handling) and FESCIV (vector after error handling)--see Section 5.7. These vectors are initialized upon cold reset and may be altered to link to user-provided additional or alternative error handling.

Note that the default error handler return is to the Monitor (to COMIN at \$A32E) after reporting an error code.

PAGE 0001 TEST OF DISK FILE HANDLERS FOR AIM 65

```

ADDR OBJECT SOURCE
      OUTALL=#E9BC
      INALL=#E993
      OUTPUT=#E97A
      OPENW=#F00
      OPENR=#F36
      CLOSEW=#F28
      CLOSER=#F4E
      EOF=#537
      CHAR =0
      **=#F80
;
; THIS TEST PROGRAM WRITES OUT TWO SECTORS OF NUMBERS TO
; TO THE DISK . THE FILE IS THEN READ AND DISPLAYED , ALL
; UNDER PROGRAM CONTROL . THE FILE ( NAME IS FROM FNAME )
; IS SUITABLE TO BE LISTED FROM DOS
;
; SET UP THE READ AND WRITE DISK PARAMETERS FIRST
;
0F80 A9 00 TEST LDA #0
; SET UP THE DRIVES
0F82 8D 00 05 STA #500 ;READ DRIVE
0F85 8D 01 05 STA #501 ;WRITE DRIVE
; SET UP THE SIDES
0F88 8D 03 05 STA #503 ;READ SIDE
0F8B 8D 04 05 STA #504 ;WRITE SIDE
; DOWNLOAD FILENAME FROM FNAME
0F8E A2 0A NAME LDX #10
0F90 BD 00 0F NAME1 LDA FNAME,X
0F93 9D 0E 05 STA #50E,X ;READ FILE NAME
0F96 9D 18 05 STA #518,X ;WRITE FILE NAME
0F99 CA DEX
0F9A 10 F4 BPL NAME1
;
; NEXT WRITE OUT A TEST FILE
;
0F9C 20 00 0F TESTWR JSR OPENW ;OPEN FILE
0F9F A2 07 ISSUE LDX #7
0FA1 A0 30 NEWLIN LDY #60
0FA3 A9 30 ZERO LDA #10
0FA5 85 00 STA CHAR
0FA7 A5 00 LOOP LDA CHAR
0FA9 C9 3A CMP #'1'
0FAB F0 F6 BEQ ZERO
0FAD 20 8C E9 WRITE JSR OUTALL
0FB0 E6 00 INC CHAR
0FB2 88 DEY
0FB3 D0 F2 BNE LOOP
0FB5 A9 00 LDA #100 ;CR
0FB7 20 8C E9 JSR OUTALL
0FBA CA DEX
0FB8 D0 E4 BNE NEWLIN
0FBD AD 37 05 END LDA EOF ;SHOULD BE #1A
0FC0 20 8C E9 JSR OUTALL
0FC3 20 28 0F DONE JSR CLOSEW

```

Figure 5-4. Example Program Using File Handler

PAGE 0002 TEST OF DISK FILE HANDLERS FOR AIM 65

```

ADDR OBJECT SOURCE
;
; NOW READ IN THE TEST FILE
;
0FC6 20 36 0F TESTRD JSR OPENR ;OPEN FILE
0FC9 20 93 E9 READ JSR INALL ;READ IN A CHARACTER
0FCC CD 37 05 CMP EOF ;SAME AS MARK WRITTEN
0FCF F0 06 BEQ EOFOUT
0FD1 20 7A E9 OUTWIT JSR OUTPUT
0FD4 4C C9 0F JMP READ
0FD7 20 4E 0F EOFOUT JSR CLOSER
0FDA 4C A1 E1 JMP #E1A1 ;COMIN1 ==>
;
; THE NAME OF THE FILE IS ...
0FDD 54 45 FNAME .BYT 'TESTFILE11'
; END
;
ERRORS=0000

```

Figure 5-4. Example Program Using File Handler (Cont'd)

Table 5-3. AIM 65 DOS 1.0 Error Definitions

Code	Error	Definition
01YY	SEEK ERROR	The FDC device did not locate the data on the disk.
02YY	Undefined	
04YY	DISK FULL	The disk is full, thus, no more data may be written on it.
08YY	DIRECTORY FULL	The directory is full, thus, no more file names may be entered.
10YY	FILE NAME NOT FOUND	The specified file name is not in the directory.
20YY	FILE NAME EXISTS	The specified file name already exists for an active file.
80YY	DIRECTORY ERROR	The directory terminator was not found within the directory limits.
XX01	BUSY	The FDC device is busy.
XX02	Undefined	
XX04	LOST DATA	The CPU did not respond to DRQ in one byte time.
XX08	CRC ERROR	If RECORD NOT FOUND error exists, an error was found in one or more ID fields, otherwise a data field error exists.
XX10	RECORD NOT FOUND	The desired track, sector, or side were not found.
XX20	RECORD TYPE/WRITE FAULT	On read record, this bit indicates the record-type code from the data field address mark (1=Deleted Data Mark, 0=Data Mark). On a write, this bit indicates a write fault.
XX40	WRITE PROTECT ERROR	Writing was attempted to a write protected disk.
XX80	NOT READY	The drive is not ready.

## 5.7 DOS VARIABLES

The DOS variables, located on page zero and five are listed in Table 5-4. Some of these variables are for internal use only (to store temporary values) while others are user-alterable and constant after initialization. The default values for the user-alterable variables, initialized by the sub-routine DOSSET, are also shown in Table 5-4.

Table 5-4. AIM 65 DOS 1.0 Variables

Addr (Hex)	Label	No. Bytes	Init Value	Definition
0D7	SRCBUF	2	-	DOS Source Buffer Vector
0D9	DSTBUF	2	-	DOS Destination Buffer Vector
500	FDISK	1	00	Buffer 1 Disk No.
501		1	00	Buffer 2 Disk No.
502	FSIDE	1	00	Buffer 1 Side No.
503		1	00	Buffer 2 Side No.
504	FDEN	1	00	Buffer 1 Density
505		1	00	Buffer 2 Density
506	FTRACK	1	00	Buffer 1 Track No.
507		1	00	Buffer 2 Track No.
508	FSECTR	1	00	Buffer 1 Sector No.
509		1	00	Buffer 2 Sector No.
50A	FBYTE	1	00	Buffer 1 Byte Count
50B		1	00	Buffer 2 Byte Count
50C	SECLEN	1	00	Buffer 1 Bytes/Sector
50D		1	00	Buffer 2 Bytes/Sector
50E	FNAMES	10	00 -> 00	Buffer 1 File Name
518		10	00 -> 00	Buffer 2 File Name
522	FINOUT	1	00	Buffer to use
523	CHAR	1	00	Character (Temporary) Storage
524	ERSTAT	1	00	Error Status
525	FILNM1	2	00 00	Temporary Storage
527	FILNM2	2	00 00	Temporary Storage
529	FILNM3	2	00 00	Temporary Storage
52B	RETCNT	1	00	Retry Count
52C	READSV	1	00	Temporary Storage
52D	LSTCHR	1	00	Last Character
52E	CLOUTV	2	00 00	Close Output Vector
530	INITFL	1	00	Density Ask Flag
531	BUFFER	1	06	Source Buffer Addr. High Byte*
532	ERRVEC	2	EB 84	Before Error Handler Vector*
534	FESCIV	2	56 E9	After Error Handler Vector*
536	ASMFLG	1	00	Assembly Flag
537	EOF	1	1A	End of File Marker

Table 5-4. AIM 65 DOS 1.0 Variables (Cont'd)

Addr (Hex)	Label	No. Bytes	Init Value	Definition
538	MENUVC	2	64 80	Before UTILITY Decode Vector*
53A	MENULS	2	8A 80	After UTILITY Decode Vector*
53C	NOVRFY	1	01	Format Disk No Verify Flag*
53D	LEN	2	-	Length Count for Read File
53F	ASMVEC	2	-	Assembler Check Vector

NOTE

\*User-alterable

1. The DOS 1.0 variables are initialized by calling the DOS routine DOSSET (\$8006).

## SECTION 6

### USING AIM 65/40 DOS VERSION 1.0

The AIM 65/40 Disk Operating System (DOS) Version 1.0 (A65/40-7090) integrates the RM 65 FDC primitive subroutines with fundamental file management functions for use on the AIM 65/40 Microcomputer. These ROM-based functions, contained in a 4K-byte R2332 ROM, are accessible from the operator through the I/O and Debug Monitor/Text Editor ROMs as well as language ROMs and from an application program. Being ROM-based, DOS operation may proceed immediately upon power turn-on without waiting for separate loading of the DOS into RAM.

Text and program source code may be written to, and read from, disk with the Editor List and Read commands, respectively. Similarly, binary data and program object code may be written to, and loaded from, disk using the Monitor Dump and Load commands, respectively. Files containing source and object code for application programs written in AIM 65/40 Assembler, BASIC, FORTH, and PL/65 languages are, therefore, supported.

DOS primary commands are selected by the operator at the Monitor command level. These commands invoke the following functions:

- Format a Disk
- List the Directory
- Backup a Disk
- List a File
- Delete a File
- Recover a File

Files are created automatically upon writing a file to disk. A file name and the disk drive number are operator entered in response to system prompts.

Disk read or write errors, both at the DOS and FDC device level, are reported upon detection. User-alterable variables allow changing of default values to application unique values.

## 6.1 INITIALIZATION

After installing the AIM 65/40 DOS ROM on the RM 65 FDC module, the DOS is ready for use. Be sure that address range \$8000-\$8FFF is selected for off-board operation on the AIM 65/40 SBC module.

The DOS is initialized automatically through the I/O ROM Auto-start linkage. The FDC module primitive subroutine variables are initialized, the DOS variables are set to default values (see Section 6.7), the IRQ vectors are initialized, the Monitor key decode linkage is altered (to allow DOS to decode the [ and ] keys), the user-defined I/O vectors are loaded for floppy disk I/O, and other temporary variables are initialized. Control then returns to the I/O ROM and subsequently to the Monitor command level. DOS commands may be entered when the Monitor prompt is displayed (the DOS may not be operated without the Monitor/Editor ROMs installed).

AIM 65/40 DOS 1.0 is designed primarily to support the AIM 65/40 Monitor/Editor and language functions in response to commands entered from the keyboard with messages directed to the display/printer. When used in this manner, the Monitor/Editor ROMs must be installed. The DOS may also be used to write and read a data file under program control (see Section 6.5). In this case, the Monitor/Editor ROMs are not required unless a Monitor subroutine is used.

### NOTE

The DOS uses two contiguous 256-byte buffers, the output buffer and the input buffer, to transfer data between RAM and a floppy disk (as well as for other intermediate data storage). The high byte of the output (or source) buffer is specified by the variable BUFFER (see Section 6.7). The input (or destination) buffer is located immediately above the output buffer. BUFFER is initialized by a cold reset to \$3E which locates the buffers at the top of 16K bytes (i.e., \$3E00-\$3FFF). If

### NOTE (Cont'd)

more RAM is available, you will probably want to move this buffer to the top of RAM, e.g., change the Buffer value to \$7E for 32K bytes of RAM.

## 6.2 DISK FORMAT

A floppy disk must be formatted in the AIM 65/40 DOS 1.0 format before it can be (see Section 6.3.3) used. The DOS disk format (illustrated in Figure 6-1) reserves track 0 for future use, uses track 1 and one-half of track 2 for the directory, and the rest of the tracks for sector header and data storage. The sectors are formatted in an interleaved manner. Files are stored sequentially on the disk.

Trk No.	Sect No.	Function	Single-Density			Double-Density		
			Byte No.	File No.		Byte No.	File No.	
				5"	8"		5"	8"
0	All	Reserved	1-128	-	-	1-256	-	-
1	1	File Entry	1- 16	1	1	1- 16	1	1
		File Status	1			1		
		00=Directory End						
		01=Active File						
		02=Deleted File						
		File Name	2- 11			2- 11		
		Start Track	12			12		
		Start Sector	13			13		
		Length-Low Byte	14			14		
		Length-High Byte	15			15		
		Null (00)	16			16		
		File Entries	17-128	2- 8	2- 8	17-256	2- 16	2- 16
2	2	File Entries	1-128	9-16	9- 16	1-256	17- 32	17- 32
	3- 8	File Entries	1-128	17-64	17- 64	1-256	33-128	33-128
	9-13	File Entries	1-128	-	65-104	1-256	-	129-208
	14-16	Free	1-128	-	-	1-256	-	-
2	All	Free	1-128	-	-	1-256	-	-

Figure 6-1. AIM 65/40 DOS 1.0 Disk Format

### 6.3 PRIMARY OPERATOR COMMANDS

The primary operator commands are selected by using one of two keys: [ and ]. The [ key directly commands the List Directory function, (see Section 6.3.1) while the ] key displays a UTILITY= prompt. One of five utility functions may then be commanded (see Sections 6.3.2 through 6.3.6). If an invalid key is pressed for a utility function, the utility menu is displayed, e.g.:

```
{]} UTILITY=<RETURN>
(F)ORMAT, (B)ACKUP, (L)IST, (D)EL, (R)EC
```

The <ESC> key is vectored through variable FESCIV (see Section 6.7) to jump to the Monitor command level. This vector is initialized upon cold reset and may be user-altered. The <ESC> key vector is also used to return control to the Monitor after reporting an error condition (see Section 6.6). Note that some other major functions reconfigure the Escape vector to return to their command level, e.g., the Text Editor and BASIC. In these cases, if the <ESC> key is pressed, the function in process will be terminated.

#### NOTE

Upon cold reset, the density selection corresponding to each disk drive is initially set to double. If a disk cannot be read at the selected density, DOS automatically switches to the other density, and attempts to read the disk again. If the read is successful, the selected density is saved for future reading from that disk drive (until a cold reset is performed). This process is essentially transparent to the operator except for a slight delay.

If DOS cannot read the disk at either density, the operator will be prompted to enter the disk density (e.g., DENSITY= ) upon the next disk

#### NOTE (Cont'd)

command. The DOS will then attempt to read the disk again at the commanded density (and, if necessary, the other density).

#### 6.3.1 List the Directory

The List Directory function displays the file name of all files recorded on the disk, the sector length of the files and the active/deleted status of each file. The number of free sectors, i.e., the amount of space left on the disk, is also reported.

To list the directory, press the [ key, then enter the disk drive number in response to the prompt.

Example:

To list the contents of the directory on disk drive No. 1:

```
{[]} DISK=1
FILE NAME      LEN S
LOGGER1.OBJ    269 A
TEXT1          193 A
TEXT2          26  A
SEC LEFT= 178
```

The data output rate may be altered by pressing the 0 key (fastest) through the 9 key (slowest) while the directory is being listed, or by using the Monitor @ command prior to the List Direct command.

#### 6.3.2 Backup a Disk

The Backup Disk function copies all the active files from one disk to another disk. Since deleted files on the input disk are not copied to the output disk, the newly created disk is essentially compressed (i.e., deleted files removed) to provide additional free space.

To backup a disk, press the ] key to request the utilities prompt, then press the B key. Enter the disk drive number of the disk to copy from, then the disk drive number of the disk to copy to, in response to prompts.

Example:

To copy a disk on disk drive No. 1 to a disk on disk drive No. 2:

```
{}} UTILITY=B DISK=1
DISK=2
```

### 6.3.3 Format a Disk

The Format Disk function initializes a disk to prepare it for subsequent use (see Section 6.2). A new disk, i.e., an unformatted disk, must be formatted before a file can be written upon it. It may also be desired to initialize a previously formatted disk to remove all existing files before re-use.

The Format Disk function also verifies proper operation of the disk media before file storage use. The disk is first initialized by writing sector headers, gaps and CRC data, then a \$EA byte pattern to all sectors. The directory is next initialized to all \$00 bytes. If the NOVRFY flag (see Section 6.7) is non-zero (default value), the sector headers are verified, otherwise control returns to the Monitor command level. If the NOVRFY flag is zero, the sector headers are not checked, allowing a faster formatting of the disk.

To format a disk, press the ] key to request the utilities prompt, then press the F key. Enter the desired disk number and density character when requested. The disk number may vary from 1 to 4 for single-sided disks and from 1 to 8 for double-sided disks (i.e., each side is treated as a separate drive number with odd numbers being the front side and even numbers being the back side). The density character may be either S (for single-density) or D (for double-density).

