USER GUIDE

DSD 880x/8

DATA SYSTEMS DESIGN

DSD 880x/8

WINCHESTER/FLOPPY DISK

STORAGE SYSTEM

USER GUIDE

Data Systems Design, Inc.
2241 Lundy Avenue
San Jose, CA 95131

Data Systems Part No: 040018-01
Revision B

Printed in USA: 2/83

WARRANTY STATEMENT

Data Systems Design's products are warranted against defects in materials and workmanship. For DSD products sold in the U.S.A., this warranty applies for ninety (90) days from date of shipment. *DSD will, at its option, repair or replace either equipment or components which prove to be defective during the warranty period. This warranty includes labor, parts, and surface travel costs of system modules or components. Freight charges for other than surface travel or for complete systems returned for repair are not included in this warranty. Equipment returned to DSD for repair must be shipped freight prepaid and accompanied by a Material Return Authorization number issued by DSD Customer Service. Repairs necessitated by shipping damage, misuse of the equipment, or by hardware, software, or interfacing not provided by DSD are not covered by this warranty.

NO OTHER WARRANTY IS EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, DSD SHALL NOT BE LIABLE FOR CONSEQUENTIAL DAMAGES.

ORDERS SHOULD BE PLACED THROUGH THE NEAREST REGIONAL SALES OFFICE.

WESTERN REGION SALES
Data Systems Design, Inc.
2560 Mission College Blvd.
Suite 108
Santa Clara, CA 95051
(408) 727-3163

CENTRAL REGION SALES
Data Systems Design, Inc.
5050 Quorum Drive
Suite 339
Dallas, TX 75240
(214) 980-4884

EASTERN REGION SALES
Data Systems Design, Inc.
51 Morgan Drive
Norwood, MA 02062
(617) 769-7620
TWX: 710-336-0120

CORPORATE HEADQUARTERS
Data Systems Design, Inc.
2241 Lundy Avenue
San Jose, CA 95131

* For products sold outside the U.S.A., contact your local DSD distributor for warranty terms.

PREFACE

This manual describes the features, specifications, and register usage of the DSD 880 Data Storage System.

Instructions for DSD 880 installation, operation, and elementary troubleshooting are included in this manual.

The material in this manual is subject to change without notice. The manufacturer assumes no responsibility for any errors which may appear in this manual.

Please note that DEC, LSI-11, LSI-11/2, LSI-11/23, PDP-11, RSX-11, RX02, RL01, and RL02 are registered trademarks of the Digital Equipment Corporation.

SAFETY

Operating and maintenance personnel must at all times observe sound safety practices. Do not replace components, or attempt repairs to this equipment with the power turned on. Under certain conditions, dangerous voltage potentials may exist when the power switch is in the off position, due to charges retained by capacitors. To avoid injury, always remove power cord before attempting repair procedures.

Data System Design, Inc. will accept no responsibility or liability for injury or damage sustained as a result of operation or maintenance of this equipment with the covers removed and power applied.

WARNING

This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions manual, may cause interference to radio communications. As temporarily permitted by regulations, it has not been tested for compliance with the limits for Class A computing devices pursuant to the sub-part J or Part 15 of the FCC rules which are designed to provide reasonable protection against such interference. The operation of this equipment in a residential area is likely to cause interference. The user, at his own expense, will be required to take whatever measures may be required to correct the interference.

CAUTION

Do not operate system until you have:

- Released the lock on the winchester drive (spindle lock).
- Rotated the head lock actuator to RUN position.

Both locks are secured in a locked position prior to shipment from the factory. See Section 3 for detailed procedures covering installation and checkout of equipment.

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont)

TABLE OF CONTENTS (Cont)

APPENDICES

TABLES

TABLE OF CONTENTS (Cont)

PAGE

TABLES

FIGURES

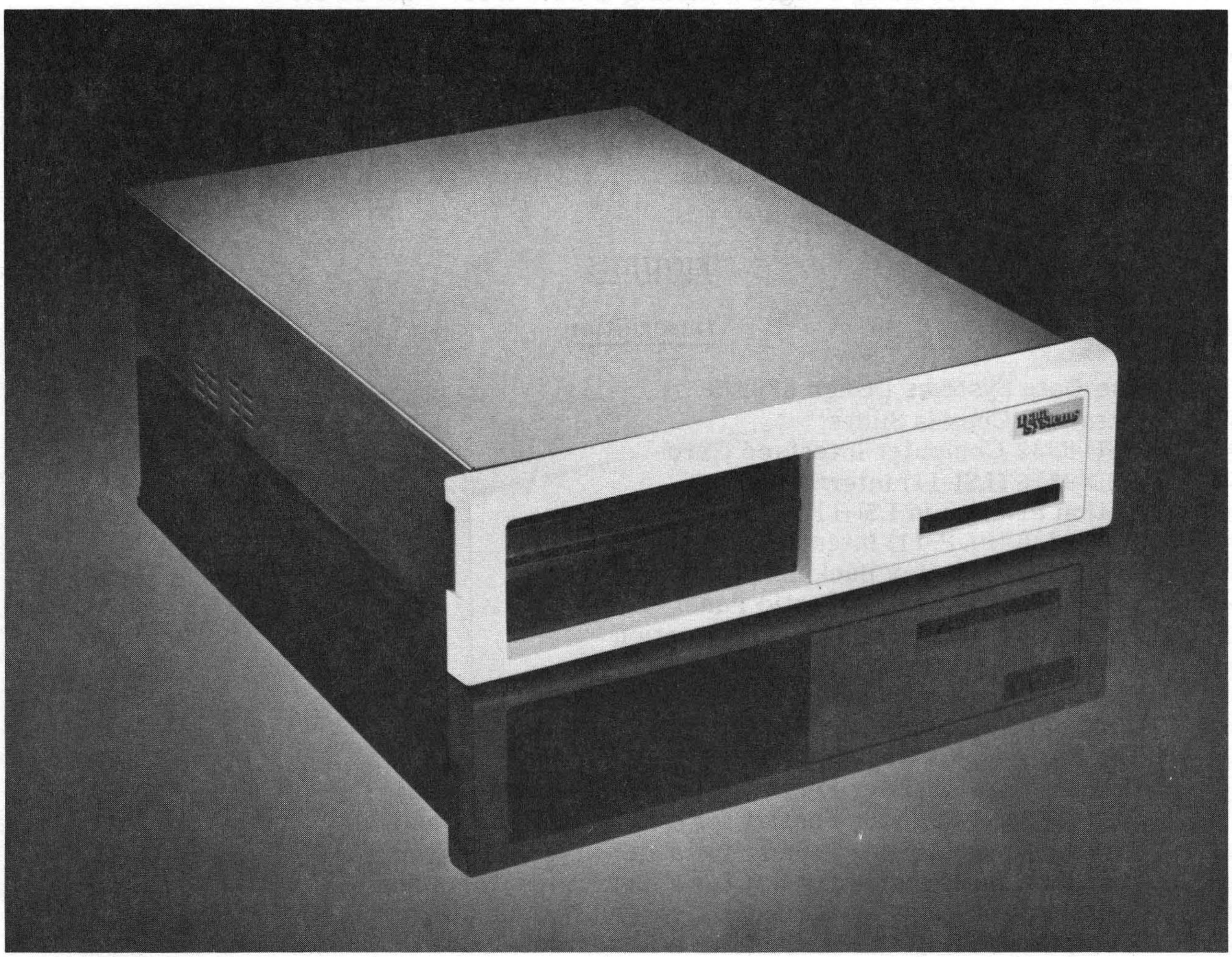


Figure 1-1. Data Systems Design 880x/8

1.0 INTRODUCTION

1.1 General Information

This manual provides user information for the DSD 880x/8 data storage system. Coverage provided includes: features, specifications, installation, operation, elementary programming and user level troubleshooting.

1.2 System Overview

The DSD 880 is a compact data storage system combining the advantages of the winchester disk system and the floppy disk system. Designed for use with computers manufactured by Digital Equipment Corporation (DEC), the DSD 880 provides the large capacity, rapid data access, and reliability of winchester disk technology and the low cost versatility of the floppy disk in a compact, system oriented package.

1.3 Features

1.3.1 System Architecture

The DSD 880 uses a unique system architecture to achieve the economy and performance available by the combination of winchester and floppy disk technologies. The winchester is configured to be compatible with a high performance disk system (the DEC RL01/RL02) while the double sided floppy disk emulates a floppy disk system (the DEC RX02). The DSD 880 is fully hardware, software and interface compatible with DEC computers. The system provides 8.8 Mbytes of on-line storage (7.8 Mbytes fixed and 1 Mbyte removable).