Since the initialization completely overwrites the disk, a command verification prompt, "ARE YOU SURE?", is displayed before continuing, to prevent inadvertent reformatting. Press Y to continue, or any other key to terminate the command.

The WAIT message is displayed during disk initialization, followed by the Monitor command prompt upon completion. The number of free sectors and bytes is dependent upon disk size and density while the initialization time is dependent upon the state of the NOVRFY flag:

Disk Size	Density Code	Approx. Format Time (Min:Sec)		Free Sectors	Free Bytes
		Verify NOVRFY≠0	No Verify NOVRFY=0		
5"	S	1:45	0:20	535	68,480
5"	D	1:45	0:20	535	136,960
8"	S	5:00	0:30	1962	251,136
8"	D	5:00	0:30	1962	502,272

Example:

Initialize a 5" disk in disk drive No. 1 to single-density then list the directory to check it.

```
{}} UTILITY= F DISK=1 DENSITY=S
ARE YOU SURE?Y
WAIT

{}} DISK=1
FILE NAME LEN S
SEC LEFT= 535
```

#### NOTE

1. An RM 65 DMA module (RM65-5104) must be used to write and read double-density on 8" disks.
2. If an RM 65 DMA module is used (with either 5" or 8" disk drives), refer to Section A.3 for operating instructions.

#### 6.3.4 List a File

The List File function lists the contents of a file from a disk to another peripheral device. If the peripheral is another disk, a file may be copied from one disk to another.

To list a file, press the ] key to request the utilities prompt, then press the L key. Enter the file name, disk drive number, and output device code as requested. Respond to output device code subprompts.

Example 1:

To list a text file to the display/printer:

```
{}} UTILITY=L FILE=TEXT3 DISK=1
OUT=<RETURN>
```

Example 2:

To list an object code file from disk drive No. 1 to disk drive No. 2:

```
{}} UTILITY=L FILE=PROG1.OBJ DISK=1
OUT=F FILE=PROG1.OBJ DISK=2
```

The data output rate to the display/printer may be altered by pressing the 0 key (fastest) through the 9 key (slowest) while the file is being listed, or by using the Monitor @ command prior to the List File command.

The listing may be suspended by pressing the <SPACE> bar, then resumed by pressing the <SPACE> bar again (or one of the other keys).

#### 6.3.5 Delete a File

The Delete File Function changes a file from the active (A) state to the deleted (D) state. In the deleted state, the file cannot be accessed by file name by any function except Recover File (see Section 6.3.5).

To delete a file, press the ] key to request the utilities prompt, then press the D key. Enter the file name and disk drive number in response to prompts. The Monitor prompt will be displayed upon completion. List the directory to verify the file deletion.

Example:

To delete file TEXT3 (currently active) from a disk on disk drive No. 1:

```
{}} UTILITY=D FILE=TEXT3 DISK=1
{}} DISK=1
FILE NAME  LEN S
TEXT1      193 A
TEXT2       94 A
TEXT3       63 D
SEC LEFT= 238
```

A deleted file may be returned to the active state by using the Recover File function.

#### 6.3.6 Recover a File

The Recover File function changes a file from the deleted (D) state to the active (A) state. In the active state, the file can be accessed by file name by other DOS functions.

To recover a file, press ] key to request the utilities prompt, then press the R key. Enter the file name and disk drive number in response to prompts. The Monitor prompt will be displayed upon completion. List the directory to verify file recovery.

Example:

To recover file TEXT3 (currently deleted) from disk drive  
No. 1:

```
{]} UTILITY=R FILE=TEXT3 DISK=1
{]} DISK=1
FILE NAME  LEN  S
TEXT1      193  A
TEXT2       44  A
TEXT3       63  A
SEC LEFT= 238
```

### 6.3.7 Extension of UTILITY Functions

The DOS links to the UTILITY functions through two indirect jump vectors. These two vectors, MENUVC and MENULS, are user-alterable and may be changed to provide alternative linkage and functions. Upon depression of the ] key, the DOS jumps through the before menu vector, MENUVC, to check for the five default command characters. If the command character is an F, B, L, D or R, the appropriate processing is performed (see Section 6.3.2 through 6.3.6). If the command character is not one of these letters, the DOS jumps through the after menu vector, MENULS, to display the command menu then redisplay the "UTILITY=" prompt.

Either one or both of these vectors may be altered to meet application dependent requirements.

## 6.4 FILE HANDLING UNDER OPERATOR CONTROL

A file may be written to, or read from, a disk under operator control by pressing F in response to the OUT=, or IN=, prompt displayed by other major functions, e.g., Monitor, Editor or language. Enter the file name and disk number in response to subprompts.

### 6.4.1 Using the AIM 65/40 Monitor

The Monitor Dump (D) function can be used to output machine code from memory to a disk file in either ASCII or binary

format, while the Monitor Load (L) function can be used to read the file from a disk into RAM (see Sections 4.8.2 and 4.8.1, respectively, in the AIM 65/40 System User's Manual).

#### a. Dumping a File in ASCII

Example:

Dump machine code, in ASCII, from addresses \$0800 through \$08FF to file PG1\*A on disk drive No. 1:

```
{D}
FROM=0800 TO=08FF OFFSET=0000 MORE?N
TYPE=A OUT=F FILE=PG1*A DISK=1
```

#### b. Dumping a File in Binary

Example:

Dump machine code, in binary, from addresses \$0800 through \$08FF to file PG1\*B on disk drive No. 1:

```
{D}
FROM=0800 TO=08FF OFFSET=0000 MORE?N
TYPE=B DUMP SYMBOLS?N OUT=F FILE=PG1*B DISK=1
```

#### c. Reading a File

Example:

Load file PG1\*B from disk drive No. 1 into RAM:

```
{L} OFFSET=0000 IN=F FILE=PG1*B DISK=1
```

### 6.4.2 Using the AIM 65/40 Editor

The Editor List (L) function can be used to output ASCII text from the Editor Text Buffer in RAM to a disk file while the Read (R) function can be used to read the file into the Text Buffer (see Sections 5.4.9 and 5.4.1, respectively, in the AIM 65/40 Systems User's Manual).

a. Writing a File

Example:

Write all the text in the Text Buffer to file PG1 on disk drive No. 1:

```
= {T}
= {L} /. OUT=F FILE=PG1    DISK=1
```

b. Reading a File

Example:

Read file PG1 from disk drive No. 1 into the Text Buffer:

```
= {R} IN=F FILE=PG1    DISK=1
*END*
```

6.4.3 Using the AIM 65/40 Assembler

Source code may be read from a disk file into the AIM 65/40 assembler and/or object code may be written to a disk file from the assembler during the assembly process. If the object code output format is binary, the symbol table may also be saved on the disk file.

NOTE

The 8 key (Re-enter Assembler command) may not be used to assemble to/from disk.

Example 1:

Assemble source code from memory (i.e., the Text Buffer) and write the generated object code in ASCII to file PG1\*A on disk drive No. 2:

```
{7}ASSEMBLER V1.0
FROM=1800 TO=1FFF
IN=M
OBJ TO MEM?N
OUT=F FILE=PG1*A    DISK=2
TYPE=A
LIST?N OUT=<RETURN>
PASS 1
PASS 2

DONE
ERRORS=0000
```

Example 2:

Assemble source code from file PG1 on disk drive No. 1 and do not generate object code.

```
{7}ASSEMBLER V1.0
FROM=1800 TO=1FFF
IN=F FILE=PG1    DISK=1
OBJ TO MEM?N
OUT=X
LIST?N OUT=<RETURN>
PASS 1
PASS 2

DONE
ERRORS=0000
```

Example 3:

Assemble source code from file PG1 on disk drive No. 1 and write the generated object code and symbol table, in binary, to file PG1\*BS on disk drive No. 1:

```
{7}ASSEMBLER V1.0
FROM=1800 TO=1FFF
IN=F FILE=PG1    DISK=1
OBJ TO MEM?N OFFSET=0000
OUT=F TYPE=B DUMP SYMBOLS?Y FILE=PG1*BS    DISK=1
LIST?N OUT=<RETURN>
PASS 1
PASS 2

DONE
ERRORS=0000
```

**NOTE**

When assembling from a source file on disk and an error occurs (such as a File Not Found error), the AIM 65/40 must be reset to resume normal disk operation.

**6.4.4 Using AIM 65/40 BASIC**

The AIM/65/40 BASIC SAVE and LOAD functions may be used to write and read a BASIC application program, to and from a disk file, respectively (see Sections 4.3.10 and 4.3.5 in the AIM 65/40 BASIC User's Manual).

**a. Saving a File**

Example:

Save the BASIC program in RAM to file PGM3 on disk drive No. 1:

```
SAVE
OUT=F FILE=PGM3      DISK=1
```

**b. Loading a File**

Example:

Load a BASIC program into RAM file PGM3 on disk drive No. 1:

```
LOAD
IN=F FILE=PGM3      DISK=1
```

**6.4.5 Using AIM 65/40 FORTH**

The source code for an application program written in AIM 65/40 FORTH may be formatted to edit in the AIM 65/40 Text Editor or to input in the FORTH screen format. Refer to Sections 5.4.9

and 5.4.1 in the AIM 65/40 System User's Manual to write source code from the Text Editor to disk or to read source code from disk into the Text Editor. Refer to Section 12 in the AIM 65/40 FORTH User's Manual for FORTH screen format usage.

**6.4.6 Using PL/65**

The source code for an application program written in AIM 65/40 PL/65 may be read from disk into the PL/65 compiler and/or the generated object code (i.e., in 6502 assembly language) may be written to disk (see Section 2.6 in the AIM 65/40 PL/65 User's Manual).

**a. Write a File**

Compile source code from memory (i.e., the Text Buffer) and write the output assembler code to file PGM2L on disk drive No. 2:

```
{5}AIM 65/40 PL/65 V1.1
IN=M OUT=F FILE=PGM2L      DISK=1
PASS(1 OR 2)?2
ERRORS=00
```

**b. Read a File**

Compile source code from file PGM2L on disk drive No. 1 and output the generated assembly language to memory:

```
{5}AIM 65/40 PL/65 V1.1
IN=F FILE=PGM2L      DISK=1
OUT=M
PASS(1 OR 2)?2
ERRORS=00
```

**6.5 FILE HANDLING UNDER PROGRAM CONTROL**

Often it is desirable to store and/or retrieve data from a floppy disk entirely under program control, e.g., when collecting data in an unattended installation. While this can be easily done by writing a program which calls the primitive

routines (see Section 4), use of the primitive routines alone do not support the directory format used by AIM 65/40 DOS 1.0. DOS compatible data files written under program control allow easy examination and handling under operator control using DOS utility functions. This section describes a method to write data to, and read data from, a floppy disk in a format that is compatible with the AIM 65/40 DOS 1.0 file structure. To operate the DOS 1.0 under program control, the AIM 65/40 Monitor must be installed.

### 6.5.1 Writing and Reading Data Files

The AIM 65/40 DOS 1.0 file structure allows the writing and reading of data, to and from disk, independent of the data meaning (e.g., program source code, program object code, or symbol table) or data format (e.g., ASCII coding or machine code). The assembly listing in Figure 6-2 details a DOS compatible file handler. This handler contains both an output driver (to write a file to disk) and an input driver (to read a file from disk). The DOS subroutines and variables used by these drivers are listed in Tables 6-1 and 6-2, respectively.

To write a file, first set up the file name, disk number and side number, then call the write open subroutine (OPENW). Next, pass data a byte at a time to the disk file by loading each byte into the Accumulator and calling the I/O ROM OUTALL subroutine. When all data bytes have been output, call the write close subroutine (CLOSEW) to close the file. The CLOSEW subroutine automatically outputs the additional fill bytes (value=\$00) to complete the last sector.

To read a file, first set up the file name, disk number and side number, then call the read open subroutine (OPENR). Next, read data from the disk file, a byte at a time, by calling the I/O ROM INALL subroutine and storing the read byte in the Accumulator into memory. When all data bytes have been received, call the read close file (CLOSER) to close the file.

```

PAGE 0001  ROUTINES TO READ AND WRITE FILES FOR AIM 65/40
ADDR OBJECT  SOURCE
;
; FDC ADDRESSES ( 5-18-82 )
DISK2=$885B
DISK1=$884A
TRKSK=$840F
CLOPEN=$823A
FOFIN=$81A8
CLSOUT=$8288
CLSIN=$8221
; FDC VARIABLES
FINOUT=$522
DSTBUF=$D9
SRCBUF=$D7
BUFFER=$52F
;
; AIM 65/40 VARIABLES
OUTFLG=$274
INFLG=$273
;
; **$4000
;
; TO OPEN A WRITE FILE , SET UP DISK, SIDE AND
; NAME BEFORE CALLING THIS OPEN ROUTINE:
; DISK = $501 - NUMBER 0-7
; SIDE = $503 - 0 FOR FIRST SIDE
; FILENAME = $518 - 10 CHARACTERS PADDED WITH #20
; DENSITY = $505 - 0 FOR DOUBLE (OPTIONAL)
;
4000 A2 01  OPENW  LDX #1          ; USE WRITE BUFFER
4002 8E 22 05  STX FINOUT
4005 20 58 88  JSR DISK2          ; TURN ON WRITE DRIVE
4008 A9 00  LDA #0          ; SET UP DESTINATION BUFFER
400A 85 D9  STA DSTBUF
400C AC 2F 05  LDY BUFFER
400F C8  INY          ; WRITES USE BUFFER+1
4010 84 DA  STY DSTBUF+1
4012 20 DF 84  JSR TRKSK          ; RESTORE WITH DENSITY CHECK
4015 20 3A 82  JSR CLOPEN         ; OPEN THE FILE
4018 A3 10  LDA ##10        ; POINT OUTALL TO FLOPPY
401A 8D 74 02  STA OUTFLG
401D 60  RTS
;
; TO WRITE DATA , CALL OUTALL WITH BYTE THE IN A
; THE USER DRIVER WILL WRITE 128/256 BYTES AT A TIME
;
; TO CLOSE THE WRITE FILE
401E 4C E8 82  CLOSEW JMP CLSOUT      ; CLOSE DISK FILE

```

Figure 6-2. Program Control File Handler



Table 6-2. Commonly Used AIM 65/40 DOS 1.0 Variables

Label	Address (Hex)	Function
BUFFER	52F	MSB of pointer to DOS Buffer 1.
CURCMD	4AE	Last command executed by FDC device.
DSTBUF	0D9	Pointer to DOS Write Buffer.
ERSTAT	524	DOS error status.
ERRVEC	530	Pointer to the DOS error processing routine.
FBYTE	50A	Buffer 1 current byte (Read).
	50B	Buffer 2 current byte (Write).
FESCIV	532	Pointer to the DOS escape processing routine.
FDISK	500	Buffer 1 disk number (Read).
	501	Buffer 2 disk number (Write).
FSIDE	502	Buffer 1 side number (Read).
	503	Buffer 2 side number (Write).
FDEN	504	Buffer 1 density (Read).
	505	Buffer 2 density (Write).
FNAMES	50E-517	Buffer 1 file name (Read) - 10 characters.
	518-521	Buffer 2 file name (Write) - 10 characters.
FINOUT	522	Buffer indicator (0=Buffer 1, 1=Buffer 2).
LEN	52C	Number of remaining sectors (Read).
STFLG	4AD	FDC Device Status Register image.
SRCBUF	0D7	Pointer to DOS Read Buffer.
STORIO	55B	Disable (if Z=1) or re-enable all IRQ interrupt sources except for the FDC module.

If an error occurs during writing or reading using this handler, the IRQ interrupts are re-enabled, the disk drive motors are turned off, and control is returned to the AIM 65/40 Monitor command level. The source of the error can be determined by examining the command type (CURCMD), the FDC status register (STFLG), and the DOS error status (ERSTAT).

Note that the address of the error handler is placed in ERRVEC and the address to jump to after the error has been serviced is placed in FESCIV. If errors are to be handled under program control, these vectors must be loaded appropriately. The error handler must isolate and service the error as required. If the disk operation is to be terminated, the IRQ interrupts must be re-enabled (JSR ION) and the disk drive motors turned off (JMP FESC).

#### 6.5.2 DOS Compatibility Considerations

The user is responsible for the meaning and coding of all data in files created under program control, including the definition and inclusion of control bytes. Sometimes the files are handled by knowing exactly how many bytes are written to a file, and therefore how many to expect upon reading it (which can be detected by logically ANDing LEN, LEN+1 and FBYTE then checking for a zero result. Any \$00 pad bytes used to fill the last sector can be ignored.