The DSD 880 is implemented with a controller/formatter that is common in both drives. A single computer interface simplifies system integration. A bit-slice processor on this interface arbitrates device requests and queues pending instructions. Each disk drive responds to a different device address, interrupt priority and interrupt vector.

The DSD 880 controller uses a bit-slice processor which switches roles between the winchester and floppy disk drives. A single phase-lock-loop data separator operates at two clock frequencies to accommodate the different data rates of the two drives.

Although the controller can emulate two devices, it cannot do so simultaneously. The computer interface arbitrates RL01 and RX02 command transfers between the controller and the CPU bus. In addition to command arbitration, the interface also performs the following functions:

1. Emulation of RL01 and RX02 command and status registers.

2. Control of data transfers between the CPU and disk controller—including Direct Memory Access (DMA) transfers.

3. Contains the DSD bootstrap load program.

1.3.2 HyperDiagnostics

With the development and introduction of highly sophisticated computer peripherals comes the need to consider new methods of testing and servicing this equipment. DSD has pursued the philosopohy of designing extensive self-testing and diagnostic capabilities into its products. Since our disk memory systems are controlled by microcomputers, self diagnostic features become a natural extension of the product design. DSD's unique HyperDiagnostics provide the operator or service person with a library of user-selectable diagnostic routines and displays indicating system or error information. These HyperDiagnostics permit system diagnosis, floppy disk formatting, winchester backup and floppy drive alignment in a stand alone configuration without tying up a company's expensive computer or test equipment resources. Subsystem faults are easily isolated to allow for quick servicing. The DSD 880 HyperDiagnostics are initiated from a display panel located behind the removable front bezel. The panel is easily accessed by qualified personnel, but is concealed in normal operation.

1.3.3 Off-Line Backup Capability

The use of a common disk controller not only achieves a more economical design, it allows additional interaction between the two disk drives. The DSD 880 controller provides stand alone winchester backup and loading, independent of the CPU. This assures that data will not be lost or destroyed in the event of a computer system failure. Backup and loading are initiated from a unique HyperDiagnostic panel built into the system. The entire winchester contents may be dumped onto floppy disks. When a floppy disk is full, the system pauses and instructs the operator to insert the next one. Reloading is simple and automatic. Each flexible disk is coded with the corresponding winchester track addresses so that it may be inserted in any order, without record keeping. The floppy disks may be single- or double-sided, and single- or double-density.

1.3.4 Reliability

Winchester technology offers the potential for much greater reliability than flexible disk drives. Since the overall system reliability will be limited to that of its weakest component, new innovations are called for to enhance system reliability.

The DSD 880 reliability is increased by automatically shutting off power to the floppy disk drive when it is not in use. This will save wear on media, bearing, belts and pulleys. Since the floppy disk will be used primarily for winchester backup and loading, the mean time between failures (MTBF) of the floppy disk drive, and hence of the overall system, will be significantly increased.

1.4 Summary

Disk memory systems combining winchesters and floppy disks are opening new application possibilities for small computer systems. Their functionality and performance rival that of large disk systems costing several times as much. When considering a winchester-based disk memory system, the user should look beyond the usual considerations of capacity and backup, and should examine the functionality and capability of the entire system.

Data Systems Design has been an industry leader in the design and manufacture of DEC-compatible disk system since 1975. The DSD 880 is a unique, hybrid design which offers a combination of price, features, and performance unavailable from any DEC product. Some of these features are summarized below:

- Cost effective data storage and retrieval
- Large capacity data storage
- Rapid data access
- Simplified system integration
- RL01, RL02, and RX02 emulation
- Off-line backup capability
- Exclusive DSD HyperDiagnostics
- Compact size

2.0 SPECIFICATIONS

2.1 General Information

This section provides specifications and operational requirements for the Data Systems Design 880x/8 Data Storage System.

Specifications include data storage capacities, recording characteristics, and data transfer rates. Also provided is a listing of the major components that comprise the DSD 880x/8 system. Physical dimensions are provided.

Requirements include those for interface cabling and connectors, and power requirements. Operating temperature range and other environmental considerations are given.

2.2 DSD 880x/8 Major Components

Table 2-1 provides a listing of the major components that comprise the DSD 880x/8 Data Storage System.

Table 2-1. DSD 880x/8 Major Components

Component	Part Number
Main Chassis	700006-01
Winchester Disk Drive	SA1004
Flexible Disk Drive	SA850/800
Controller/Formatter Card (8840)	808840-01
or	
Controller/Formatter Card (8841)	808841-02
PDP 11 Interface Card (8830)	808830-01
LSI-11 Interface Card (8832)	808832-01
or	
LSI-11 Interface Card (8836)	808836-01
Diagnostic Panel (8833)	808833-01
Power Supply Assembly 115 Volt	900230-01
Power Supply Assembly 230 Volt	900230-02

2.3 Recording Characteristics

The winchester drive furnished with the DSD 880x/8 data storage system records data using the modified frequency modulation technique (MFM).

The floppy disk system of the DSD 880x/8 is capable of recording data in single-density using the industry standard IBM 3740 format, double frequency (FM) code as well as the double-density DEC RX02 format using the DEC-modified frequency (MFM) technique. Product specifications are given in Tables 2-2, 2-3 and 2-4.

2.4 Cable and Connector Requirements

The DSD 880x/8 is furnished with all internal cables installed and configured for proper operation. A 10-foot long, 26-pin interface cable is supplied for connecting the DSD 880x/8 main chassis to the DSD 8832, 8836, or 8830 computer interface card which is installed at the backplane of the host computer.

Table 2-2. DSD 880 Drive Specifications

	Winchester Drive		Floppy Drive			
	Normal Mode	Extended Mode	Single-Sided Mode		Double-Sided Mode	
GENERAL:	SA1004	SA1004				
Mode Switch Selectable?	Yes	No	No		No	
Emulates	Full RL01	RL02	RX02		'Extended RX02'	
Modifications to DEC Operating Software	None	Section 5	None		See Section 5	
Diskettes used	—	—	Single-Sided		Single- and Double-Sided	
			Single-Density	Double-Density	Single- and Density	Double-Density
Formatted Capacity	5.2 Mbytes	7.8 Mbytes	256 Kbytes	512 Kbytes	up to 512 Kbytes	up to 1 Mbyte
DATA ORGANIZATION:						
Recording format	—		IBM 3740	DEC RX02	IBM 3740	DEC RX02
Recording technique	MFM		Double Frequency	DEC Modified MFM	Double-Frequency	DEC Modified MFM
Bytes/Sector	256		128	256	128	256
Data Integrity	Header CRC/Data CRC		Header CRC/Data CRC			
Bad Track Management	Spare Track Assignment is User Transparent					

Table 2-2. DSD 880 Drive Specifications (Cont)

	Winchester Drive		Floppy Drive
SPEEDS:	Normal Mode	Extended Mode	
Access Times:			
Average	47 msecs	70 msecs	174 msecs
Maximum	107 msecs	150 msecs	410 msecs
Track-to-Track	19 msecs		18 msecs
Head Load Time	—		50 msecs
Head Switching Time	20 microsecs		100 microsecs
Start/Stop Time	5 seconds for disk to reach 95% of nominal speed and 2 minutes maximum for thermal stabilization		2 seconds for diskette rotational speed stabilization
Nominal Rotational Speed	3125 RPM ±3%		360 RPM ± 2%
Average Latency	9.6 msecs		83 msecs
Data Transfer Rate:			
Within a track	142.2 Kbytes/sec		20 Kbytes/sec
across entire disk	106.7 Kbytes/sec		18 Kbytes/sec
burst rate	4 microsec/word plus DMA Overhead		4 microsec/word plus DMA Overhead
Data Transfer Length	5.1K words in normal mode 64K words in extended mode		—

Table 2-3. LSI-11 Interface Board Specifications

LSI-11 Interface	Winchester Drive	Floppy Drive
Device Address:		
Standard (as shipped)	774400	777170
Alternate*	774440, 774420, 774360	777160, 777150, 777140
Hardware Bootstrap Start Address:		
Standard (as shipped)	773000	773000
Alternate*	771000, 766000	771000, 766000
Interrupt Vector:		
Standard (as shipped)	160	264
Alternate*	150, 260, 400	274, 270, 254
Backplane Requirement:	One dual-wide Q-bus slot in any Q-bus backplane	

*Jumper Selectable

Table 2-4. PDP-11 Interface Board Specifications

PDP-11 Interface	Winchester Drive	Floppy Drive
Device Address:		
Standard (as shipped)	RLCS = 774400	RXCS = 777170
Alternate (in word increments of 10 octal)	760000-777770	760000-777770
Bootstrap Base Address:		
Standard (as shipped)	771000	771000
Alternate (in word increments of 1000 octal)	760000-777000	760000-777000
Interrupt Vector:		
Standard (as shipped)	160	264
Alternate (in word increments of 4 octal)	000-774	000-774
Backplane Requirement:	One quad-wide Small Peripheral Controller (SPC) slot in any Unibus backplane	