Another approach is to include one or more special characters to indicate the end of a file. The end of file character(s) must not conflict with valid data values using their approach, however.

In order for a data file to be listed using the DOS list utility function (UTILITY=L, see Section 6.3.4), the data must be coded in ASCII and terminated with an ASCII End of File (EOF) character (byte value = \$1A).

An example program using the handler shown in Figure 6-2 is listed in Figure 6-3. This example writes a series of ASCII characters to disk then reads the file. Since an \$1A EOF terminator is used, the written file can be examined using the UTILITY=L DOS function. Constants are used for the file name, disk number and side number in this example.

### 6.6 DOS ERROR REPORTING

The DOS reports an error condition and value indicating the cause of the error whenever improper disk operation is detected. These errors may occur due to reasons such as improper disk drive operation, incorrect disk drive/FDC module connection, bad disk media, and operator error (e.g., wrong disk installed, wrong file name specified, wrong disk drive specified, attempted use of unformatted disk, etc.).

The error reporting format is:

DISK ERROR=

where:

XX = DOS detected errors plus seek error from the FDC device on the FDC module.

YY = The contents of the FDC device Status Register (less SEEK error) on the FDC module (see Section 4.5).

The error definitions are described in Table 6-3.

The error reporting linkage is through two vectors, ERRVEC (vector before error handling) and FESCIV (vector after error handling)--see Section 6.7. These vectors are initialized upon cold reset and may be altered to link to user-provided additional or alternative error handling.

Note that the default error handler return is to the Monitor (to ESCIN at \$A32E) after reporting an error code.

```

PAGE 0001  TEST OF DISK FILE HANDLERS FOR AIM 65/40
ADDR OBJECT  SOURCE
                                OUTALL=$F32B
                                INALL=$F233
                                OUTPUT=$F352
                                OPENW=$4000
                                OPENR=$4021
                                CLOSEW=$401E
                                CLOSER=$4039
                                CHAR  =0
                                **$3000
                                ;
                                ; THIS TEST PROGRAM WRITES OUT TWO SECTORS OF NUMBERS TO
                                ; THE DISK . THE FILE IS THEN READ AND DISPLAYED , ALL
                                ; UNDER PROGRAM CONTROL . THE FILE ( NAME IS FROM FNAME )
                                ; IS SUITABLE TO BE LISTED FROM DOS .
                                ;
                                ; SET UP THE READ AND WRITE DISK PARAMETERS FIRST
                                ;
3000 A9 00  TEST  LDA #0
                                ; SET UP THE DRIVES
3002 8D 00 05  STA $500  ; READ DRIVE
3005 8D 01 05  STA $501  ; WRITE DRIVE
                                ; SET UP THE SIDES
3008 8D 03 05  STA $503  ; READ SIDE
300B 8D 04 05  STA $504  ; WRITE SIDE
                                ; DOWNLOAD FILENAME FROM FNAME
300E A2 0A  NAME  LDX #10
3010 8D 64 30  NAME1 LDA FNAME,X
3013 9D 0E 05  STA $50E,X  ; READ FILE NAME
3016 9D 18 05  STA $518,X  ; WRITE FILE NAME
3019 CA  DEX
301A 10 F4  BPL NAME1
                                ;
                                ; NEXT WRITE OUT A TEST FILE
                                ;
301C 20 00 40  TESTWR JSR OPENW  ; OPEN FILE
301F A2 07  ISSUE LDX #7
3021 A0 3C  NEWLIN LDY #60
3023 A9 30  ZERO  LDA #0
3025 85 00  STA CHAR
3027 A5 00  LOOP  LDA CHAR
3029 C9 3A  CMP #1
302B F0 F6  BEQ ZERO
302D 20 2B F3  WRITE JSR OUTALL
3030 E6 00  INC CHAR
3032 88  DEY
3033 D0 F2  BNE LOOP
3035 A9 0D  LDA ##0D  ; CR
3037 20 2B F3  JSR OUTALL
303A CA  DEX
303B D0 E4  BNE NEWLIN
303D A9 1A  END  LDA ##1A  ; SHOULD BE EOF
303F 20 2B F3  JSR OUTALL
3042 20 1E 40  DONE JSR CLOSEW

```

Figure 6-3. Example Program Using File Handler

Table 6-3. AIM 65/40 DOS 1.0 Error Definitions

```

PAGE 0002 TEST OF DISK FILE HANDLERS FOR AIM 65/40
ADDR OBJECT SOURCE
;
; NOW READ IN THE TEST FILE
;
3045 20 21 40 TESTRD JSR OPENR ; OPEN FILE
3048 20 33 F2 READ JSR INALL ; READ IN A CHARACTER
304B C9 1A CMP #$1A ; EOF MARK
304D F0 0F BEQ EOFOUT
304F C9 00 CMP #$00 ; ALWAYS SEND LF WITH CR
3051 D0 05 BNE OUTWIT
3053 20 52 F3 JSR OUTPUT
3056 A9 0A LDA #$0A
3058 20 52 F3 OUTWIT JSR OUTPUT
305B 4C 48 30 JMP READ
305E 20 39 40 EOFOUT JSR CLOSER
3061 4C 14 A3 JMP $A314 ; COMIN1 ==>
;
; THE NAME OF THE FILE IS ...
3064 54 45 FNAME .BYT 'TESTFILE01'
.END

ERRORS=0000
    
```

Code	Error	Definition
01YY	SEEK ERROR	The FDC device did not locate the data on the disk.
02YY	Undefined	
04YY	DISK FULL	The disk is full, thus, no more data may be written on it.
08YY	DIRECTORY FULL	The directory is full, thus, no more file names may be entered.
10YY	FILE NAME NOT FOUND	The specified file name is not in the directory.
20YY	FILE NAME EXISTS	The specified file name already exists for an active file.
80YY	DIRECTORY ERROR	The directory terminator was not found within the directory limits.
XX01	BUSY	The FDC device is busy.
XX02	Undefined	
XX04	LOST DATA	The CPU did not respond to DRQ in one byte time.
XX08	CRC ERROR	If RECORD NOT FOUND error exists, an error was found in one or more ID fields, otherwise a data field error exists.
XX10	RECORD NOT FOUND	The desired track, sector, or side were not found.
XX20	RECORD TYPE/WRITE FAULT	On read record, this bit indicates the record-type code from the data field address mark (1=Deleted Data Mark, 0=Data Mark). On a write, this bit indicates a write fault.
XX40	WRITE PROTECT ERROR	Writing was attempted to a write protected disk.
XX80	NOT READY	The drive is not ready.

Figure 6-3. Example Program Using File Handler (Cont'd)

## 6.7 DOS VARIABLES

The DOS variables, located on page zero and five are listed in Table 6-4. Some of these variables are for internal use only (to store temporary values) while others are user-alterable and constant after initialization. The default values for the user-alterable variables, initialized by a cold reset if the AIM 65/40 monitor is installed or the subroutine DOSSETT, are also shown in Table 6-4.

Table 6-4. AIM 65/40 DOS 1.0 Variables

Addr (Hex)	Label	No. Bytes	Init Value	Definition
0D7	SRCBUF	2	-	DOS Source Buffer Vector
0D9	DSTBUF	2	-	DOS Destination Buffer Vector
500	FDISK	1	00	Buffer 1 Disk No.
501		1	00	Buffer 2 Disk No.
502	FSIDE	1	00	Buffer 1 Side No.
503		1	00	Buffer 2 Side No.
504	FDEN	1	00	Buffer 1 Density
505		1	00	Buffer 2 Density
506	FTRACK	1	00	Buffer 1 Track No.
507		1	00	Buffer 2 Track No.
508	FSECTR	1	00	Buffer 1 Sector No.
509		1	00	Buffer 2 Sector No.
50A	FBYTE	1	00	Buffer 1 Byte Count
50B		1	00	Buffer 2 Byte Count
50C	SECLN	1	00	Buffer 1 Bytes/Sector
50D		1	00	Buffer 2 Bytes/Sector
50E	FNAMES	10	00 -> 00	Buffer 1 File Name
518		10	00 -> 00	Buffer 2 File Name
522	FINOUT	1	00	Buffer to use
523	CHAR	1	00	Character Temporary Storage
524	ERSTAT	1	00	Error Status
525	FILNM1	2	00 00	Temporary Storage
527	FILNM2	2	00 00	Temporary Storage
529	FILNM3	2	00 00	Temporary Storage
52B	RETCNT	1	00	Retry Count
52C	LEN	2	00 00	Length Count for Read File
52E	INITFL	1	00	Density Ask Flag*
52F	BUFFER	1	3E	Source Buffer Addr High Byte*
530	ERRVEC	2	CB 84	Before Error Handler Vector*
532	FESCIV	2	34 A3	After Error Handler Vector*
534	BEEPC	3	4C 67 F4	JMP BEEP*
537	SPACE	3	4C 7A F3	JMP BLANK*
53A	CRLFC	3	4C 5E AE	JMP CRLF*
53D	CRLOWC	3	4C 8F F3	JMP CRLOW*
540	OUTCL	3	4C 4E A5	JMP EBCRCL*

Table 6-4. AIM 65/40 DOS 1.0 Variables (Cont'd)

Addr (Hex)	Label	No. Bytes	Init Value	Definition
543	FNAMEC	3	4C 2A FA	JMP FNAME*
546	NOUTC	3	4C AC F3	JMP NOUT*
549	NUMAC	3	4C A4 F3	JMP NUMA*
54C	OUTALC	3	4C 2B F3	JMP OUTALL*
54F	OUTPUC	3	4C 52 F3	JMP OUTPUT*
552	RCHEK	3	4C 7D B0	JMP VARCHK*
555	REDOC	3	4C 9B F2	JMP REDOUT*
558	WHERE	3	4C B9 AE	JMP WHEREO*
55B	STORIO	3	8D 80 FF	STA PRIRTY*
55E		1	60	RTS*
55F	MENUVC	2	62 80	Before UTILITY Decode Vector*
561	MENULS	2	85 80	After UTILITY Decode Vector*
563	NOVRFY	1	01	Format Disk No Verify Flag*

NOTES

\*User-alterable

1. The DOS 1.0 variables are initialized by a cold reset (refer to the AIM 65/40 System User's Manual) only if the AIM 65/40 Monitor is installed.

APPENDIX A

DISK DRIVE CONFIGURATION

This appendix describes the actual configuration of some of the more popular disk drives. The drives included are:

Shugart SA400  
Shugart SA450  
Pertec FD200  
Shugart SA800

A.1 5" DISK DRIVE CONFIGURATION

To operate the RM 65 FDC module in single- or double-density mode with these 5" drives, perform the following steps:

- a. Set up switch S1 for common bank operation with both the Program ROM and I/O enabled.

S1-1 = OPEN  
S1-2 = OPEN  
S1-3 = CLOSED  
S1-4 = CLOSED

- b. Set up the jumpers for no DMA, no precompensation (which is not required for 5" double density) and the number of heads.

E1 = EITHER  
E2 = EITHER  
E3 = B for only single sided drives  
(SA400 or FD200)  
A for double sided drives  
(SA450)

- c. Set up the standard/mini-floppy headers (JB1, JB2) for mini-floppy drives (shown in Figure 2-2a).

Table 6-4. AIM 65/40 DOS 1.0 Variables (Cont'd)

Addr (Hex)	Label	No. Bytes	Init Value	Definition
543	FNAMEC	3	4C 2A FA	JMP FNAME*
546	NOUTC	3	4C AC F3	JMP NOUT*
549	NUMAC	3	4C A4 F3	JMP NUMA*
54C	OUTALC	3	4C 2B F3	JMP OUTALL*
54F	OUTPUC	3	4C 52 F3	JMP OUTPUT*
552	RCHEK	3	4C 7D B0	JMP VARCHK*
555	REDOC	3	4C 9B F2	JMP REDOUT*
558	WHERE	3	4C B9 AE	JMP WHEREO*
55B	STORIO	3	8D 80 FF	STA PRIRTY*
55E		1	60	RTS*
55F	MENUVC	2	62 80	Before UTILITY Decode Vector*
561	MENULS	2	85 80	After UTILITY Decode Vector*
563	NOVRFY	1	01	Format Disk No Verify Flag*

NOTES

\*User-alterable

1. The DOS 1.0 variables are initialized by a cold reset (refer to the AIM 65/40 System User's Manual) only if the AIM 65/40 Monitor is installed.

APPENDIX A

DISK DRIVE CONFIGURATION

This appendix describes the actual configuration of some of the more popular disk drives. The drives included are:

Shugart SA400  
Shugart SA450  
Pertec FD200  
Shugart SA800

A.1 5" DISK DRIVE CONFIGURATION

To operate the RM 65 FDC module in single- or double-density mode with these 5" drives, perform the following steps:

- a. Set up switch S1 for common bank operation with both the Program ROM and I/O enabled.

S1-1 = OPEN  
S1-2 = OPEN  
S1-3 = CLOSED  
S1-4 = CLOSED

- b. Set up the jumpers for no DMA, no precompensation (which is not required for 5" double density) and the number of heads.

E1 = EITHER  
E2 = EITHER  
E3 = B for only single sided drives  
(SA400 or FD200)  
A for double sided drives  
(SA450)

- c. Set up the standard/mini-floppy headers (JB1, JB2) for mini-floppy drives (shown in Figure 2-2a).

- d. Ensure that the FDC module has both +5 VDC and +12 VDC. The +5 VDC can be verified by reading the Program ROM (\$8000) and comparing this with the Program Listing. The +12 VDC can be verified by reading the FDC device registers (\$8F00 - \$8F03). If these four locations are \$FF, +12 VDC is not reaching the module.

The 5" disk drives must also be prepared as described in the following sections.

#### A.1.1 Shugart SA400

The Shugart SA400 is a single-sided 5" drive. As factory configured, the SA400 is ready to operate as the only drive. If additional drives are used, make the following changes to each:

**CAUTION**

Refer to the SA400 user instructions accompanying your drives to confirm this setup.

- a. Install the termination resistor pack only in the last drive on the Floppy Disk cable.
- b. Open the MX shunt.
- c. Short the HL shunt and open the MH shunt.
- d. Short only one--1, 2 or 3 (Drive select 4 is not available) and open the remaining two positions to assign a unique number to each drive.

Ensure that both +5 VDC and +12 VDC are present at power connector J2.

#### A.1.2 Shugart SA450

The Shugart SA450 is a double-sided 5" drive. As factory configured, this drive is ready to operate as the only drive. If additional drives are used, make the following changes to each:

**CAUTION**

Refer to the SA450 user instructions accompanying your drives to confirm this setup.

- a. Install the termination resistor pack only in the last drive on the Floppy Disk cable.
- b. Open the MX shunt.
- c. Open the MM and the MS shunts.
- d. Short only one--1, 2, 3 or 4 and open the remaining three positions to assign a unique number to each drive.

Ensure that both +5 VDC and +12 VDC are present at power connector J2.

#### A.1.3 Pertec FD200

The Pertec FD200 is a single-sided 5" drive (although this drive allows the front and back side of the disk to be used, this is a non-standard format that is only compatible with Pertec drives).

If additional drives are used, the following changes must be made to each:

**CAUTION**

Refer to the FD200 user instructions  
accompanying your drives to confirm this  
setup.

- a. Install the termination resistor pack only in the last drive.
- b. Cut the PC etch between the pads at DC.
- c. Short only one--1, 2, 3 or 4 and open the remaining three switch positions to assign a unique number to each drive.
- d. Leave the head-load option in the standard configuration with W2 installed and W1 removed.

Ensure that both +5 VDC and +12 VDC are present on the power connector J3.

**A.2 8" DISK DRIVE CONFIGURATION**

To operate the RM65 FDC module in single- or double-density mode with 8" drives, perform the following steps:

- a. Set up switch S1 for common bank operation with both the Program ROM and I/O enabled.

S1-1 = OPEN  
S1-2 = OPEN  
S1-3 = CLOSED  
S1-4 = CLOSED

- b. Set up the jumpers for no DMA, no precompensation and the number of heads.

E1 = EITHER  
E2 = EITHER  
E3 = B for only single sided drives  
(SA800)  
A for double sided drives  
(SA850)

- c. Set up the standard/mini-floppy headers (JB1, JB2) for standard floppy drives (shown in Figure 2-2b).
- d. Ensure that the FDC module has both +5 VDC and +12 VDC (refer to Section A.1.d).

**A.2.1 Shugart SA800/801**

The Shugart SA800 is a single-sided 8" drive. Configure each SA800 drive as follows to use with the FDC module:

**CAUTION**

Refer to the SA800 user instructions  
accompanying your drives to confirm this  
setup.

- a. Short only one of the DS1, DS2, DS3 or DS4 jumpers and leave the other three open to assign a unique number to each drive.
- b. If only one drive is used, install jumpers at T3, T4, T5 and T6. If additional drives are used, remove jumpers at T3, T4, T5 and T6 from all drives except for the last one on the Floppy Drive cable.
- c. Remove jumper at 801.
- d. Install jumpers at A, B, HL, I, L, R, RI, RR, S, T1, WP, X, Z, 32 and 800.

- e. Leave all remaining positions open.

Ensure that +5, +24 and -5 VDC are all present at connector J5, and 110 VAC is applied to connector J4.

### A.3 DMA Configuration

The RM 65 FDC module will operate in all modes--5" or 8", single- or double-density--with the RM 65 DMA Controller module (RM65-5104). To operate in the 8" double-density mode this module is required, but all modes will show performance improvement when the DMA module is used. For an operating description and complete configuration details, refer to the RM 65 DMA Controller User's manual. To configure the DMA module for operation, perform the following steps:

- a. Set up switches S1-1 to S1-8 for the desired base address of the DMA module. This module address can be any page of the RM 65 memory map not being used, where the switches reflect the selected page. Thus, for a typical base address of \$7F00;

S1-1 to S1-7 = CLOSED  
S1-8 = OPEN

- b. Set up switches S1-9 and S1-10 for common bank operation for use with AIM 65 and AIM 65/40 single bank systems;

S1-9 = OPEN  
S1-10 = OPEN

- c. Set up the DMA interrupt request to be sent over the BIRQ/ interrupt line;

E1 = A

The DMAC module requires RAM that is directly addressable from the RM 65 bus--such as an RM 65 8K Static RAM module--to supplement the AIM 65 or AIM 65/40 on-board RAM. To configure the 8K RAM module for operation, perform the following steps:

- a. Set up switches S1-1 to S1-8 for the desired base address of the RAM module. This module address can be any two 4K byte blocks of the RM 65 memory map, with each of the four switches assigning a block. Thus, for typical base addresses of \$1000 and \$2000;

S1-1 = CLOSED  
S1-2 to S1-5 = OPEN  
S1-6 = CLOSED  
S1-7 to S1-8 = OPEN

- b. Set up switches S2-1 and S2-2 for common bank operation;

S2-1 = OPEN  
S2-2 = OPEN

- c. Set up the RAM for no write protection;

S2-3 = OPEN  
S2-4 = OPEN

The FDC module does not have any special configuration. The DMA request selection jumper can be set for either BDRQ1/ or BDRQ2/.

In addition to configuring the module, the DMA variables in the FDC firmware must also be set:

- a. The DMA module address must be written into the variable DMAADR. Thus, for a typical base address of \$7F00;

DMAADR (\$4B9) = \$7F

- b. The variable DMAFLG must set up with a DMA module enable (bit 0), DMA transfer source bank address (bit 7), and DMA transfer destination bank address (bit 6). Thus, for single bank operation in Bank 0;

DMAFLG (\$4BA) = \$01

When the DMA module enable bit of DMAFLG is set, many of the firmware routines make use of the module for data transfers. This use is automatic--each routine will set up the DMA module and initiate the transfers as required. No software changes are required to take advantage of the DMA module.

NOTE

When using the RM 65 DMA Controller module with the FDC module, the disk read and write buffers (RDBUF, WRBUF) must be on the supplemental RM 65 RAM module. For file management, the source and destination buffers (set up from BUFFER) must be on the supplemental RM 65 RAM module.

APPENDIX B

FORTH AND THE RM 65 FDC MODULE

This appendix describes the actual code used to interface the RM 65 FDC module with AIM 65 and AIM 65/40 FORTH. This example uses a single 5" drive with one side and double density recording. The example is entered into the text editor and compiled using the SOURCE word. For a detailed description of the code words, refer to the AIM 65/40 FORTH User's Manual, Section 12. There are seven major words created which supplement the use of the FDC module:

INIT  
Initializes the FDC module and turns on drive one, side one in single density mode.

MOTORON  
Turns on the drive from SRCDSK, side from SRCSID, and density from SRCDEN.

MOTOROFF  
Turns off the selected drive.

FORMAT  
Initializes the disk in the selected drive. A formatted disk will have all sectors filled with \$E5, which on the AIM 65 is displayed as a blank, and printed as "%". On the AIM 65/40, this is displayed as a blinking "E.", and printed as "e".

WIPE  
n ---  
Clears screen n by filling it with null characters (\$00).

LIST  
n ---  
Lists screen n as 16 numbered lines (0 to F) with 64 characters each.

When the DMA module enable bit of DMAFLG is set, many of the firmware routines make use of the module for data transfers. This use is automatic--each routine will set up the DMA module and initiate the transfers as required. No software changes are required to take advantage of the DMA module.

NOTE

When using the RM 65 DMA Controller module with the FDC module, the disk read and write buffers (RDBUF, WRBUF) must be on the supplemental RM 65 RAM module. For file management, the source and destination buffers (set up from BUFFER) must be on the supplemental RM 65 RAM module.

APPENDIX B

FORTH AND THE RM 65 FDC MODULE

This appendix describes the actual code used to interface the RM 65 FDC module with AIM 65 and AIM 65/40 FORTH. This example uses a single 5" drive with one side and double density recording. The example is entered into the text editor and compiled using the SOURCE word. For a detailed description of the code words, refer to the AIM 65/40 FORTH User's Manual, Section 12. There are seven major words created which supplement the use of the FDC module:

INIT  
Initializes the FDC module and turns on drive one, side one in single density mode.

MOTORON  
Turns on the drive from SRCDSK, side from SRCSID, and density from SRCDEN.

MOTOROFF  
Turns off the selected drive.

FORMAT  
Initializes the disk in the selected drive. A formatted disk will have all sectors filled with \$E5, which on the AIM 65 is displayed as a blank, and printed as "%". On the AIM 65/40, this is displayed as a blinking "E.", and printed as "e".

WIPE  
n ---  
Clears screen n by filling it with null characters (\$00).

LIST  
n ---  
Lists screen n as 16 numbered lines (0 to F) with 64 characters each.

P

n ---

Places the text following EDIT (up to 64 characters) into line n on the current screen (i.e., the last screen accessed with LIST or WIPE).

Other words that are present in FORTH and are also useful in the creation and execution of source code include:

LOAD

n ---

Compiles source code into the dictionary starting at screen n, line 0 and continuing until a ;S is encountered. The ;S should be within 4 lines of the last line of source code. More than one sequential screen is loaded by using --> to point to the next screen.

EMPTY-BUFFERS

Marks all RAM block buffers as empty.

FLUSH

Writes out any updated RAM block buffers to the disk.

Figure B-1 lists the FDC interface program for AIM 65 FORTH.

Figure B-2 lists the AFDC interface program for AIM 65/40 FORTH.

```

< AIM 65 - FORTH DOUBLE DENSITY DISK EXAMPLE >
< USE WITH FDC FIRMWARE DOS V1.0 - 5/26/82 >
HEX FORGET TASK 500 ALLOT < MOVE ABOVE FDC AND DOS RAM >
1 CONSTANT S# < ONLY 1 SCREEN NEEDED >
100 UB/BUF ! < 256 FOR DOUBLE DENSITY >
4 UB/SCR ! < 4 FOR DOUBLE DENSITY >
LIMIT B/BUF 4 + B/SCR * S# * - UFIRST ! < TOP OF RAM >
0 OFFSET ! < OFFSET NOT NEEDED WITH 1 DRIVE >
FIRST DUP USE ! PREV ! < SET UP FIRST BUFFER >
EMPTY-BUFFERS < CLEAR OUT THE BUFFER AREA >
CODE INIT1 XSAVE STX, 886C JSR, < CALL INIT >
< SET DRIVE PARAMETERS IN UDRV, USIDE, & UDEN >
0 # LDA, 4B4 STA, < DRIVE ONE INTO UDRV >
0 # LDY, 4B2 STY, < SIDE ONE INTO USIDE >
0 # LDX, 4BC STX, < DOUBLE DENSITY INTO UDEN >
8C53 JSR, < MOTON > XSAVE LDX, NEXT JMP, END-CODE
: INIT 8B0D A400 ! < SET UP IROHAN > INIT1 ;
: SIZEOK OVER 230 < ; < 16 SECTOR * 35 TRACK >
: BBUF DUP 4F1 < RDBUF > ! 4F3 < WRBUF > ! ;
: T&S SWAP 10 /MOD ; < LEAVE TRACK & SECTOR >
: DERROR CR ." DISK ERROR - " ; < PRINT ERROR MESSAGE >
CODE SEEK XSAVE STX, TOP LDA, 8938 JSR, < CALL SEEK >
XSAVE LDX, 99 # AND, PUSH0A JMP, END-CODE
CODE DREAD XSAVE STX, TOP LDA, 8D29 JSR, < CALL R0SEC >
XSAVE LDX, 8D # AND, PUSH0A JMP, END-CODE
CODE DWRITE XSAVE STX, TOP LDA, 8D61 JSR, < CALL WRTEC >
XSAVE LDX, FD # AND, PUSH0A JMP, END-CODE
: DATA < FETCH A BYTE > ROT BBUF T&S SEEK DUP
IF DERROR ." SEEK A=" . < SEEK ERROR > ELSE DROP
THEN DROP 1+ < SECTOR 1 TO 16 > SWAP
IF DREAD DUP IF DERROR ." READ A=" . < READ ERROR >
ELSE DROP THEN < DO NOTHING >
ELSE DWRITE DUP IF DERROR ." WRITE A=" . < ERROR >
ELSE DROP THEN < DO NOTHING > THEN DROP ;
: DISK SIZEOK IF DATA
ELSE CR ." BLOCK TOO LARGE ERROR " ABORT THEN ;
< DISK CFA UR/W ! < STORE INTERFACE WORD >
< UTILITIES THAT MUST BE AVAILABLE TO USER >
CODE FORMAT XSAVE STX, 896D JSR, < CALL FORMAT >
XSAVE LDX, NEXT JMP, END-CODE
CODE MOTOROFF XSAVE STX, 8CF7 JSR, < CALL MOTOFF >
XSAVE LDX, NEXT JMP, END-CODE
CODE MOTORON XSAVE STX, 8CF7 JSR, < CALL MOTOFF >
4B4 LDA, < UDRV > 4B2 LDY, < USIDE >
4BC LDX, < UDEN > 8C53 JSR, < CALL MOTON >
XSAVE LDX, NEXT JMP, END-CODE
: P SCR @ < LINE > OVER SWAP BLANKS < CLEAR OUT LINE >
0 WORD < PARSE TEXT > HERE COUNT 40 MIN < 64 CHAR LINES >
ROT SWAP MOVE < MOVE TEXT > UPDATE < MARK BUFFER > ;
: LIST DUP CR ." SCR # " . < PRINT SCREEN AND SAVE > SCR !
10 0 DO CR I 3 .R SPACE I SCR @ .LINE LOOP CR ;
: WIPE B/SCR * B/SCR BOUNDS < SCREEN # TO BLOCK RANGE >
DO I BLOCK B/BUF BLANKS UPDATE LOOP FLUSH ;
: TASK ; < THROUGH WITH CODE > FINIS

```

Figure B-1. AIM 65 FORTH Floppy Disk Example

## APPENDIX C

## FDC FIRMWARE PROGRAM LISTING

```

< AIM 65/40 -- FORTH DOUBLE DENSITY DISK ROUTINES >
< USE WITH RM 65 FDC FIRMWARE DOS V1.0 - 5/18/82 >
HEX FORGET TASK < FDC AND DOS RAM $4A0-$563 >
4 CONSTANT S# < ONLY 1 SCREEN NEEDED >
100 UB/BUF ! < 256 FOR DOUBLE DENSITY >
4 UB/SCR ! < 4 FOR DOUBLE DENSITY >
LIMIT 200 - DUP ULIMIT ! B/BUF 4 + B/SCR * S# * - UFIRST ! < 16K OF RAM >
0 OFFSET ! < OFFSET NOT NEEDED WITH 1 DRIVE >
FIRST DUP USE ! PREV ! < SET UP FIRST BUFFER >
EMPTY-BUFFERS < CLEAR OUT THE BUFFER AREA >
CODE INIT1 XSAVE STX, 8860 JSR, < CALL INIT >
  < SET DRIVE PARAMETERS IN UDRV, USIDE, & UDEN >
  0 # LDA, 484 STA, < DRIVE ONE INTO UDRV >
  0 # LDY, 482 STY, < SIDE ONE INTO USIDE >
  0 # LDX, 480 STX, < DOUBLE DENSITY INTO UDEN >
  8C53 JSR, < MOTON > XSAVE LDX, NEXT JMP, END-CODE
  ! INIT FD46 4A0 ! < UIRDEM > IRQOUT > 8BED 22B
  ! < SET UP IRDMAN INIT1 >
  ! SIZEOK OVER 220 < > < 16 SECTOR * 35 TRACK >
  ! BBUF DUP 409 < RDBUF > ! 4CB < WRTEUF > !
  ! T&S SWAP 10 /MOD < > < LEAVE TRACK & SECTOR >
  CODE SEEK XSAVE STX, TOP LDA, 8938 JSR, < CALL SEEK >
  XSAVE LDX, 99 # AND, PUSH0A JMP, END-CODE
  CODE DREAD XSAVE STX, TOP LDA, 8D29 JSR, < CALL R0SEC >
  XSAVE LDX, 80 # AND, PUSH0A JMP, END-CODE
  CODE DWRITE XSAVE STX, TOP LDA, 8D61 JSR, < CALL WRTSEC >
  XSAVE LDX, FD # AND, PUSH0A JMP, END-CODE
  ! INTDIS FF FF80 C! < MASK OUT ALL IRQ BUT FDC >
  ! INTENB 00 FF80 C! < RESTORE THE IRQ MASK >
  ! DERROR INTENB CR ! " DISK ERROR - " < RECOVER & PRINT >
  ! DATA < FETCH A BYTE > ROT BBUF T&S SEEK DUP
  ! IF DERROR ! " SEEK A=" ! < SEEK ERROR > INTDIS ELSE DROP
  ! THEN DROP 1+ < SECTOR 1 TO 16 > SWAP
  ! IF DREAD DUP ! IF DERROR ! " READ A=" ! < READ ERROR >
  ! ELSE DROP ! THEN < DO NOTHING >
  ! ELSE DWRITE DUP ! IF DERROR ! " WRITE A=" ! < ERROR >
  ! ELSE DROP ! THEN < DO NOTHING > THEN DROP
  ! DISK SIZEOK ! IF INTDIS DATA INTENB
  ! ELSE CR ! " BLOCK TOO LARGE ERROR " ABORT ! THEN ;
  / DISK CFA UR/W ! < STORE INTERFACE WORD >
  < UTILITIES THAT MUST BE AVAILABLE TO USER >
  CODE FORMAT XSAVE STX, 890D JSR, < CALL FORMAT >
  XSAVE LDX, PUSH0A JMP, END-CODE
  ! FORMAT INTDIS FORMAT INTENB
  CODE MOTOROFF XSAVE STX, 8CF7 JSR, < CALL MOTOFF >
  XSAVE LDX, NEXT JMP, END-CODE
  CODE MOTORON XSAVE STX, 8CF7 JSR, < CALL MOTOFF >
  484 LDA, < UDRV > 482 LDY, < USIDE >
  48C LDX, < UDEN > 8C53 JSR, < CALL MOTON >
  XSAVE LDX, NEXT JMP, END-CODE
  ! P SCR @ < LINE > OVER SWAP BLANKS < CLEAR OUT LINE >
  ! 0 WORD < PARSE TEXT > HERE COUNT 40 MIN < 64 CHAR LINES >
  ! ROT SWAP MOVE < MOVE TEXT > UPDATE < MARK BUFFER >
  ! LIST DUP CR ! " SCR # " ! < PRINT SCREEN AND SAVE > SCR !
  ! 10 0 DO CR I 3 .R SPACE I SCR @ .LINE LOOP CR
  ! WIPE B/SCR * B/SCR BOUNDS < SCREEN # TO BLOCK RANGE >
  ! DO I BLOCK B/BUF BLANKS UPDATE LOOP FLUSH
  ! TASK < THROUGH WITH CODE > FINIS