2.5 Power Specifications

Input Voltage	100 Vac or 120 Vac $\pm$ 10% 220 Vac or 240 Vac $\pm$ 10%	
	50 Hz $\pm$ 1 Hz 60 Hz $\pm$ 1 Hz	
Chassis Current (maximum)	120V/60 Hz	220V/50 Hz
Busy	6A	3A
Starting Current	28A Max @ 115 Vac 14A Max @ 230 Vac	
Heat Dissipation (BTU/hr)	Normal	Maximum
Chassis	1055	1175
Fuse Ratings (all Slo - Blo)	Main	Winchester
	4A @ 120 Vac 2A @ 220 Vac	2A @ 120 Vac 1A @ 220 Vac

INTERFACE

	LSI-11 (Q-Bus)	PDP-11 (Unibus)
Current Consumption (+5V)		
Nominal	2.5A	2.8
Maximum	3A	3.3
Heat Dissipation (BTU/hr)		
Nominal		43
Maximum		52

2.6 Physical Specifications

CHASSIS

Size	Chassis	5.25" H X 17.6" W X 23.74" D (13.3 cm X 44.7 cm X 76.2 cm)	
	Shipping Carton	12.5" H X 24.5" W X 30.0" D (31.75 cm X 62.2 cm X 76.2 cm)	
Weight	Chassis	56.6 lb	(25.7 Kg)
	System Packed for Shipping	80 lb	(36.3 Kg)
Mounting	Rack Slides	Fits in standard DEC rack	

2.7 Environmental Requirements

All disk systems manufactured by Data Systems Design perform efficiently in a normal computer room environment. Temperature, humidity, and cleanliness are three environmental considerations that can affect the reliability of diskette use.

2.7.1. Environmental Specifications

TEMPERATURE			
	Operating	Chassis	41°F to 104°F (5°C to 40°C)
		Diskettes	50°F to 120°F (10°C to 51°C)
		Diskette Maximum Rate of Change	(15°/hr)
	Non-Operating	Chassis	-40°F to 150°F (-40°C to 66°C)
		Diskettes	-40°F to 120°F (-40°C to 51°C)
HUMIDITY		Chassis	10% to 78% (non-condensing)
		Diskettes	8% to 80% (With a maximum wet bulb temperature of 78°F (25.5°C)
ALTITUDE		Chassis (operating)	6000 feet maximum

2.7.2 Cleanliness

Cleanliness is important wherever diskettes are to be stored, handled, and used. Store the diskettes in areas free of dust and corrosive chemicals. The storage area should also be free of strong magnetic fields which might damage the recorded data. When handling a diskette, never touch the exposed magnetic media.

If the DSD 880x/8 is operated in an environment which has a high concentration of abrasive airborne particles, the useful life of the diskettes will be reduced and the data error rate increased.

3.0 INSTALLATION

3.1 General Information

This chapter provides information on unpacking and inspection, installation, configuration, and initial check out of your DSD 880 Data Storage System.

3.2 Unpacking and Inspection

When your DSD 880 shipment arrives, inspect the shipping container immediately for evidence of mishandling during transit. If the container is damaged, request that the carrier's agent be present when the package is opened.

Compare the packing list attached to the shipping container against your purchase order to verify that the shipment is correct.

Unpack the shipping container and inspect each item for external damage such as broken controls and connectors, dented corners, bent panels, scratches, and loose components.

If any damage is evident, notify Data Systems Design Customer Service immediately.

Retain the shipping container and packing material for examination in the settlement of claims, or for future use. Retain the cardboard shipping disk which is installed in the flexible disk drive.

3.3 Power Requirements

The DSD 880 is available in configurations for nominal line voltages of either 120 or 240 Vac. The line frequency must be within 1 Hz (cycles per second) of either 50 or 60 Hz.

NOTE

> The voltage and frequency configuration of the DSD 880 cannot be field modified.

3.4 Installing the DSD 880 Chassis

The DSD 880 chassis must be installed within 10 feet of the interface module's location to accommodate the length of the interconnecting cable. If the computer system operator will be changing diskettes often, it may be convenient to install the chassis close to the console terminal.

The DSD 880 may be either mounted in a standard 19-inch rack or placed on a table top. The rack installation hardware consists of the items listed in Table 3-1.

Table 3-1. Rack Installation Hardware

Quantity	Item
1	Chassis Slide, Left
1	Chassis Slide, Right
2	Slide Mtg. Bracket, Rear
12	Screw, 10-32 X 1/2" Phillips Pan Hd.
4	Screw, 8-32 X 3/8 " Flat Hd.
2	Screw, 8-32 X 1/4" Phillips Pan Hd.
10	Nut, #10 Retainer
4	Hex Nut, 10-32
12	Washer, #10 Flat
4	Washer, #10 Star, External Tooth
2	Washer, #8 Star, External Tooth
2	Captive Screw, 10-32 X 5/8"

The DSD 880 chassis should be mounted in such a way that the air flow behind the fan is unrestricted. The temperature of the air entering the chassis should not exceed 104°F (40°C).

NOTE

The winchester drive furnished as a part of the DSD 880 system is shipped with a "spindle lock mechanism" which is in the locked position to prevent shipping damage. Prior to installation and operation, this lock must be removed. The drive motor can be damaged if power is applied while the spindle is locked.

If the DSD 880 is to be rack mounted, the user should ascertain that the 8840 controller card is configured to meet the desired operating parameters before rack installation is made. The DSD 880 is shipped properly configured for the disk drives furnished with the system, and with the flexible disk drive automatic power on/off option selected.

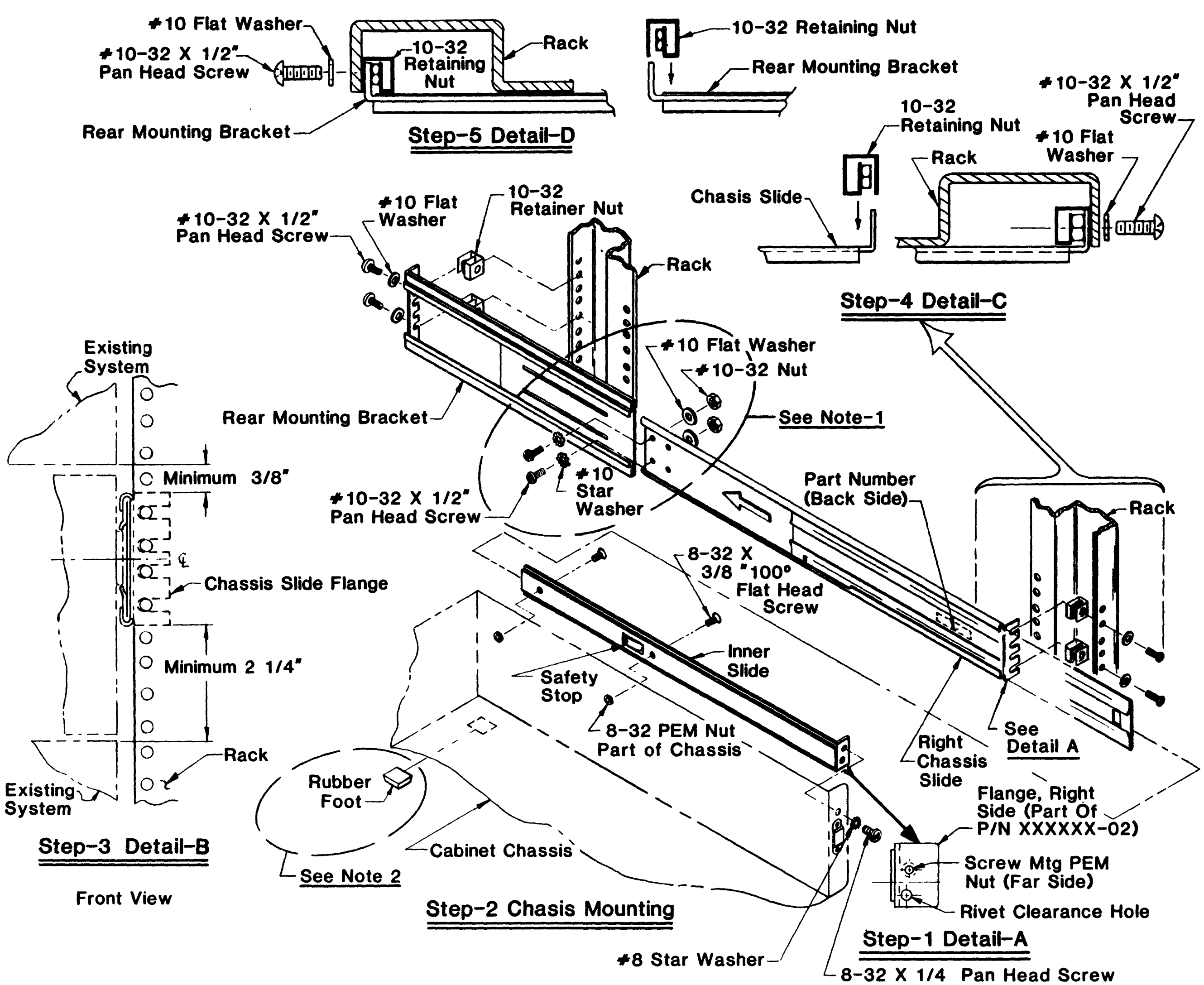
The following procedure should be used to mount the DSD 880 in the standard 19-inch instrumentation rack:

1. Attach the chassis slides to the rack using the hardware supplied. Note that the left and right rear extender brackets are not interchangeable. Figure 3-1 illustrates the correct relationship of the rack mounting components.

2. Insert the DSD 880x/8 into the chassis slides and push the unit into the rack.

3. Remove the front bezel from the DSD 880x/8 and install the retaining screws.

4. Replace the bezel by locating the guide pin and pressing firmly until the retaining mechanism engages firmly.



ASSEMBLY INSTRUCTIONS

STEP-1. Unpack your chassis slide kit and identify the right and left chassis slide by the stamped part no: Left is P/N XXXXXX-01, Right is P/N XXXXXX-02. (See detail-A.)

2. After identifying the right and left chassis slides (see chassis mounting), remove the inner slides by fully extending the slides and then releasing the safety stop. Assemble the inner slides to chassis using the fasteners shown.

3. To position the chassis slides, use the recommended dimensions (see detail-B). The positioning is contingent upon mounting your new system underneath or above the existing system. Align the flange of the chassis slide with the two nearest mounting holes of the rack.

4. After determining which two holes/slots will be used, slide the retaining nuts into the appropriate slots on the mounting flange of the chassis slides (see detail-C). Fasten the chassis slides to the rack using the fasteners shown.

5. Slide the rear mounting bracket over the chassis slide until the flange meets the back of the rack. Align the bracket with the two nearest mounting holes on the rack. It is important to keep the slide and rear bracket level.

NOTE: The rear has the same slot spacing relative to the center of the chassis slide. Slide the retainer nuts into the appropriate mounting slots, re-align the bracket to the holes and fasten with the hardware shown. (See detail-D)

NOTE-1. For the extra long racks, additional hardware has been supplied for stiffening the assy.

2. Remove rubber feet from system before installing into rack.

Figure 3-1. Installing Chassis Slides

3.5 Interface Module and Cable Installation

3.5.1 LSI-11 (8832/8836) Interface Configuration and Installation

CAUTION

> Ensure that system power is off before installing the interface module and cable, or before changing the interface switch positions.

The DSD 880 LSI-11 interface card is a dual-wide card, labeled P/N 808836. The DSD 8832 is shown in Figure 3-2, and the 8836 in Figure 3-3. The following features can be selected through jumpers on the LSI-11 interface card. Refer to Table 3-2.

1. RL Device Register Addresses
2. RX Device Register Addresses
3. Bootstrap Base Addresses
4. Bootstrap Enable/Disable
5. RL Interrupt Vector Addresses
6. RX Interrupt Vector Addresses
7. DMA Burst Length
8. RL Interrupt Priority Level

When performing a bootstrap load function on the LSI-11, which contains a line-time clock (LTC), a problem arises because the LTC is continuously interrupting (at a 60 Hz rate) and the LSI-11 computer powers-up with interrupts enabled. This means that an LTC interrupt can occur before the bootstrap can load the proper interrupt vectors.

The 8832/8836 boards contain logic that permit booting with the LTC enabled. The circuit, on power-up, clamps the Q-bus interface line BEVENT L low, preventing LTC interrupts. DEC operating software normally polls for the presence of BDV11 LTC card, with a bus adress of 777546. The 8832/8836 senses any access of this address and removes the clamp on BEVENT L, thus permitting normal operation of the LTC. However, the 8832 /8836 does not reply to this address.

If customer applications require, the clamping of BEVENT L may be disabled by cutting trace J-4 on the 8832/8836 board. See Figure 3-3 for location of J-4 cut trace option.

The 8836 Q22 upgrade has been incorporated to support 22 bit addressing. The 8836Q now monitors the DMA request line on the Q-bus and aborts any ongoing transfer within four microseconds of DMA request going low. Other devices will get as much bus time as needed without contention by the 8836 to cause data late problems. The 8836Q supports the fifth register defined by the DEC RLV12 controller and works compatibly with the PDP-11/23+ with 22 bit addressing.

The RLCS register is forced to be at an address that is a multiple of 20 (774360, 774400=standard, 774420=alternate, and 774400) and results in changes in the alternate RLCS addresses (see Table 3-2).

The 8836Q also allows using the 880 floppy directly in a 22-bit address context by providing a third register, RXBAE at 777174 (RXCS+4), which allows loading the six extended address bits.

Note that both RLBAE (RLCS+10) and RXBAE (RXCS+4) share their low two bits with the extended address bits, A16 and A17, as loaded into RLCS and RXCS. A write into RXCS loads the low two bits of RXBAE and a write into RXBAE loads the extended address bits previously set by writing into RXCS. The same is true for RLCS and RLBAE.

The 22-bit address extension is totally downward compatible. If these bits are not set, everything defaults to the low 256 Kbytes.

Fast mode begins attempting 16 word bursts (50 microseconds) and aborts before the next word, when DMA request is sensed. If DMA request is set within the first ten microseconds (three words) of a burst, the 8836 will hold off re-acquiring the bus for approximately eight microseconds. If DMA request is sensed later, the hold-off will be approximately two microseconds.

Throttle mode (F jumper cut) continues two word transfers as before, but also aborts the second word on DMA request. It then gets off the bus for seven to ten microseconds.

3.5.2 LSI-11 (DSD P/N 808832 or DSD P/N 808836) Interface Installation Procedure

The following procedure describes how to install the LSI-11 interface module:

1. VERIFY LINE POWER IS OFF.

2. Plug one end of the interface cable into the interface module so that pin 1 (the striped side) is closest to the edge of the board. Note that the position of the clipped pin on the module connector matches the position of the plugged hole on the cable connector.

3. Plug the opposite end of the interface cable into the keyed connector mounted on the rear panel of the chassis. Note that the position of the clipped pin on the module connector matches the position of the plugged hole on the cable connector.

Now you are ready to plug the module into the lowest numbered available Q-bus slot.

NOTE

No open Q-Bus slots are allowed between the processor and the DSD 8836 interface module. Since this module uses both interrupts and Direct Memory Access (DMA), a break in either of the grant propagation chains will prevent the interface module from obtaining control of the Q-Bus. Figure 3-4 shows how Q-Bus slots are numbered on the standard backplanes available from DEC. Some Q-Bus interface cards (e.g., serial interfaces and memory) do not pass the DMA grant signal. Ensure that the DMA signal is reaching the LSI-11 interface (8836).