```

Figure B-2. AIM 65/40 FORTH Floppy Disk Example

```

0003 ;;;;;;;;;;;;;;;;;;;;;;;;;;
0004 ;
0005 ;       RM 65 FDC FIRMWARE ;
0006 ;
0007 ;       REVISION 2.0      ;
0008 ;       MAY 18, 1982     ;
0009 ;
0010 ;
0011 ;
0012 ;;;;;;;;;;;;;;;;;;;;;;;;;;
    
```

```

0014 ; ROCKWELL INTERNATIONAL
0015 ; ELECTRONIC DEVICES DIVISION
0016 ; P.O. BOX 3669
0017 ; 3310 MIRALOMA AVENUE
0018 ; ANAHEIM CAL. U.S.A. 92803
    
```

```

0020 ; MODULE REGISTER EQUATES
0022 8F00 FCOMR = $8F00 ; COMMAND REG
0023 8F00 FSTAR = FCOMR ; STATUS REG
0024 8F01 FCYLR = FCOMR+1 ; CYLINDER NBR
0025 8F02 FSECR = FCOMR+2 ; SECTOR NBR
0026 8F03 FDAR = FCOMR+3 ; DATA REG
0027 8F04 DSTAR = FCOMR+4 ; DRIVE STATUS REG
0028 8F04 DCONR = FCOMR+4 ; DRIVE CONTROL REG
0029 8F15 FSTOP = FCOMR+$15 ; STOP CPU

0031 ; COMMAND CODES
0033 ; TYPE 1:
0034 0000 RECOD = $00 ; RESTORE
0035 0010 SKCOD = $10 ; SEEK A CYL
0036 ; TYPE 2:
0037 00B0 RSCOD = $B0 ; READ A SECTOR
0038 00A0 WSCOD = $A0 ; WRITE A SECTOR
0039 ; TYPE 3:
0040 00C4 RACOD = $C4 ; READ AN ID FIELD
0041 00E4 RTCOD = $E4 ; READ A TRACK
0042 00F4 WTCOD = $F4 ; WRITE A TRACK
0043 ; TYPE 4:
0044 00D0 FICOD = $D0 ; FORCE AN INTERRUPT

0046 ; COMMAND FLAGS
0048 0004 V = $04 ; VERIFY AFTER COMMAND
0049 0010 M = $10 ; MULTPL SECTORS(TO END OF TRK)
0050 0008 S = $08 ; 2ND SIDE SELECT
0051 0002 C = $02 ; SIDE COMPARE ENABLE
0052 0004 E = $04 ; 15 MS DELAY FOR HEAD SETTLING

0054 ; DMA REG ADDRESSES
0056 9000 DMASRC = $9000 ; SOURCE CONTROL
0057 9001 DMASR1 = $9001 ; MS SOURCE ADDR
0058 9002 DMASR2 = $9002 ; LS SOURCE ADDR
0059 9004 DMADST = $9004 ; DEST CONTROL
0060 9005 DMADS1 = $9005 ; MS DEST ADDRESS
0061 9006 DMADS2 = $9006 ; LS DEST ADDRESS
0062 9008 DMACNT = $9008 ; BYTE CNT CONTROL
0063 9009 DMACT1 = $9009 ; MS BYTE COUNT
0064 900A DMACT2 = $900A ; LS BYTE COUNT
0065 900C DMACMD = $900C ; CMD/STATUS
    
```

```
0067 0000          *=$DB
0069              ;PAGE ZERO INDIRECTS
0071 00DB          PTR      *==+2
0072 00DD          DMAPTR  *==+2
0074 00DF          *=$4A0
0076 04A0          IRGOUT  *==+2          ; IRG EXIT ADDRESS
0077 04A2          FLAG    *==+1          ; FLAG=ERR DETECT FLAG1= RD/WRT
0078 04A3          CNTR    *==+1          ; COUNTER
0079 04A4          TEMP    *==+2          ; TEMP STORAGE
0080 04A6          TEMP2   *==+1          ; TEMP STOR
0081 04A7          MSC     *==+1
0082 04A8          LSC     *==+1
0083 04A9          NCYL    *==+1
0084 04AA          NSEC    *==+1
0085 04AB          CURSEC  *==+1          ; CURR SECT NBR
0086 04AC          LSECR   *==+1          ; LAST SECT TO READ OR WRITTEN
0087 04AD          STFLG   *==+1          ; STATUS FLG
0088 04AE          CURCMD  *==+1          ; CURRENT CMD BEING EXECUTED
0089 04AF          WRTFLG  *==+1          ; WRITE CMD FLAG
0090 04B0          SELFLG  *==+1
0091 04B1          MDFLG   *==+1
0092 04B2          USIDE   *==+1          ; SIDE
0093 04B3          UCYL    *==+1          ; CYLINDER
0094 04B4          UDRV    *==+1          ; DRIVE
0095 04B5          CURCYL  *==+4          ; CURR CYL FOR EACH DRV (0-3)
0096 04B7          DMAADR  *==+1          ; DMA MSB I/O ADDR
0097 04BA          DMAFLG  *==+1          ; DMA FLAG
0098 04BB          FORFLG  *==+1          ; WRITE TRACK FLAG
0099 04BC          *==+2
0101              ; THE FOLLOWING MUST BE IN ORDER--SEE INIT
0102 04BE          DBLCNT  *==+1          ; DOUBLE DENSITY CTR (255)
0103 04BF          SNGLCT  *==+1          ; SINGLE DENSITY CTR (127)
0104 04C0          NBYTE   *==+1          ; 128
0105 04C1          NHEAD   *==+1          ; 1
0106 04C2          CYL5    *==+1          ; 35
0107 04C3          CYL8    *==+1          ; 77
0108 04C4          SECTK5  *==+1          ; 16
0109 04C5          SECTK8  *==+1          ; 26
0110 04C6          HDEL    *==+1          ; DELAY FOR HEAD LOAD(30)
0111 04C7          SFLAG   *==+1          ; SIDE CMP & DATA MARK (C+E)
0112 04CB          TUNDEL  *==+1          ; DELAY FOR NON-SHUGART TUNNEL
0113 04C9          RDBUF   *==+2          ; USER BUFR FOR READ (1000)
0114 04CB          WRBUF   *==+2          ; USER BUFR FOR WRT (2000)
0115 04CD          RATE    *==+1          ; STEPPING RATE (03)
0116 04CE          RETRY   *==+1          ; NBR OF RETRIES
0117 04CF          PADB    *==+1
0118 04CF          TABLE5 *--1; SECTOR          INTERLEAVING TABLE
                                *==+16
0119 04D0          TABLE8 *--1; FOR          5 AND 8 INCH RESPECTIVELY
0120 04DF          *==+26
0121 04E0
```

0122 ;END OF ORDER

```

0124 04FA          *=$886C
0126 886C  AD 00 8F  INIT  LDA  FSTAR  ; INIT STATUS REG UPON PWR UP
0127 886F  AD 04 8F          LDA  DSTAR
0128 8872  A9 00          LDA  #00
0129 8874  A2 1B          LDX  #DBLCNT-FLAG-1
0130 8876  9D A2 04  INIT1 STA  FLAG, X  ; CLEAR RAM BUT NOT IRQOUT
0131 8879  CA          DEX
0132 887A  10 FA          BPL  INIT1
0133 887C  A2 3B          LDX  #SETUP-INTTBL-1
0134 887E  8D 88 88  INIT3 LDA  INTTBL, X  ; TABLE OF DEFAULT VALUES
0135 8881  9D BE 04          STA  DBLCNT, X  ; AREA WHERE VARIABLES ARE LOC
0136 8884  CA          DEX  ; NEXT VALUE
0137 8885  10 F7          BPL  INIT3
0138 8887  60          RTS
0139 8888          INTTBL
0140 8888  FF          .BYT  255, 127, 128, 1, 33, 77, 16, 26, 30
0141 8891  06          .BYT  C+E
0142 8892  64          .BYT  100
0143 8893  00 3E          .WOR  $3E00, $3F00
0144 8897  07          .BYT  7, 5, $E5
0145 889A  06          T5  .BYT  6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 0, 2, 3, 4, 5
0146 88AA  06          T8  .BYT  6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18
0147 88B7  13          .BYT  19, 20, 21, 22, 23, 24, 25, 26, 0, 2, 3, 4, 5
    
```

```

0150 88C4  78          SETUP SEI
0151 88C5  D8          CLD
0152 88C6  A9 00          LDA  #00
0153 88C8  8D 88 04  STA  FORFLG  ; DISABLE INTERRUPTS
0154 88CB  8D AD 04  STA  STFLG   ; MAKE SURE DECIMAL MODE CLEAR
0155 88CE  AD B1 04  LDA  MOFLG   ; RESET DRIVE STATUS FLAG
0156 88D1  F0 2B  STA  STFLG   ; ZERO FOR NON-FORMAT COMMAND
0157 88D3  AD B0 04  LDA  MOFLG   ; IS MOTOR ON?
0158 88D6  F0 26  BEQ  ERCOND  ; MOTOR NOT ON
0159 88D8  20 16 8C  LDA  SELFLG  ; HAS DRIVE BEEN SELECTED?
0160 88DB  20 04 89  BEQ  ERCOND  ; NO, DRIVE MUST BE SELECTED
0161 88DE  A9 80  JSR  FORM5   ; MAKE SURE IT'S OFF
0162 88E0  AE C2 04  JSR  SETIRQ  ; ENABLE FDC IRQ & CPU STOP
0163 88E3  AC C4 04  LDA  #128   ; BYTES/SECT FOR FM
0164 88E6  2C 04 8F  LDX  CYL5    ; NBR CYL/DRV
0165 88E9  10 01  LDY  SECTK5  ; SECTORS/TRACK FOR 5"
0166 88EB  0A          BIT  DSTAR   ; CK DENSITY(BIT 7) & DRV (B6)
0167 88EC  8D C0 04  BPL  STUP1   ; FM
0168 88EF  70 06  ASL  A          ; MAKE 256
0169 88F1  AE C3 04  STUP1 STA  NBYTE
0170 88F4  AC C5 04  BVS  INITO   ; 5" DRIVE
0171 88F7  8C AA 04  LDX  CYL8
0172 88FA  8E A9 04  LDY  SECTK8
0173 88FD  60          INITO STY  NSEC
          STX  NCYL
          RTS
0175 88FE  A9 80          ERCOND LDA  #80   ; MOTOR-NOT-ON ERROR
0176 8900  8D AD 04  STA  STFLG
0177 8903  60          RTS
0179 8904  AD B1 04  SETIRQ LDA  MOFLG
0180 8907  09 40          ORA  #40
0181 8909  8D 04 8F  STA  DCONR   ; ENABLE FDC IRQ LATCH
0182 890C  60          RTS          ; ENABLE CPU STOP
    
```

```

0184 890D 20 CA 8D   FORMAT JSR   SETIT   ;CK MOTOR ON & SETUP PARAMETER
0185 8910 A9 00       LDA     #00
0186 8912 8D B3 04   STA     UCYL
0187 8915 20 B2 89   FORMO JSR   SEEKNV  ;SEEK TO DESIRED CYL
0188 8918 29 99       AND     #$99
0189 891A D0 1B       BNE     FORERR  ;MUST BE ZERO OR ELSE ERROR
0190 891C 20 94 89   JSR     WRTRK   ;WRITE A TRACK
0191 891F 29 FD       AND     #$FD
0192 8921 D0 14       BNE     FORERR
0193 8923 20 3B 89   JSR     FMTSK   ;SEEK CURRENT CYL
0194 8926 29 99       AND     #$99
0195 8928 D0 0D       BNE     FORERR
0196 892A EE B3 04   INC     UCYL
0197 892D AD B3 04   LDA     UCYL   ;CURRENT CYL
0198 8930 CD A9 04   CMP     NCYL   ;HAVE ALL CYL'S BEEN DONE?
0199 8933 D0 E0       BNE     FORMO   ;CONT TO FORMAT
0200 8935 A9 00       LDA     #00
0201 8937 60       FORERR RTS
  
```

```

0203 ;ENTER WITH TRACK NBR IN ACC 'A'
0205 8938 8D B3 04   SEEK   STA     UCYL   ;SAVE THE CYL NBR
0206 893B 20 CA 8D   FMTSK JSR     SETIT   ;MOTOR ON & SETUP PARAMETERS
0207 893E AE B4 04   LDX     UDRV
0208 8941 8D B5 04   LDA     CURCYL,X ;CURR CYL FOR DRV
0209 8944 8D 01 8F   STA     FCYLR
0210 8947 AD B3 04   LDA     UCYL   ;DESIRED CYL
0211 894A 9D B5 04   STA     CURCYL,X ;SV CURR CYL FOR DRV
0212 894D 8D 03 8F   STA     FDAR    ;PUT CYL INTO DATA REG
0213 8950 F0 04       BEG     RSTOR   ;SEEK TO CYL 0
0214 8952 A9 10       LDA     #SKCOD  ;SEEK TO A CYL
0215 8954 D0 05       BNE     REST    ;EXECUTE
0216 ;***
0217 8956 A0 01       RSTOR LDY     #01   ;RESET SECTOR TO ONE
0218 8958 8C 02 8F   STY     FSECR  ;A ALREADY CONTAINS RESET CODE
0219 895B 0D CD 04   REST  ORA     RATE ;STEPPING RATE
0220 895E 8D AE 04   STA     CURCMD ;SAVE CURR CMD
0221 8961 AD AF 04   LDA     WRTFLG ;WAS LAST CMD A WRITE?
0222 8964 10 0B       BPL     RESTEE  ;NO
0223 8966 4E AF 04   LSR     WRTFLG ;CLEAR WRT FLAG
0224 8969 AD CB 04   LDA     TUNDEL ;IS DRIVE A SHUGART?
0225 896C F0 03       BEG     RESTEE  ;YES
0226 896E 20 D9 8E   JSR     MICROD ;DELAY FOR TUNNEL ERASE
0227 8971 AD AE 04   RESTEE LDA  CURCMD
0228 8974 20 D6 8B   JSR     EXEC04

0230 8977 A9 01       RESTED LDA  #1
0231 8979 2C 00 8F   BIT     FSTAR
0232 897C D0 F9       BNE     RESTED
0233 897E AD AD 04   LSTFLG LDA  STFLG ;GET ERROR FROM INTERRUPT
0234 8981 60       RTS

0236 ;SEEK WITHOUT VERIFY
0238 8982 AD CD 04   SEEKNV LDA  RATE
0239 8985 4B       PHA
0240 8986 29 FB       AND     #$FB
0241 8988 8D CD 04   STA     RATE
0242 898B 20 3B 89   JSR     FMTSK
0243 898E 6B       PLA
0244 898F 8D CD 04   STA     RATE
0245 8992 D0 EA       BNE     LSTFLG ;ALWAYS GOES
  