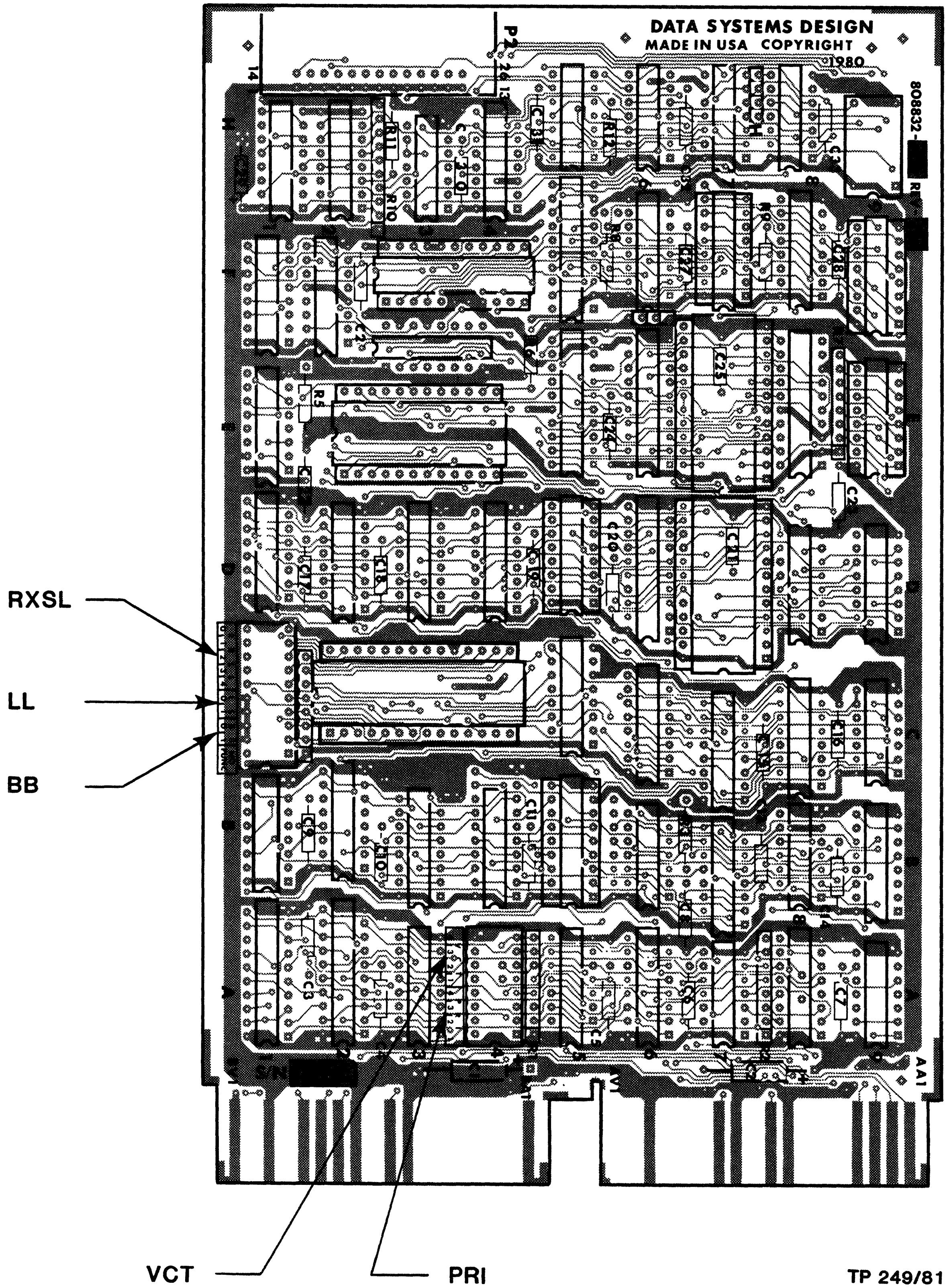
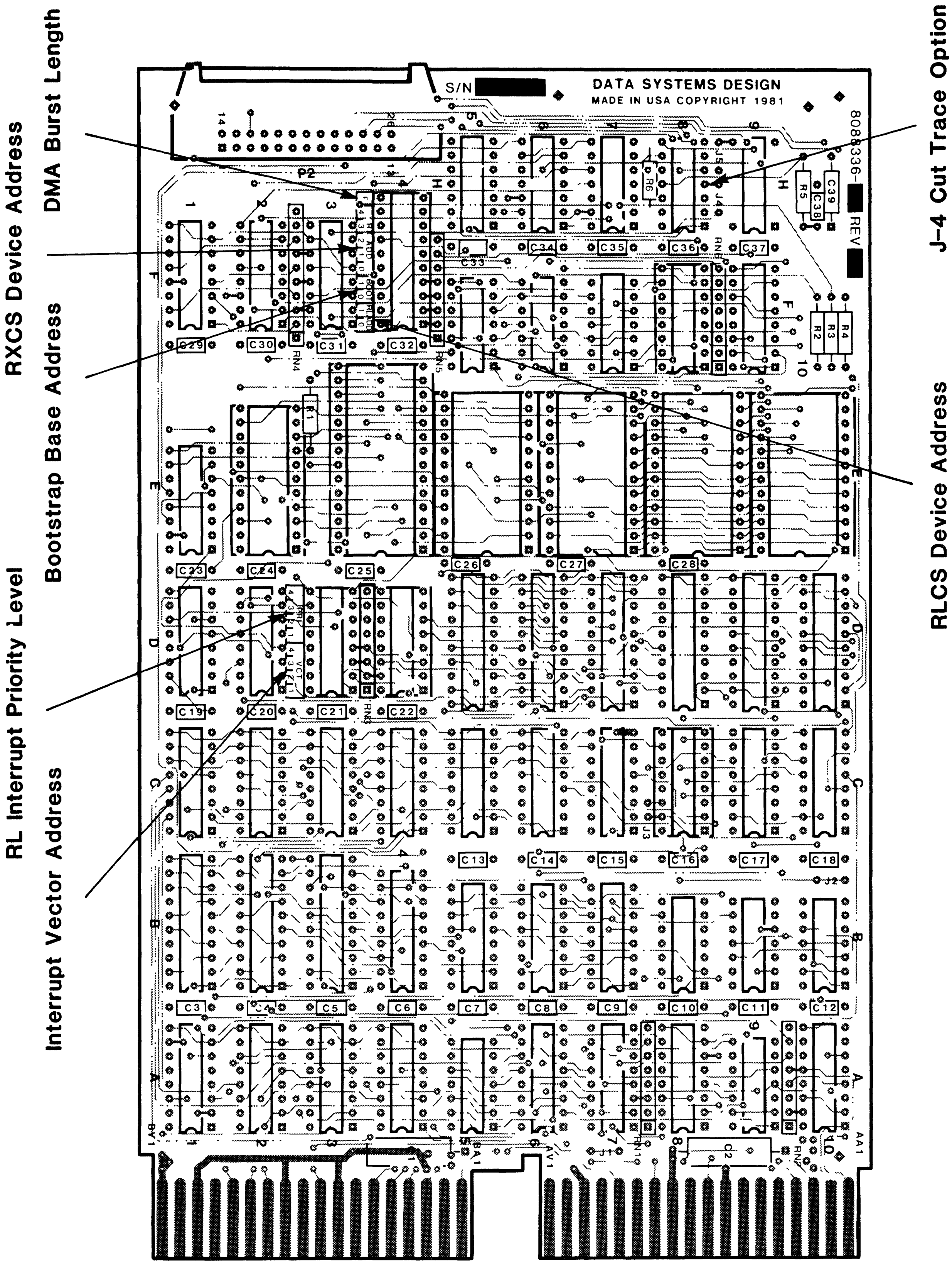


Figure 3-2. DSD 8832 Computer Interface Card



TP 229/81

Figure 3-3. DSD 8836 (LSI-11) Interface Card

Table 3-2. DSD 8832/8836 Interface Jumper Settings

(Refer to Figure 3-3 for jumper locations)

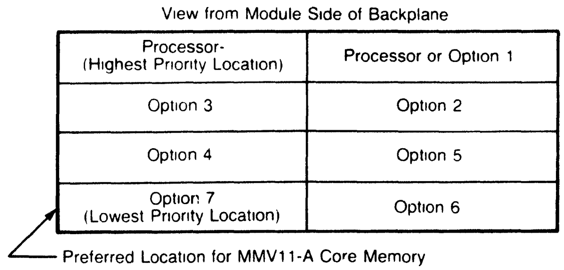
IC Position F4
Shown as shipped

	4	3	2	1	0	1	0	1	0	
										RLCS Device Address
								O	O	774440
								O	S	774360
								S	O	774420
								S	S	774400 (Standard)
										Bootstrap Base Address
						O	O			166000
						S	O			171000
						O	S			Disable Bootstrap
						S	S			173000 (Standard)
										RXCS Device Address
	S	O	O	O	O					177150
	O	S	O	O	O					177140
	O	O	S	O	O					177160
	O	O	O	S	O					177170 (Standard)
	O	O	O	O	S					Disable RX
										DMA Burst Length (8836 Only)
O										Two word burst (3-way interleaving)
S										Eight word burst (supports 2-way interleaving)
	O	O	O	S	O	S	S	S	S	As shipped by DSD

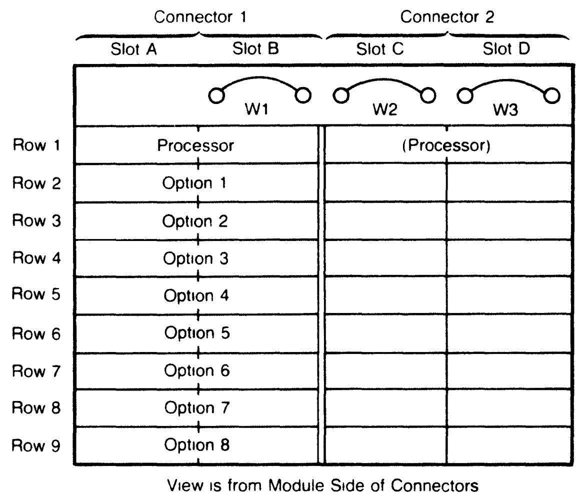
IC Position D3
Shown as shipped

PRI 4	PRI 3	PRI 2	PRI 1	VCT 4	VCT 3	VCT 2	VCT 1	
								RL Interrupt Vector Address
						O	O	150
						S	O	260
						O	S	400
						S	S	160 (Standard)
								RX Interrupt Vector Address
				O	O			274
				S	O			254
				O	S			270
				S	S			264 (Standard)
								RL Interrupt Priority Level
S	S	S	O					7
S	S	O	S					6
S	O	S	S					5 Required on LSI-11/23 with RSTS or UNIX (Standard)
O	S	S	S					4
								RX Interrupt Priority Level (Fixed at 5)
S	O	S	S	S	S	S	S	As shipped by DSD

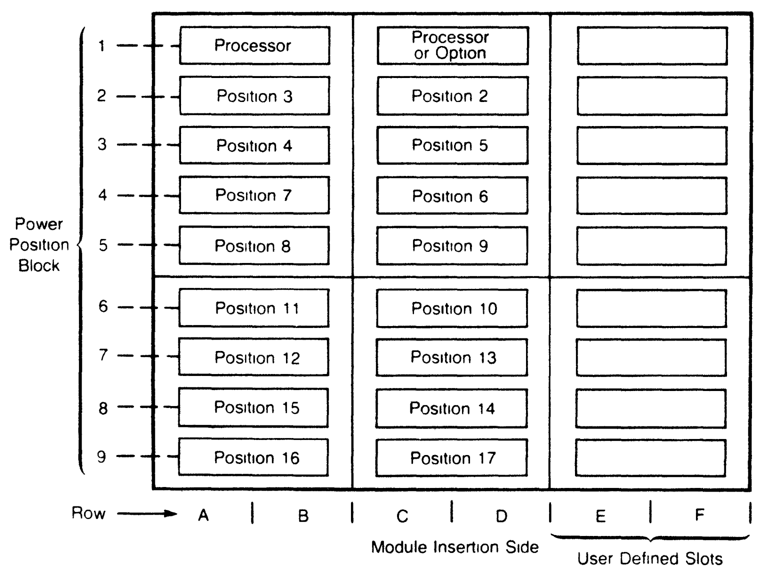
S = Short
O = Open



DEC Backplane H9270



DEC Backplane H9273-A



DEC Backplane DDVII-8

TP 231/81

Figure 3-4. Option Priority in LSI-11 Backplanes

3.5.3 PDP-11 (8830) Interface Jumper Configuration

The DSD 880 PDP-11 interface card is a quad-wide card, labelled P/N 808830. The DSD 8830 is shown in Figure 3-5.

The following features can be selected through jumpers on the PDP-11 interface card. Refer to Tables 3-3, 3-4, and 3-5.

1. RL Device Register Addresses
2. RX Device Register Addresses
3. Bootstrap Base Address
4. Bootstrap Disable
5. RL Disable
6. RX Disable
7. RL Interrupt Priority Level

3.5.4 PDP-11 (DSD P/N 808830) Interface Installation Procedure

The following procedure describes how to install the PDP-11 module.

1. VERIFY LINE POWER IS OFF.

2. Check that the jumpers on the interface module are configured correctly.

3. Plug one end of the interface cable into the interface module so that pin 1 (striped side) is closest to the module handle.

4. Confirm that the position of the clipped pin on the module connector matches the position of the plugged hole on the cable connector.

5. Plug the module into a convenient Small Peripheral Controller (SPC) slot using connectors C, D, E, and F.

6. Verify that there are no open SPC slots between the DSD 8830 interface and the processor. Each slot between the 8830 interface and the processor must be occupied by either an interface board or a bus grant continuity card. Bus grant continuity cards plug into connector D of an SPC slot. See Figure 3-6. The DSD 880 system will not operate with open SPC slots between the interface and the processor.

7. Insure there is no backplane jumper or foil trace between backplane pins CA1 and CB1 of the SPC slot selected for the DSD 8830 interface board. SPC slots are not wired for DMA devices. The Non-Processor Grant (NPG) bypass jumper must be removed for DMA devices such as the 8830 interface to operate.

If the 8830 interface board is removed from the backplane, a jumper wire connecting pins CA1 and CB1 must be reinstalled to provide NPG continuity to devices along the chain. A bus grant continuity card will not replace this jumper.

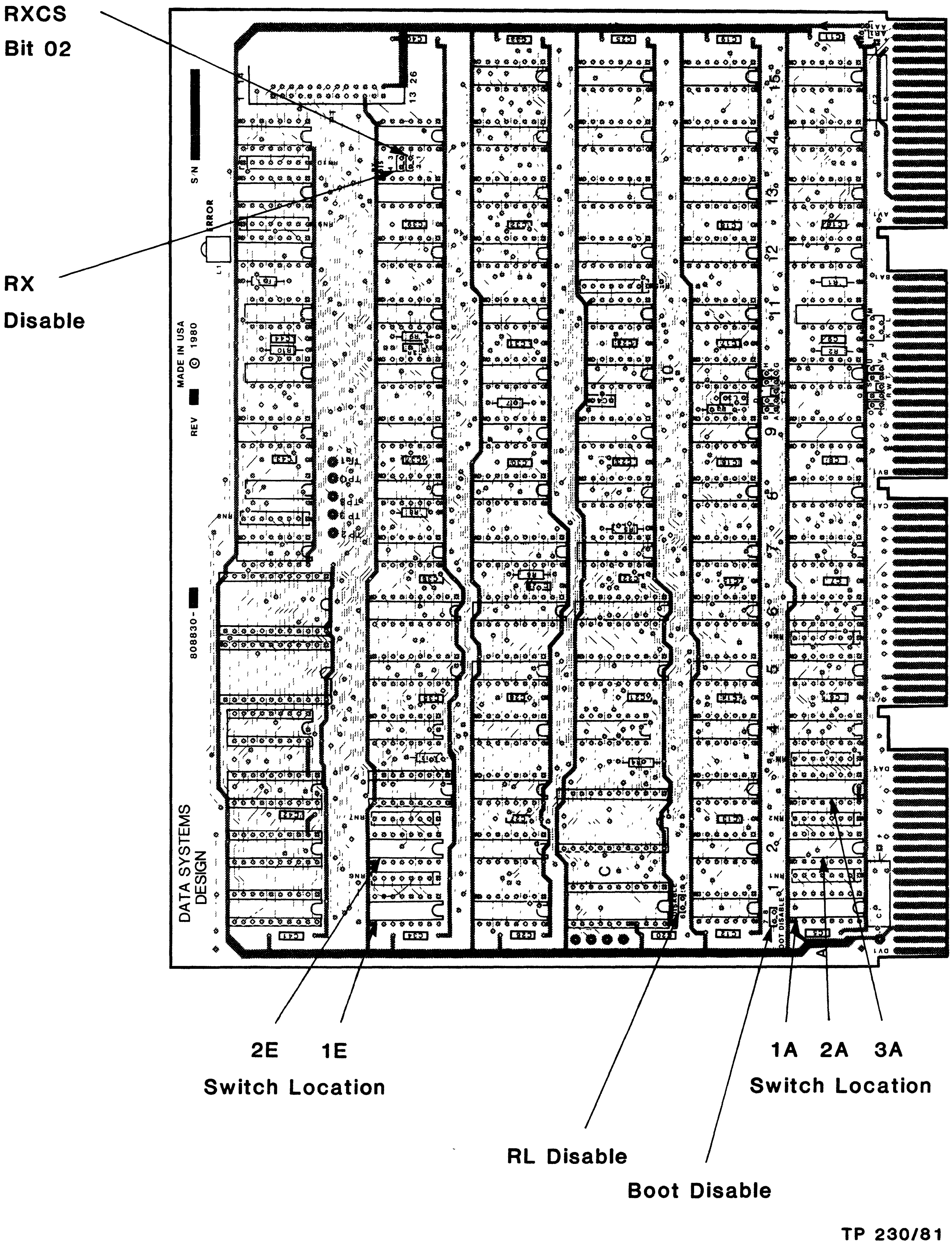


Figure 3-5. DSD 8830 (PDP-11) Interface Card

Table 3-3. PDP-11 Interface Jumper Settings

S = Short
0 = Open

1A	2A	3A	1E	2E		
1 2 3 4 5 6 7 8	1 2 3 4 5 6 7 8 *	1 2 3 4 5 6 7 8	1 2 3 4 5 6 7	1 2 3 4 5 6 7		
					RXCS Device Register Address	
0 0	0 0 0 0 S S 0 0 S				777170	(Standard)
0 0	S 0 0 0 S S 0 0 S				777160	
0 0	0 S 0 0 S S 0 0 S				777150	Typical Alternates
0 0	S S 0 0 S S 0 0 S				777140	
X S	X X X X X X X X X				76XXX0	
X 0	X X X X X X X X X				77XXX0	
					RLCS Device Register Address	
0 0		S S S S S 0 S S			774400	(Standard)
0 0		0 S S S S 0 S S			774410	
0 0		S 0 S S S 0 S S			774420	
0 0		0 0 S S S 0 S S			774430	
X S		X X X X X X X X			76XXX0	
X 0		X X X X X X X X			77XXX0	
					Bootstrap Base Address	
0 S S 0					771000	(Standard)
S 0 S 0					772000	
0 S 0 0					775000	
S 0 0 0					776000	
					RX Interrupt Vector Address	
			0 S 0 0 S 0 S		264	(Standard)
			S S 0 0 S 0 S		260	
			0 0 S 0 S 0 S		254	
			S 0 0 0 S 0 S		230	
					RL Interrupt Vector Address	
				S S 0 0 0 S S	160	(Standard)
				0 0 S 0 0 S S	154	
				S 0 S 0 0 S S	150	
				0 S S 0 0 S S	144	