```

```
0247           .MACRO W1 BYTE
0248 LDA #BYTE
0249 BIT FSTOP ; STOP CPU
0250 STA FDAR
0251           .ENDM

0253           .MACRO W2 BYTE, BCNT
0254 LDX #BCNT-1
0255 LDA #BYTE
0256 BIT FSTOP ; STOP CPU
0257 STA FDAR ; WRITE THE BYT
0258 DEX
0259 BNE *-7
0260 BIT FSTOP
0261 STA FDAR
0262           .ENDM

0264           .MACRO SW2 BYTE, BCNT
0265 LDX #BCNT
0266 LDA #BYTE
0267 BIT FSTOP
0268 STA FDAR
0269 DEX
0270 BNE *-7
0271           .ENDM

0273           .MACRO W3 BYTE, COUNT, DATA
0274 LDX #COUNT
0275 LDA #DATA
0276 BIT FSTOP ; STOP CPU
0277 STA FDAR ; WRITE THE BYTE
0278 LDA BYTE
0279 BIT FSTOP
0280 STA FDAR
0281 DEX ; COUNT IT
0282 BNE *-7
0283 BIT FSTOP
0284 STA FDAR
0285           .ENDM
0286           .MACRO SW3 BYTE, COUNT, DATA
0287 LDX #COUNT
0288 LDA #DATA
0289 BIT FSTOP
0290 STA FDAR
0291 LDA BYTE
0292 BIT FSTOP
0293 STA FDAR
0294 DEX
0295 BNE *-7
0296           .ENDM

0298           .MACRO WA BYTE
0299 LDA BYTE ; PICK UP VALUE
0300 BIT FSTOP ; STOP CPU
0301 STA FDAR ; WRITE BYTE
```

```
0302           .ENDM

0304           .MACRO WY
0305 BIT FSTOP ; STOP CPU
0306 STY FDAR ; WRITE BYTE
0307           .ENDM
```

```

0309          ;SEEK MUST HAVE BEEN COMPLETED PRIOR
0310          ;TO CALLING THIS ROUTINE.

0312 8994    20 CA 8D    WRTRK JSR   SETIT    ;MOTOR ON & SETUP PARAMETERS
0313 8997    A2 04    WRTRK1 LDX   #>TABLEB
0314 8999    2C 04 8F          BIT   DSTAR
0315 899C    70 04          BVS   FIVE    ;FIVE INCH DRIVE
0316 899E    A9 DF          LDA   #CTABLEB ;INTERLEAVE TABLE
0317 89A0    D0 02          BNE   FPUT
0318          ;
0319 89A2    A9 CF          FIVE LDA   #CTABLE5
0320 89A4    85 DB    FPUT STA   PTR
0321 89A6    86 DC          STX   PTR+1
0322 89A8    3B          SEC          ;SET WRT FLAG
0323 89A9    6E AF 04          ROR   WRFLG
0324 89AC    A9 F4    WRTRK3 LDA   #WTCOD
0325 89AE    8D 8B 04          STA   FORFLG ;LET INTERRUPTS KNOW IT'S A WR
0326 89B1    A0 01          LDY   #01    ;SECTOR CTR
0327 89B3    2C 04 8F          BIT   DSTAR
0328 89B6    8D AE 04          STA   CURCMD
0329 89B9    8D 00 8F          STA   FCOMR
0330 89BC    30 03          BMI   WTMFM   ;1=DOUBLE DENSITY
0331 89BE    4C 04 8B          JMP   WTMFM   ;0=SINGLE DENSITY

0333          ;DOUBLE DENSITY
0334 89C1    70 41          WTMFM BVS   WT5MFM ;5" DRV DOUBLE DENSITY
0335          ;8" DRV DOUBLE DENSITY ----- NEED DMA !!!!!
0336 89C3          W2     $4E,80
0337 89D6          W2     00,12
0338 89E9          W2     $F6,3
0339 89FC          W1     $FC
0340 8A04          WT5MFM W2     $4E,50 ;GAP 1
0341          ;LOOP FOR EACH SECT
0342 8A17    A2 0B          LDX   ##B
0343 8A19    A9 00          LDA   #00
0344 8A1B    2C 15 8F          WSMFM BIT   FSTOP ;STOP CPU
0345 8A1E    8D 03 8F          STA   FDAR ;WRITE THE DATA
0346 8A21    CA          DEX
0347 8A22    D0 F7          BNE   *-7
0348 8A24    2C 15 8F          BIT   FSTOP
0349 8A27    8D 03 8F          STA   FDAR
0350 8A2A          W2     $F5,3
0351 8A3D          W1     $FE
0352 8A45          WA     UCYL ;TRACK
0353 8A4E          WA     USIDE ;SIDE
0354 8A57          WY          ;SECTOR NBR
0355 8A5D          W1     1 ;LENGTH
0356 8A65          W1     $F7
0357 8A6D          W2     $4E,22
0358 8A80          W2     00,12
0359 8A93          W2     $F5,3
0360 8AA6          W3     PADB,$FF,$FB ;256 BYTES/SECT
0361 8AC2          W1     $F7
0362 8ACA          W2     $4E,54 ;GAP 3
0363 8ADD    B1 DB          LDA   (PTR),Y ;GET NEXT SECTOR

```

```

0364 8ADF    F0 1B          BEQ   WTDOND ;YES, ALL SECTORS DONE
0365 8AE1    2C 15 8F          BIT   FSTOP
0366 8AE4    8E 03 8F          STX   FDAR
0367 8AE7    AB          TAY
0368 8AEB    A2 09          LDX   #9
0369 8AEA    A9 00          LDA   #00
0370 8AEC    2C 15 8F          BIT   FSTOP
0371 8AEF    8D 03 8F          STA   FDAR
0372 8AF2    4C 1B 8A          JMP   WSMFM ;NO

0374 8AF5    A0 FF          WTDONS LDY   #$FF
0375 8AF7    D0 02          BNE   WTDON
0376 8AF9    A0 4E          WTDOND LDY   #$4E
0377 8AFB    2C 15 8F          WTDON BIT   FSTOP
0378 8AFE    8C 03 8F          STY   FDAR
0379 8B01    4C FB 8A          JMP   WTDON

0381          ;SINGLE DENSITY
0382 8B04    70 22          WTMFM BVS   WT5FM ;5" DRV
0383          ;
0384 8B06          SW2   $FF,40
0385 8B13          SW2   00,6
0386 8B20          W1     $FC
0387 8B28          WT5FM SW2   $FF,26 ;GAP 1
0388          ;LOOP TO FORMAT EACH SECT
0389 8B35          WSMFM SW2   00,6
0390 8B42          W1     $FE
0391 8B4A          WA     UCYL ;TRACK
0392 8B53          WA     USIDE ;SIDE
0393 8B5C          WY          ;SECT NBR
0394 8B62          W1     0 ;LENGTH
0395 8B6A          W1     $F7
0396 8B72          SW2   $FF,11
0397 8B7F          SW2   00,6
0398 8B8C          SW3   PADB,$80,$FB ;128 BYTES OF DATA
0399 8BA2          W1     $F7
0400 8BAA          SW2   $FF,27 ;GAP 3
0401 8BB7    B1 DB          LDA   (PTR),Y ;GET NEXT SECT #
0402 8BB9    AB          TAY
0403 8BBA    F0 03          BEQ   WSMFM ;YES, ALL SECT DONE
0404 8BBC    4C 35 8B          JMP   WSMFM

0406 8BBF    4C F5 8A          WSMFM JMP   WTDONS

0408          ;5" INTERLEAVED SECTOR ORDER
0409          ;1, 6, 11, 16, 5, 10, 15, 4, 9, 14, 3, 8, 13, 2, 7, 12

0411          ;8" INTERLEAVED SECTOR ORDER
0412          ;1, 6, 11, 16, 21, 26, 5, 10, 15, 20, 25, 4, 9, 14, 19
0413          ;24, 3, 8, 13, 18, 23, 2, 7, 12, 17, 22

```

```

0415          ; EXECUTE TYPE 2 COMMAND
0417          ; NOTE: SELECTED SIDE MUST BE PUT INTO CMD FOR PROPER
0418          ;       HEADER SIDE VERIFICATION...

0420 8BC2    8D AE 04    EXEC02 STA  CURCMD ; TYPE 2 ENTRY
0421 8BC5    AD 04 BF    LDA  DSTAR  ; GET SELECTED SIDE
0422 8BC8    4A          LSR  A        ; SIDE INTO CARRY
0423 8BC9    AD AE 04    LDA  CURCMD ; RESTORE CMD
0424 8BCC    B0 02      BCS  EXEC01 ; SIDE 1
0425 8BCE    09 08      ORA  #S      ; SIDE 2
0426 8BD0    0D C7 04    EXEC01 ORA  SFLAG ; ONLY FOR TYPE 2
0427 8BD3    8D AE 04    EXEC03 STA  CURCMD ; ENTRY FOR TYPE 3 COMMANDS
0428 8BD6    8D 00 BF    EXEC04 STA  FCOMR  ; EXECUTE CMD TYPES 1 & 4
0429 8BD9    A0 00      LDY  #00     ; INIT DATA XFER POINTER

0431          ; THE FDC CHIP REQUIRES A DELAY AFTER BEING
0432          ; ISSUED A COMMAND. STOPPING THE CPU BEFORE
0433          ; THIS DELAY IS COMPLETE WILL CAUSE ERRORS
0434          ; 14 USEC FOR 8" DOUBLE
0435          ; 28 USEC FOR 8" SINGLE
0436          ; AND FOR 5" DOUBLE
0437          ; 56 USEC FOR 5" SINGLE

0439 8BDB    2C 04 BF    BIT  DSTAR  ; DELAY FOR STATUS REGISTER
0440 8BDE    30 05      BMI  DOUBLD ; GO IF DOUBLE DENSITY
0441 8BE0    A0 04      LDY  #4
0442 8BE2    88          WAITS1 DEY
0443 8BE3    D0 FD      BNE  WAITS1
0444 8BE5    50 05      DOUBLD BVC  BIGDSK ; GO IF 8 INCH DRIVES
0445 8BE7    A0 04      LDY  #4
0446 8BE9    88          WAITS2 DEY
0447 8BEA    D0 FD      BNE  WAITS2
0448 8BEC    60          BIGDSK RTS      ; COMMAND NOW BEING EXECUTED
    
```

```

0451          ; NOTE!!!! FLOPPY DISK CANNOT BE INTERRUPTED
0452          ;       DURING A FORMAT, WRITE OR READ OR
0453          ;       ELSE DATA LOST WILL OCCUR !!!!!

0455 8BED    48          IRQHAN PHA          ; SAVE ACC 'A'
0456 8BEE    AD AE 04    LDA  CURCMD ; WAS AN FDC CMD ISSUED?
0457 8BF1    F0 06      BEG  IRQRTN ; NO
0458 8BF3    AD 00 BF    LDA  FSTAR  ; IF IRQ IS FROM FDC
0459 8BF6    4A          LSR  A        ; IT WOULD BE IDLE
0460 8BF7    90 07      BCC  IRGFDC ; IRQ FROM FDC
0461 8BF9    AD 00 BF    IRQRTN LDA  FSTAR
0462 8BFC    68          PLA          ; RESTORE ACCUM
0463 8BFD    6C A0 04    JMP  (IRGOUT) ; DO INTERRUPT EXIT

0465 8C00    8A          IRGFDC TXA
0466 8C01    48          PHA
0467 8C02    20 1C 8C    JSR  FORM6
0468 8C05    AD BB 04    LDA  FORFLG ; SEE IF WRITE TRACK
0469 8C08    F0 08      BEG  NOTFOR
0470 8C0A    68          PLA
0471 8C0B    68          PLA
0472 8C0C    68          PLA
0473 8C0D    68          PLA
0474 8C0E    68          PLA
0475 8C0F    4C 7E 89    JMP  LSTFLG ; EXIT WITH ERRORS IN A
0476 8C12    68          NOTFOR PLA
0477 8C13    AA          TAX
0478 8C14    68          PLA
0479 8C15    40          RTI

0481 8C16    AD 00 BF    FORM5 LDA  FSTAR ; WAIT TILL CMD ENDS
0482 8C19    4A          LSR  A        ; HAS CMD ENDED?
0483 8C1A    B0 FA      BCS  FORM5 ; NO, STILL BUSY
0484 8C1C    AD BA 04    FORM6 LDA  DMAFLG ; USING DMA?
0485 8C1F    F0 0A      BEG  NODMA  ; NO
0486 8C21    98          TYA
0487 8C22    48          PHA          ; SAVE Y
0488 8C23    A0 0C      LDY  #<DMACMD ; COMMAND REGISTER
0489 8C25    A9 00      LDA  #00     ; HALT DMA
0490 8C27    91 DD      STA  (DMAPTR),Y
0491 8C29    68          PLA
0492 8C2A    A8          TAY
0493 8C2B    AD B1 04    NODMA LDA  MOFLG
0494 8C2E    29 BF      AND  #BF     ; DISABLE IRQ LATCH
0495 8C30    8D 04 BF    STA  DCONR  ; DISABLE CPU STOP
0496 8C33    AE AE 04    LDX  CURCMD
0497 8C36    8E A2 04    STX  FLAG   ; SAVE FOR EXTERNAL USE
0498 8C39    AD AD 04    LDA  STFLG ; IS STATUS FLG SET?
0499 8C3C    D0 09      BNE  FORM7 ; YES, RETURN STATUS TO CALLER
0500 8C3E    AD 00 BF    LDA  FSTAR
0501 8C41    E0 20      CPX  #20    ; TYPE 1?
0502 8C43    B0 02      BCS  FORM7 ; NO
0503 8C45    29 D9      AND  #D9    ; YES, REMOVE BAD BITS
0504 8C47    A2 00      FORM7 LDX  #00
0505 8C49    8E AE 04    STX  CURCMD ; CLR CURR CMD
    
```

```

0506 8C4C 29 FD      AND  ##FD
0507 8C4E 8D AD 04  STA  STFLG
0508 8C51 58        CLI
0509 8C52 60        RTS
    
```

```

0511      ; ENTER WITH DESIRED DRIVE (0-3) IN ACC AND
0512      ;   SET INDEX REG 'Y' TO SIDE.
0513      ;   SET INDEX REG 'X' WITH A
0514      ;       1 FOR SINGLE DENSITY
0515      ;       0 FOR DOUBLE DENSITY

0517 8C53 DB      MOTON CLD      ; CLEAR DECIMAL MODE
0518 8C54 48      PHA
0519 8C55 A9 20    LDA  ##20
0520 8C57 2C B1 04 BIT  MOFLG      ; MOTOR ON?
0521 8C5A D0 5F    BNE  SELCTD    ; YES, SO SELECT DRIVE
0522 8C5C 68      PLA
0523 8C5D 8D B4 04 STA  UDRV
0524 8C60 8C B2 04 STY  USIDE
0525 8C63 A9 20    LDA  ##20
0526 8C65 20 F0 BC JSR  SETFLG    ; SET FLAGS (MOTOR ON), A=##20
0527 8C68 20 A6 BC JSR  WAIT
0528 8C6B 20 7C BC JSR  MTR1
0529 8C6E A2 00    LDX  #00      ; RESET NUMBER OF HEADS
0530 8C70 AD 04 BF LDA  DSTAR    ; GET HEAD DATA
0531 8C73 29 20    AND  ##20      ; BIT 5 FOR # OF HEADS
0532 8C75 F0 01    BEQ  INTST
0533 8C77 EB      INX      ; TWO HEADS
0534 8C78 8E C1 04 INTST STX  NHEAD
0535 8C7B 60      RTS

0537 8C7C 8A      MTR1  TXA
0538 8C7D F0 08    BEQ  MTR3      ; DRIVE IS DOUBLE DENSITY
0539 8C7F A9 80    LDA  ##80      ; YES
0540 8C81 0D B1 04 ORA  MOFLG
0541 8C84 8D B1 04 STA  MOFLG
0542 8C87 AD B2 04 MTR3  LDA  USIDE    ; SIDE
0543 8C8A 0D B1 04 ORA  MOFLG    ; SET SIDE
0544 8C8D 8D B1 04 STA  MOFLG
0545 8C90 AE B4 04 LDX  UDRV      ; DRIVE NBR
0546 8C93 E0 04    CPX  #4        ; VALID DRIVE ?
0547 8C95 30 03    BMI  MTR4      ; YES
0548 8C97 4C FE 88 JMP  ERCOND    ; BAD DRIVE #
0549
0550 8C9A 8D B7 BC MTR4  LDA  DRVTBL, X
0551 8C9D 0D B1 04 ORA  MOFLG
0552 8CA0 8D B0 04 STA  SELFLG    ; DRIVE SELECT FLAG SET
0553 8CA3 20 F0 BC JSR  SETFLG    ; SET FLAGS
0554 8CA6 AD C6 04 WAIT  LDA  HDEL
0555 8CA9 8D A6 04 STA  TEMP2    ; DELAY ROUTINE
0556 8CAC A9 FF    LDA  ##FF
0557 8CAE 20 D9 BE WAITO LDA  MICROD  ; 10US DELAY
0558 8CB1 CE A6 04 DEC  TEMP2
0559 8CB4 D0 F6    BNE  WAITO
0560 8CB6 60      SELCT4 RTS
0561
0562 8CB7 42      ; DRVTBL  BYT  $42, $44, $48, $50
    
```

```

0564 ; ENTER WITH DESIRED DRIVE (0-3) IN ACC AND
0565 ; SET INDEX REG 'Y' TO SIDE.
0566 ; SET INDEX REG 'X' WITH A
0567 ; 1 FOR SINGLE DENSITY
0568 ; 0 FOR DOUBLE DENSITY
0569 ; ONLY ONE DRIVE CAN BE SELECTED AT A TIME

0571 8CBB 68 SELCTD PLA ; ENTRY FROM MOTON

0573 8CBC D8 SELECT CLD ; MAKE SURE DECIMAL MODE CLEAR
0574 8CBD CD B4 04 CMP UDRV ; SAME DRIVE?
0575 8CC0 D0 14 BNE SELCT1 ; NO
0576 8CC2 CC B2 04 CPY USIDE ; SAME SIDE?
0577 8CC5 D0 12 BNE SELCT2 ; NO
0578 8CC7 8A TXA
0579 8CC8 6A ROR A
0580 8CC9 6A ROR A ; PUT DENSITY IN MSB OF A
0581 8CCA 4D B1 04 EOR MOFLG ; POSITIVE IF THE SAME
0582 8CCD 30 0D BMI SELCT3
0583 8CCF 29 20 AND ##20
0584 8CD1 D0 E3 BNE SELCT4 ; AND MOTOR ON!!
0585 8CD3 4C FE 88 JMP ERCOND ; MOTOR OFF

0587 8CD6 8D B4 04 SELCT1 STA UDRV
0588 8CD9 8C B2 04 SELCT2 STY USIDE
0589 8CDC 20 E2 8C SELCT3 JSR DESEL
0590 8CDF 4C 7C 8C JMP MTR1
  
```

```

0592 BCE2 A9 20 DESEL LDA ##20 ; LEAVE MOTOR ON
0593 BCE4 2D B1 04 AND MOFLG
0594 BCE7 20 F0 8C DESEL1 JSR SETFLG ; SET FLAGS
0595 BCEA A9 00 LDA #00
0596 BCEC 8D B0 04 STA SELFLG ; SET SELECT FLG OFF
0597 BCEF 60 RTS

0599 BCF0 8D 04 8F SETFLG STA DCONR ; STATUS FLAG
0600 BCF3 8D B1 04 STA MOFLG ; MOTOR ON FLAG
0601 BCF6 60 RTS

0603 BCF7 A9 00 MOTOFF LDA #00
0604 BCF9 F0 EC BEQ DESEL1 ; ALWAYS GOES
  
```

```

0606          ;ENTER WITH 'RDBUF' POINTING TO MEMORY LOCATION WHERE
0607          ; DATA WILL BE READ

0609 BCFB 20 CA 8D RDTRK JSR SETIT ;MOTOR ON & SETUP PARAMETERS
0610 BCFE F0 0E          BEG RDT1 ;NO
0611 8D00 20 A6 8E          JSR WHTK ;SETUP TK COUNT
0612 8D03 20 0C 8E          JSR RDDMA ;YES-SETUP DMA
0613 8D06 A9 E4          LDA #RTCOD
0614 8D08 20 D3 8B          JSR EXEC03
0615 8D0B 4C 39 8D          JMP RDS3 ;EXECUTE FDC
0616
0617 8D0E 20 E3 8E RDTRK JSR RDBPTR
0618 8D11 A9 E4          LDA #RTCOD ;CMD TO READ A WHOLE TRACK
0619 8D13 8D BB 04          STA FORFLG
0620 8D16 20 D3 8B          JSR EXEC03
0621 8D19 2C 15 8F RDTRK BIT FSTOP ;STOP CPU
0622 8D1C AD 03 8F          LDA FDAR ;YES, GET THE BYTE
0623 8D1F 91 DB          STA (PTR),Y ;STORE IT INTO USERS BUFFER
0624 8D21 C8          INY ;BUMP PTR
0625 8D22 D0 F5          BNE RDTKO ;PTR NOT AT END
0626 8D24 E6 DC          INC PTR+1 ;IS L/O PART OF PTR AT END
0627 8D26 4C 19 8D          JMP RDTKO
  
```

```

0629          ;ENTER WITH 'RDBUF' POINTING TO MEMORY LOC WHERE DATA
0630          ; WILL BE PUT. SET ACC 'A' WITH THE SECTOR NUMBER.
0631          ;

0633 8D29 20 C7 8D RDSEK JSR SETSEC ;SECTOR SETUP
0634 8D2C F0 16          BEG RDS1 ;NO DMA
0635 8D2E 20 BB 8E          JSR SECCNT ;SETUP DMA COUNT
0636 8D31 20 0C 8E          JSR RDDMA ;YES-SETUP DMA
0637 8D34 A9 80          LDA #RSCOD
0638 8D36 20 C2 8B RDSEK JSR EXEC02
0639 8D39 20 77 89 RDSEK JSR RESTED ;WAIT FOR FDC INTERRUPT
0640 8D3C 20 7A 8E RDSEK JSR QDMA ;DMA ACTIVE, Z=0 IF STILL
0641 8D3F D0 FB          BNE RDS4
0642 8D41 4C 7E 89          JMP LSTFLG ;LOAD WITH STFLG AND RETURN

0644 8D44 20 E3 8E RDSEK JSR RDBPTR ;SET UP PTR TO READ BUFFER
0645 8D47 A9 80          LDA #RSCOD ;CMD TO READ A SECTOR
0646 8D49 20 C2 8B RDSEK JSR EXEC02 ;EXECUTE THE CMD
0647 8D4C 20 52 8D RDSEK JSR INBYTE
0648 8D4F 4C 77 89          JMP RESTED

0650 8D52 2C 15 8F INBYTE BIT FSTOP ;STOP CPU
0651 8D55 AD 03 8F          LDA FDAR ;GET DATA FROM DATA REG
0652 8D58 91 DB          STA (PTR),Y ;STUFF IT
0653 8D5A C8          INY
0654 8D5B CC C0 04          CPY NBYTE ;GOT ALL ?
0655 8D5E D0 F2          BNE INBYTE ;NO
0656 8D60 60          RTS
  
```

```

0658 ; ENTER WITH 'WRBUF POINTING TO MEMORY WHERE DATA
0659 ; WILL BE READ. SET ACC 'A' WITH THE SECTOR NUMBER.
0660 ;
;
0662 8D61 20 C7 8D WRTSEC JSR SETSEC ; SECTOR SETUP
0663 8D64 F0 0A BEQ WRT1 ; NO DMA
0664 8D66 20 BB 8E JSR SECCNT ; SET DMA COUNT
0665 8D69 20 59 8E JSR WRTDMA ; YES-SETUP DMA
0666 8D6C A9 A0 LDA #WSCOD
0667 8D6E D0 C6 BNE RDS2 ; EXECUTE FDC
0668 ; ***
0669 ;
0670 8D70 20 F1 8E WRT1 JSR WRTPTR ; SET UP WRT BUFR IN PTR
0671 8D73 A9 A0 LDA #WSCOD ; CMD TO WRITE A SECTOR
0672 8D75 20 C2 8B JSR EXEC02 ; EXEC THE CMD
0673 8D78 20 7E 8D JSR OUTBYT
0674 8D7B 4C 77 89 JMP RESTED
;
0676 8D7E B1 D8 OUTBYT LDA (PTR),Y ; GET BYTE THAT IS TO BE WRITTE
0677 8D80 2C 15 8F BIT FSTOP ; STOP CPU
0678 8D83 8D 03 8F STA FDAR
0679 8D86 C8 INY ; GO TO NEXT BYTE
0680 8D87 CC C0 04 CPY NBYTE ; HAVE ALL BYTES BEEN READ?
0681 8D8A D0 F2 BNE OUTBYT ; NO, GET MORE DATA
0682 8D8C 60 RTS

```

```

0684 ; ENTER WITH 'RDBUF' POINTING TO MEMORY LOCATION
0685 ; WHERE DATA WILL BE READ. ACC WILL CONTAIN THE
0686 ; SECTOR NUMBER AND INDEX REG 'X' WILL CONTAIN
0687 ; THE LAST SECTOR TO BE READ.
0688 ;
;
0690 8D8D 20 C1 8D RDMSC JSR SETMSC ; SETUP
0691 8D90 F0 12 BEQ RDM1 ; NO DMA
0692 8D92 20 BB 8E JSR BYTES ; SETUP BYTE COUNT
0693 8D95 20 0C 8E JSR RDDMA ; YES-SETUP DMA
0694 8D98 A9 90 LDA #RSCOD+M
0695 8D9A 20 C2 8B RDMDMA JSR EXEC02
0696 8D9D 20 7A 8E RRDMA1 JSR QDMA
0697 8DA0 D0 FB BNE RRDMA1
0698 8DA2 F0 15 BEQ FORCOF ; ALWAYS GOES
;
0700 8DA4 20 E3 8E RDM1 JSR RDBPTR ; SET UP PTR TO READ BUFR
0701 8DA7 A9 90 LDA #RSCOD+M ; CMD TO READ MULTIPLE SECT
0702 8DA9 20 C2 8B JSR EXEC02
0703 8DAC 20 52 8D RDM01 JSR INBYTE ; XFER DATA IN
0704 8DAF 20 D8 8D JSR TONEXT ; BUMP PTR
0705 8DB2 B0 F8 BCS RDM01 ; NO
0706 8DB4 A9 0D RDM05 LDA #13
0707 8DB6 20 D9 8E JSR MICROD
0708 8DB9 A9 D0 FORCOF LDA #FICOD ; STOP EXECUTING
0709 8DBB 20 D6 8B JSR EXEC04
0710 8DBE 4C 16 8C JMP FORM5
;
0712 8DC1 8E AC 04 SETMSC STX LSECR ; LAST SECTOR TO BE READ
0713 8DC4 8D AB 04 STA CURSEC ; CURR SECTOR
0714 8DC7 8D 02 8F SETSEC STA FSECR ; SECTOR # DESIRED
0715 8DCA 20 C4 88 SETIT JSR SETUP ; CK FLAGS & SETUP PARAMETERS
0716 8DCD AD AD 04 LDA STFLG ; ANY ERRORS
0717 8DD0 10 05 BPL ALLOK
0718 8DD2 68 PLA ; REMOVE RETURN ADDRESS
0719 8DD3 68 PLA
0720 8DD4 4C 7E 89 JMP LSTFLG
;
0722 8DD7 AD BA 04 ALLOK LDA DMAFLG ; DMA ?
0723 8DDA 60 RTS
;
0725 8DDB 20 CA 8E TONEXT JSR BUMPTR ; BUMP PTR
0726 8DDE A0 00 LDY #0
0727 8DE0 EE AB 04 INC CURSEC ; GOTO NEXT SECT
0728 8DE3 AD AC 04 LDA LSECR
0729 8DE6 CD AB 04 CMP CURSEC ; ALL SECT XFER'ED ?
0730 8DE9 60 RTS

```

```

0732 ; ENTER WITH 'WRBUF' POINTING TO MEMORY WHERE DATA
0733 ; WILL BE READ. ACC WILL CONTAIN THE SECTOR NBR
0734 ; AND INDEX REG 'X' WILL CONTAIN LAST SECTOR TO BE
0735 ; READ
0736 ;
;
0738 8DEA 20 C1 8D WRTMSC JSR SETMSC ; SETUP
0739 8DED F0 0B BEQ WRTM1 ; NO DMA
0740 8DEF 20 8B 8E JSR BYTES ; SET BYTE COUNT
0741 8DF2 20 59 8E JSR WRTDMA ; YES-SETUP DMA
0742 8DF5 A9 80 LDA #WSCOD+M
0743 8DF7 4C 9A 8D JMP RDMDMA ; EXECUTE FDC
0744 ;
0745 8DFA 20 F1 8E WRTM1 JSR WRTPTR ; SET UP WRT BUFR IN PTR
0746 8DFD A9 80 LDA #WSCOD+M ; CMD TO WRITE MULTIPLE SECT
0747 8DFF 20 C2 8B JSR EXEC02 ; EXECUTE THE CMD
0748 8E02 20 7E 8D WRMSO JSR OUTBYT ; XFER DATA OUT
0749 8E05 20 DB 8D JSR TONEXT ; BUMP PTR
0750 8E08 B0 FB BCS WRMSO ; NO
0751 8E0A 90 AB BCC RDM05
0752 ; ***

```

```

0754 ; NOTE!!! A) CANT TRANSFER FROM AIM65/40 MEM TO DISK U
0755 ; DMA, NEED EXTERNAL MEM.....
0756 ; B) CPU MUST BE STOPPED BY DMA IF FIRMWARE IS
0757 ; EXTERNAL RAM/ROM OR ELSE CPU WILL READ BA
0758 ; DATA AND GET LOST.....
;
0760 0E0C 20 01 0E RDDMA JSR SETDMA ; SET DMAADR->DMAPTR
0761 0E0E 20 03 0E JSR RDBPTR
0762 0E12 A9 0E LDA #>FDAR ; SOURCE=FDAR DATA REG
0763 0E14 A0 01 LDY #1
0764 0E16 91 DD STA (DMAPTR),Y ; DMASR1
0765 0E18 A9 03 LDA #<FDAR
0766 0E1A C0 INY
0767 0E1B 91 DD STA (DMAPTR),Y ; DMASR2
0768 0E1D A0 04 LDY #4 ; A=3
0769 0E1F 91 DD STA (DMAPTR),Y ; DMADST
0770 0E21 A9 3A LDA #3A ; READ DMA CMD, INH DMA INT, STO
0771 0E23 48 PHA
0772 0E24 AE CA 04 LDX RDBUF+1 ; DEST=RDBUF
0773 0E27 AD C9 04 LDA RDBUF
0774 0E2A 48 CDMA PHA
0775 0E2B 8A TXA
0776 0E2C A0 05 LDY #5
0777 0E2E 91 DD STA (DMAPTR),Y ; DMADS1-MS DEST ADDR
0778 0E30 C8 INY
0779 0E31 68 PLA
0780 0E32 91 DD STA (DMAPTR),Y ; DMADS2-LS DEST ADDR
0781 0E34 AD AB 04 LDA LSC ; LS BYTE COUNT
0782 0E37 A0 0A LDY #0A
0783 0E39 91 DD STA (DMAPTR),Y ; DMACT2
0784 0E3B AD A7 04 LDA MSC ; MS BYTE COUNT
0785 0E3E 88 DEY
0786 0E3F 91 DD STA (DMAPTR),Y ; DMACT1
0787 0E41 A9 04 LDA #04 ; DECR DMA BYTE CNT REG
0788 0E43 88 DEY
0789 0E44 91 DD STA (DMAPTR),Y ; DMACT
0790 0E46 A0 0C LDY #0C
0791 0E48 B1 DD LDA (DMAPTR),Y ; DMACMD-CL DMA STATUS/IRG
0792 0E4A AD BA 04 LDA DMAFLG ; GET BANK INFO
0793 0E4D 29 C0 AND #0C ; ISOLATE IT
0794 0E4F 8D A6 04 STA TEMP2
0795 0E52 68 PLA
0796 0E53 0D A6 04 ORA TEMP2
0797 0E56 91 DD STA (DMAPTR),Y ; DMACMD-ACTIVATE DMA
0798 0E58 60 RTS