*Jumper pins 1 and 2 between ICs 13E and 14E. Opening this jumper causes address bit 2 on RXCS to be a don't care. The orientation of the switches for this diagram do not correspond to their orientation on the PCB.

Table 3-4. 8830 Interrupt Priority Settings

Connections	Standard*	Alternate		
	Priority 5	Priority 4	Priority 6	Priority 7
N to J	Open	Short	Open	Open
N to K	Short	Open	Open	Open
N to L	Open	Open	Short	Open
N to M	Open	Open	Open	Short
O to P	Short	Open	Short	Short
Q to R	Open	Short	Short	Short
S to T	Short	Short	Open	Short
U to V	Short	Short	Short	Open
W to P	Open	Short	Open	Open
W to R	Short	Open	Open	Open
W to T	Open	Open	Short	Open
W to V	Open	Open	Open	Short
A to B	Short	Open	Short	Short
C to D	Open	Short	Short	Short
E to F	Short	Short	Open	Short
G to H	Short	Short	Short	Open
I to A	Open	Short	Open	Open
I to C	Short	Open	Open	Open
I to E	Open	Open	Short	Open
I to G	Open	Open	Open	Short

*8830s are shipped fabricated to priority 5.

Use at any other priority requires the following:

1. Cut required connections open.
2. Insert 0.025" square wire-wrap pins at appropriate connection points.
3. Wire wrap required connection closed.

Table 3-5. 8830 Jumper Configurations

8830 jumpers are shipped configured for a standard configuration where RX, RL, and BOOT are enabled and RXCS address bit is fixed at D.

Jumper		Function	In	Out	Shipped
Number	Location				
1-2	13E	RXCS address bit 2	0	Don't Care	In
3-4	13E	RX Disable	Disable	Enable	Out
5-6	1C	RL Disable	Disable	Enable	Out
7-8	1B	BOOT Disable	Disable	Enable	Out

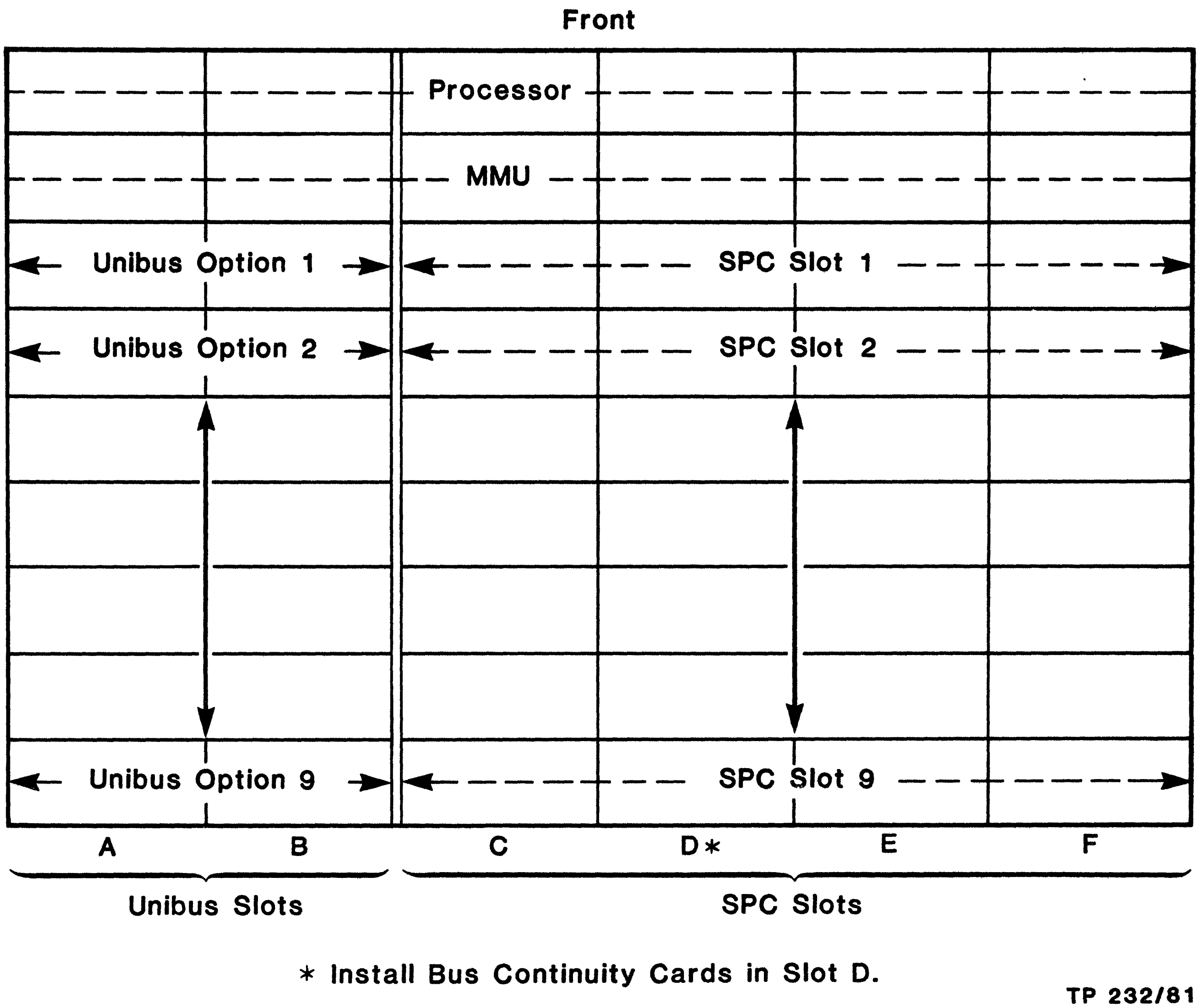


Figure 3-6. Typical Unibus Hex Backplane

3.6 AC Power Cord Installation

To install the ac power cord:

1. Ensure that the DSD 880 power on/off switch is in the off position.

2. Plug the female end of the power cord into the connector on the back of the DSD 880 chassis.

3. Plug the male end of the power cord into an ac power receptable that provides the proper ac input voltage for the DSD 880 (90 to 130V rms, on domestic models, or 198 to 250V rms on international models configured for the higher voltage.)

3.7 Initial Checkout and Acceptance Testing

After installation of the DSD 880, an initial power-up and testing sequence should be completed prior to placing the system into regular service. Be sure the winchester spindle lock and head lock have been removed prior to operation. DSD recommends the following procedure be followed:

NOTE

> Prior to applying power and performing acceptance testing, the operator should be familiar with the normal operating procedures of Section 4 and the use of DSD HyperDiagnostics tests in Section 7 of this manual.

1. Remove the DSD 880 front bezel by grasping the bezel and pulling forward. Removal of the front bezel will allow access to the HyperDiagnostics panel.

2. Assure either that power is applied to the host computer, or that the interface cable is not connected.

3. Apply power to the DSD 880 using the power on/off switch on the rear panel of the chassis.

4. Insert a blank, write enabled, floppy disk into the floppy disk drive.

NOTE

> Any data present on the floppy disk used in the following sequence of tests will be destroyed during the tests.

5. Perform the DSD 880 HyperDiagnostic Switch and Light test using the procedure that follows:

   A. Place the floppy and winchester write protect switches in the off position, select MODE = 3, CLASS = 0 and depress the EXECUTE pushbutton. Verify that 30 is displayed by the seven segment displays.