```

```

0800 8E59 20 B1 8E   WRTDMA JSR   SETDMA   ;DMAADR->DMAPTR
0801 8E5C 20 F1 8E   JSR     WRTPTR
0802 8E5F A9 03           LDA     #03      ;INCR SOURCE ADDR REG
0803 8E61 A0 00           LDY     #0
0804 8E63 91 DD           STA     (DMAPTR),Y ;DMASRC
0805 8E65 AD CC 04       LDA     WRBUF+1 ;SOURCE=WRBUF
0806 8E68 C8             INY
0807 8E69 91 DD           STA     (DMAPTR),Y ;DMASR1
0808 8E6B AD CC 04       LDA     WRBUF
0809 8E6E C8             INY
0810 8E6F 91 DD           STA     (DMAPTR),Y ;DMASR2
0811 8E71 A9 39           LDA     #*39    ;DMA WRITE CMD
0812 8E73 48             PHA
0813 8E74 A2 8F           LDX     #>FDAR  ;DEST=FDC DATA REG
0814 8E76 A9 03           LDA     #<FDAR
0815 8E78 D0 B0           BNE     CDMA    ;DO IT
0816                    ;      ***

0818 8E7A A0 0C   QDMA  LDY     #<DMACMD
0819 8E7C A9 40   LDA     #*40
0820 8E7E 31 DD   AND     (DMAPTR),Y
0821 8E80 60     RTS

0823 8E81 AD B9 04   SETDMA LDA     DMAADR
0824 8E84 85 DE   STA     DMAPTR+1
0825 8E86 A9 00   LDA     #0
0826 8E88 85 DD   STA     DMAPTR
0827 8E8A 60     RTS
    
```

```

0829                    ; CALCULATE # OF BYTES FROM CURSEC TO LSECR
0830                    ;
0831 8E8B AC AB 04   BYTES LDY     CURSEC ; CURRENT SECTOR
0832 8E8E A2 00   LDX     #0          ; MS
0833 8E90 8A       TXA          ; LS
0834 8E91 38       BYTE1 SEC
0835 8E92 CE C0 04   DEC     NBYTE      ; MAKE 0=FF
0836 8E95 6D C0 04   ADC     NBYTE      ; BYTES/SECTOR
0837 8E98 90 01   BCC     BYTE4     ; ?
0838 8E9A EB       INX          ; MS
0839 8E9B EE C0 04   BYTE4 INC     NBYTE ; RESTORE OLD VALUE
0840 8E9E CC AC 04   CPY     LSECR     ; DONE ?
0841 8EA1 F0 20   BEQ     SEC1     ; YES
0842 8EA3 CB       INY
0843 8EA4 D0 EB   BNE     BYTE1    ; CONTINUE
0844                    ;      ***
0845                    ;
0846                    ; SET DMA CNT TO WHOLE TK
0847                    ;
0848 8EA6 AD 04 8F   WHTK  LDA     DSTAR  ; DEN/SIZE DATA
0849 8EA9 2A       ROL     A
0850 8EAA 2A       ROL     A          ; PUT BITS 6,7 ->0,1
0851 8EAB 2A       ROL     A
0852 8EAC 29 03   AND     #3
0853 8EAE AA       TAX
0854 8EAF BD B7 8E   LDA     TKTBL,X ; MS
0855 8EB2 AA       TAX
0856 8EB3 A9 00   LDA     #00
0857 8EB5 F0 0C   BEQ     SEC1
0858                    ;      ***
0859                    ; TK COUNTS EXTEND INTO TRAILER BUT S
0860                    ; STOP SHORT OF FINAL INDEX TO ALLOW
0861                    ; DMA TO COMPLETE.....
0862                    ;
0863 8EB7 14       TKTBL BYT  $14,$0C,$27,$18 ; S8,S5,DB,D5
0864                    ;
0865                    ; SET SECTOR BYTE COUNT
0866                    ;
0867 8EBB A2 00   SECCNT LDX     #00
0868 8EBD AD C0 04   LDA     NBYTE   ; BYTE CNT
0869 8EC0 D0 01   BNE     SEC1   ; SINGLE ?
0870 8EC2 EB       INX
0871 8EC3 8E A7 04   SEC1  STX     MSC
0872 8EC6 8D AB 04   STA     LSC
0873 8EC9 60     RTS
0874                    ; INCREMENT DATA TRANSFER POINTER
0875                    ;
0876 8ECA 98       BUMPTR TYA          ; SECTORS
0877 8ECB D0 02   BNE     BUMPER
0878 8ECD E6 DC   INC     PTR+1 ; IF SIZE IS 256
0879 8ECF 18       BUMPER CLC
0880 8ED0 65 DB   ADC     PTR
0881 8ED2 85 DB   STA     PTR
0882 8ED4 90 02   BCC     BUMPO1
0883 8ED6 E6 DC   INC     PTR+1
    
```

```
0884 BED8 60          BUMP01 RTS

0886                ; DELAY OF 10 USEC FOR EACH CNT OF 'A'
0887                ;
0888 BED9 8D A3 04    MICROD STA  CNTR
0889 BEDC EA          MICR1  NOP      ; 2 USEC
0890 BEDD CE A3 04    DEC      CNTR    ; 5 USEC
0891 BEE0 D0 FA      BNE      MICR1    ; 3 USEC
0892 BEE2 60          RTS

0894                ; SET RDBUF VARIABLES INTO PTR VARIABLES.

0896 BEE3 AD C9 04    RDBPTR LDA  RDBUF  ; SET UP READ BUFR IN PTR
0897 BEE6 85 DB      STA  PTR
0898 BEE8 AD CA 04    LDA  RDBUF+1 ; H/O
0899 BEEB 85 DC      STA  PTR+1
0900 BEED 4E AF 04    LSR  WRTFLG  ; CLEAR WRITE FLAG
0901 BEF0 60          RTS

0903 BEF1 AD CB 04    WRTPTR LDA  WRBUF  ; SET UP WRT BUFR IN PTR
0904 BEF4 85 DB      STA  PTR
0905 BEF6 AD CC 04    LDA  WRBUF+1
0906 BEF9 85 DC      STA  PTR+1
0907 BEFB 38          SEC
0908 BEFC 6E AF 04    RDR  WRTFLG  ; SET WRT FLAG
0909 BEFF 60          RTS
```

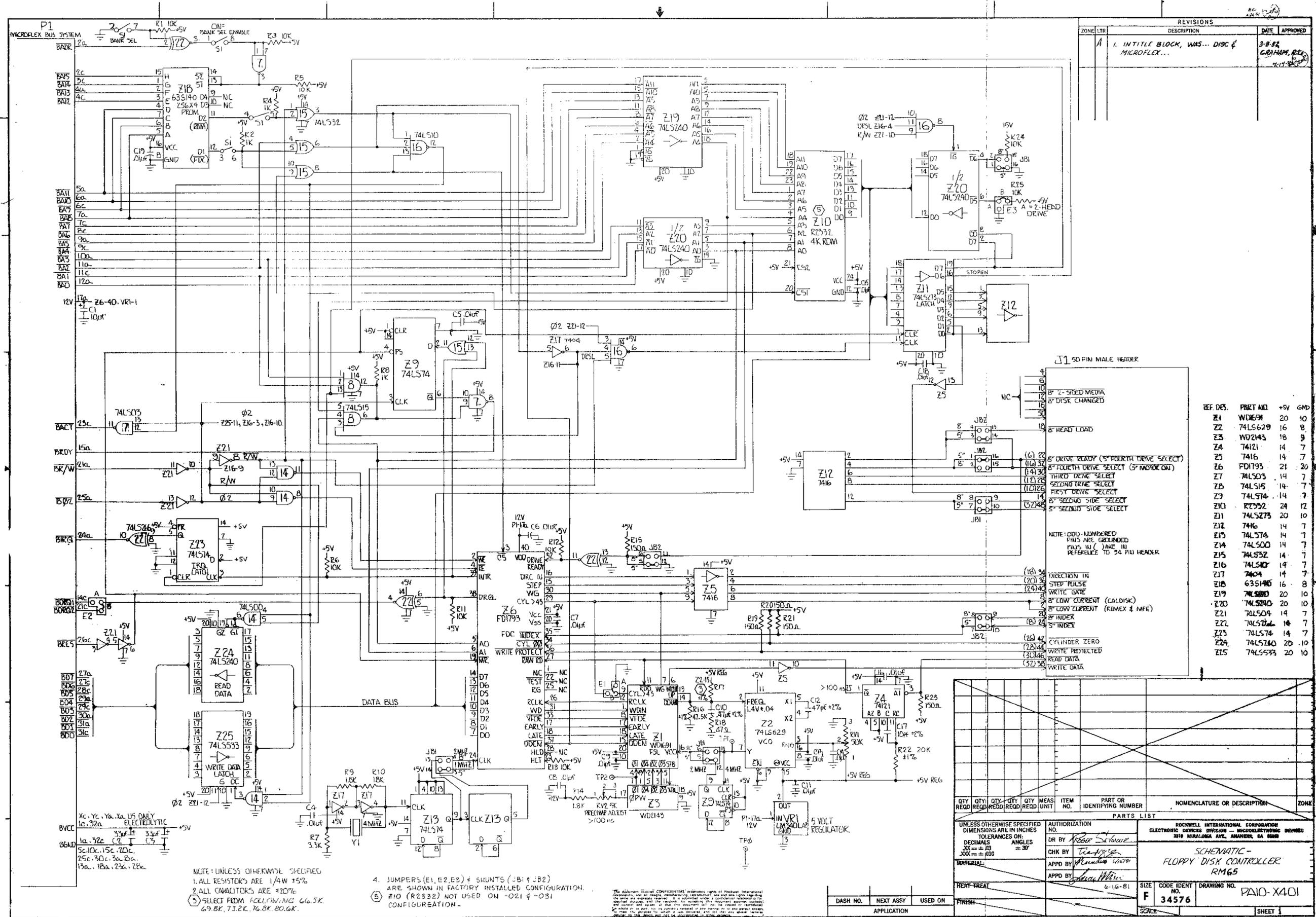
```
0911 BF00 43          .BYT 'COPYRIGHT 1982 '
0912 BF0F 52          .BYT 'ROCKWELL '
0913 BF18 49          .BYT 'INTERNATIONAL '
0914                  .END          FDC

ERRORS = 0000 <0000>
```



RDBUF	04C9	#0113	0772	0773	0896	0898			
RDDMA	8EOC	0612	0636	0693	#0760				
RDMDMA	8D9A	#0695	0743						
RDMSC	8D8D	#0690							
RDM01	8DAC	#0703	0705						
RDM05	8DB4	#0706	0751						
RDM1	8DA4	0691	#0700						
RDSEC	8D29	#0633							
RDS1	8D44	0634	#0644						
RDS2	8D36	#0638	0667						
RDS3	8D39	0615	#0639						
RDS4	8D3C	#0640	0641						
RDTK0	8D19	#0621	0625	0627					
RDTRK	8CFB	#0609							
RDT1	8DOE	0610	#0617						
RECOD	0000	#0034							
REST	895B	0215	#0219						
RESTED	8977	#0230	0232	0639	0648	0674			
RESTEE	8971	0222	0225	#0227					
RETRY	04CE	#0116							
RRDMA1	8D9D	#0696	0697						
RSCOD	0080	#0037	0637	0645	0694	0701			
RSTOR	8956	0213	#0217						
RTCOD	00E4	#0041	0613	0618					
S	0008	#0050	0425						
SECCNT	8EBB	0635	0664	#0867					
SECTK5	04C4	#0108	0163						
SECTK8	04C5	#0109	0170						
SEC1	8EC3	0841	0857	0869	#0871				
SEEK	8938	#0205							
SEEKNV	8982	0187	#0238						
SELCTD	8C8B	0521	#0571						
SELECT1	8CD6	0575	#0587						
SELECT2	8CD9	0577	#0588						
SELECT3	8CDC	0582	#0589						
SELECT4	8CB6	#0560	0584						
SELECT	8CBC	#0573							
SELFLG	04B0	#0090	0157	0552	0596				
SETDMA	8EB1	0760	0800	#0823					
SETFLG	8CF0	0526	0553	0594	#0599				
SETIRG	8904	0160	#0179						
SETIT	8DCA	0184	0206	0312	0609	#0715			
SETMSC	8DC1	0690	#0712	0738					
SETSEC	8DC7	0633	0662	#0714					
SETUP	88C4	0133	#0150	0715					
SFLAG	04C7	#0111	0426						
SKCOD	0010	#0035	0214						
SNGLCT	04BF	#0103							
STFLG	04AD	#0087	0154	0176	0233	0498	0507	0716	
STUP1	88EC	0165	#0167						
SW2	MACR								
SW3	MACR								
TABLE5	04CF	#0118	0319						
TABLE8	04DF	#0120	0313	0316					
TEMP	04A4	#0079							

TEMP2	04A6	#0080	0555	0558	0794	0796			
TKTBL	8EB7	0854	#0863						
TONEXT	8DDB	0704	#0725	0749					
TUNDEL	04CB	#0112	0224						
T5	8B9A	#0145							
T8	8BAA	#0146							
UCYL	04B3	#0093	0186	0196	0197	0205	0210	0352	0391
UDRV	04B4	#0094	0207	0523	0545	0574	0587		
USIDE	04B2	#0092	0353	0392	0524	0542	0576	0588	
V	0004	#0048							
WA	MACR								
WAIT	BCA6	0527	#0554						
WAITS1	8BE2	#0442	0443						
WAITS2	8BE9	#0446	0447						
WAITO	8CAC	#0556	0559						
WHTK	8EA6	0611	#0848						
WRBUF	04CB	#0114	0805	0808	0903	0905			
WRMS0	8E02	#0748	0750						
WRTDMA	8E59	0665	0741	#0800					
WRTFLG	04AF	#0089	0221	0223	0323	0900	0908		
WRTMSC	8DEA	#0738							
WRTM1	8DFA	0739	#0745						
WRTPTR	8EF1	0670	0745	0801	#0903				
WRTRK	8994	0190	#0312						
WRTRK1	8997	#0313							
WRTRK3	89AC	#0324							
WRTSEC	8D61	#0662							
WRT1	8D70	0663	#0670						
WSCOD	00A0	#0038	0666	0671	0742	0746			
WSFM	8B35	#0389	0404						
WSFMO	8BBF	0403	#0406						
WSMFM	8A1B	#0344	0372						
WTCOD	00F4	#0042	0324						
WTDON	8AFB	0375	#0377	0379					
WTDOND	8AF9	0364	#0376						
WTDONS	8AF5	#0374	0406						
WTFM	8B04	0331	#0382						
WTMFM	89C1	0330	#0334						
WT5FM	8B28	0382	#0387						
WT5MFM	8A04	0334	#0340						
WY	MACR								
W1	MACR								
W2	MACR								
W3	MACR								
NARG	****								



REVISIONS			
ZONE	LTR	DESCRIPTION	DATE APPROVED
A		1. INTITILE BLOCK, WAS... DISC & MICROFLEX...	3-8-81 GRAHAM, R.E. 11-1-81

REF. DES.	PART NO.	+5V	GND
Z1	WD659	20	10
Z2	74LS629	16	8
Z3	WD2143	18	9
Z4	74121	14	7
Z5	7416	14	7
Z6	FD1793	21	20
Z7	74LS03	14	7
Z8	74LS15	14	7
Z9	74LS74	14	7
Z10	R7952	24	12
Z11	74LS273	20	10
Z12	7416	14	7
Z13	74LS74	14	7
Z14	74LS00	14	7
Z15	74LS52	14	7
Z16	74LS07	14	7
Z17	7414	14	7
Z18	635140	16	8
Z19	74LS00	20	10
Z20	74LS00	20	10
Z21	74LS04	14	7
Z22	74LS04	14	7
Z23	74LS74	14	7
Z24	74LS240	20	10
Z25	74LS532	20	10

PARTS LIST			
QTY	QTY	QTY	QTY
READ	READ	READ	READ
ITEM NO.	PART OR IDENTIFYING NUMBER	NOMENCLATURE OR DESCRIPTION	ZONE
UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES			
TOLERANCES ON: DECIMALS ANGLES			
XX ± .03 30X ± .010			
MATERIAL:			
NEXT TREAT:			
FINISH:			
DASH NO. NEXT ASSY USED ON APPLICATION			
AUTHORIZATION NO. DR BY: Robert Schmale			
CHK BY: [Signature]			
APPD BY: [Signature]			
APPD BY: [Signature]			
DATE: 6-16-81			
SIZE: F		CODE IDENT: 34576	DRAWING NO. PAIO-X401
SCALE: SHEET 1			

Xc, Yc, Ya, Xa, Us ONLY  
1c, 32a ELECTROLYTIC  
33uf 33uf  
1a, 32c 2 1 3  
5c, 10c, 15c, 20c  
25c, 30c, 35c, 40c  
15a, 18a, 22a, 25a

NOTE: UNLESS OTHERWISE SPECIFIED  
1. ALL RESISTORS ARE 1/4W ±5%  
2. ALL CAPACITORS ARE ±20%  
3. SELECT FROM FOLLOWING: 66.5K, 69.8K, 73.2K, 76.8K, 80.6K

4. JUMPERS (E1, E2, E3) & SHUNTS (JBI & JB2) ARE SHOWN IN FACTORY INSTALLED CONFIGURATION.  
E10 (R2332) NOT USED ON -021 & -031 CONFIGURATION.

THIS DOCUMENT IS UNCLASSIFIED EXCEPT WHERE SHOWN OTHERWISE. IT IS SUBJECT TO A CONFIDENTIALITY AGREEMENT WITH THE U.S. GOVERNMENT. IT IS HEREBY RELEASED UNDER A CONFIDENTIALITY AGREEMENT WITH THE U.S. GOVERNMENT. IT IS HEREBY RELEASED UNDER A CONFIDENTIALITY AGREEMENT WITH THE U.S. GOVERNMENT.

PAIO-X401