   B. Observe the fault, winchester ready, floppy fault, winchester fault, and floppy write protect indicators. Verify that each illuminates and extinguishes independently of the other indicators before proceeding.

   C. Rotate the MODE switch through positions zero through seven, verify that the switch position is displayed by the left digit of the seven segment displays.

   D. Rotate the CLASS switch through positions zero through seven, verify that the switch position is displayed by the right digit of the seven segment displays.

   E. Place the floppy write protect switch in the on position, verify that the floppy write protect and floppy fault indicators illuminate, and that the value 88 is flashing in the seven segment displays.

   F. Place the floppy write protect switch in the off position and the winchester write protect switch in the on position. Verify that the winchester write protect and winchester fault indicators illuminate, and that the value 99 is flashing in the seven segment displays.

   G. Place the winchester write protect switch in the off position.

6. If no malfunctions are detected during the 880 switch and light test, perform the DSD 880 HyperDiagnostic sequential scan floppy disk (50) and sequential scan fixed disk (54) tests as given in Section 7 of this manual.

   If no errors are detected during the sequential scan floppy disk (50) test cycle, the DSD 880 will halt with 00 displayed in the seven segment display. The sequential scan fixed disk (54) test runs until halted. If an error is detected during any portion of the test sequence, the DSD 880 will halt with an error code flashing in the seven sector display. For an explanation of each of the tests and for the meanings of any error codes displayed refer to Section 7 of this manual.

7. Select the desired normal operating MODE and CLASS (see Table 4-2), then depress the EXECUTE pushbutton momentarily. The selected MODE and CLASS will be displayed while the EXECUTE pushbutton is depressed. Upon release of the EXECUTE pushbutton, verify that the code 00 is displayed, indicating that both the floppy and winchester drives were successfully initialized.

8. Reconnect the interface cable and apply power to the host computer if necessary.

3.8 DSD 880 Initial Program Installation

This section provides a description of the DSD supplied software available and guidance in the integration of the DSD 880 into the user's operating system.

3.8.1 DSD Supplied Programs

The DSD 880 is shipped from the factory preformatted with bad track and bad sector file information on the winchester. A floppy diskette is included which contains the DSD supplied programs and command files. Several of these programs are also shipped on the winchester as an aid in initial testing of the DSD 880. Appendix A contains a directory listing of these devices/diskettes.

The main programs supplied are:

FLPEXR - a stand alone diagnostic/utility program for operations on the floppy drive. See Appendix C.

RLEXR - a stand alone diagnostic program for operations on the winchester drive in RL emulation mode. See Apendix D.

WINEXR- a stand alone diagnostic/utility program for operations on the winchester drive in direct access mode and for disk formatting and bad track mapping. See Appendix E.

DSDMON - a bootable diagnostic monitor that allows the user to select one of the diagnostic programs for execution. See paragraph 3.8.3.

3.8.2 Command Files

Command files are supplied for the main operations necessary to utilize the extended features of the DSD 880 and to assist the user in the initial loading of the operating system onto the DSD 880. A full listing of each command file is contained in Appendix B of this manual. Usage of each command file is described in the appropriate section of the manual.

Command files are also provided to facilitate backup and restores of the DSD 880 winchester. These command files should be considered as representative only; individual users should tailor the commands to their particular needs. These files are called 88XFLP.COM, FLPX88.COM.

3.8.3 Use of DSDMON

DSDMON is the DSD diagnostic monitor program that allows the user to select which diagnostic program is to be executed from the distribution diskette. It is a secondary bootstrap program that loads RT-11 format files into memory and initiates execution of that program. Although DSDMON accesses files through an RT-11 type format, RT-11 is not required to run DSDMON.

To initiate a program, boot the diskette through the hardware bootstrap procedures. The program will output on the console:

DSD DIAGNOSTIC MONITOR PROGRAM V3A

DSDMON

The program to be initiated is specified by typing:

R filename <CR>

DSDMON assumes an extention type of .SAV. If the file is not found on the diskette, DSDMON will output:

FILE NOT FOUND

If the file is found on the diskette, it will be brought into memory and execution begun. DSDMON also supports the following commands:

T	filename<CR>	- Types the specified file contents on the console terminal
H	<CR>	- Types a help file on the console terminal
R	filename<CR>	- Load and run specified program
L	filename<CR>	- Load specified file then return control to DSDMON

DSD supplied diagnostics are configured such that, if they are initiated from an RT-11 system, control can be returned to RT-11. If invoked from DSDMON, they will still prompt for "RETURN TO RT-11?", however, such return is not possible and a Y (yes) reply will cause the diagnostic to be reinitialized. In order to run a different diagnostic, DSDMON must be booted again. DSDMON can be restarted at the last location in memory (for a 28K word system, this address is 157776).

3.8.4 Transfer of RT-11 to DSD 880

A. Transfer of RT-11 V3B to the 880 winchester:

1. Procure a DY bootable RT-11 distribution diskette with a DL handler (DL.SYS) on it.

2. Boot this diskette and prepare to copy all files onto the 880 winchester.

NOTE

The 880 winchester as shipped contains an RT-11 directory and all the diagnostic diskette files. These may be retained by skipping the following step.

INIT DL0:/NOQ<CR>

3. Copy all the RT-11 files on the distribution disk onto the 880 winchester.

   COPY/SYS DY0:*.* DL0:<CR>

4. If the bootable RT-11 V3B distribution diskette does not contain a DL monitor, then it must be copied from one of the other distribution diskettes (#2 or #3).

   This can be done most easily if another device is available to use as a system device. If only the DSD 880 is available, proceed as follows:

   a) .SET USR NOSWAP <CR>
      .R DIR <CR>
      *
      Remove the bootable system disk.
      Write protect the floppy drive using the front panel switch. Insert the other distribution disks one at a time and type:

      *DY0:/B/E <CR>

      Determine the disk containing DLMNSJ.SYS and note the starting block number and length.

      Example:

      DLMNSJ.SYS   74   150

      Where 74 is the length and 150 is the starting block number.

5. Make the 880 winchester hardware bootable:

   COPY/BOOT DL0:DLMNSJ.SYS DL0:<CR>
   or
   COPY/BOOT DL0:DLMNFB.SYS DL0:<CR>

   Remove the distribution disk containing DLMNSJ.SYS.
   Reinsert the bootable disk first booted on.
   Unprotect the floppy drive using the front panel switch.

      Type:<CTRL C>
      .LOA DL:<CR>
      .R DUP<CR>
      *DL0:DLMNSJ.SYS=/C:4000.:64.<CR>
      *DL0:A=DY0:/I: (starting read block):(starting read block and length of file):(starting write block)=4000.

      For example, with the starting block and length given in the directory example:

      *DL0:DLMNSJ.SYS/I:150.:214.:4000./W<CR>

      The system will ask "CONTINUE?"
      Remove the bootable system disk and insert the diskette containing DLMNSJ.SYS found above.

Type: Y <CR>
The system will copy the blocks specified on the 880 winchester and type: "INSERT SYSTEM DISK, ARE YOU READY?"

Remove the other distribution diskettes and insert the bootable system diskette.

Type: Y <CR >

There should now be a copy of the DL monitor (DLMNSJ.SYS) on DL0.

3.8.5 Transfer of RT-11 V4 to the 880 Winchester

1. Boot the bootable distribution diskette in DY0:.

2. Prepare to copy the RT-11 V4 distribution diskette contents onto the winchester.

NOTE

The DSD 880 winchester is shipped with a copy of the DSD diagnostic disk on the winchester.

These contents may be retained by skipping the following step:

Type: INIT DL0:/NOQ<CR>

3. Copy all files from the floppy to the 880 winchester.

   .COPY/SYS DY0:*.* DL0:<CR>

4. Bind the device monitor to the DL handler to make it bootable.

   .COPY/BOOT DL0:RT11SJ DL0:<CR>

5. Bootstrap the RT-11 on the 880 winchester.

   .BOOT DL0:<CR>

3.8.6 Double-Sided Support Under RT-11 (Version 3B)

Double-sided support under RT-11 V38 may be activated by one of two methods. DSD supplies a software device handler which is equivalent to the DEC device handler with appropriate flags and conditionals enabled for double-sided support. This handler may be assembled into the RT-11 DY monitor (FB or SJ) by following the system generation procedure as supported by DEC. Alternately, to save the effort required to perform a SYSGEN, DSD supplies a command file which will automatically patch the RT-11 V3B monitor to activate the two-sided features.

If the user elects to perform a SYSGEN, the DSD handler DYDSD.MAC (found on the DSD diagnostic diskette) must first be renamed to DY.MAC and substituted for the

MACRO-11 source file, DY.MAC provided by DEC. The DSD handler, containing double-sided support may then be installed into the RT-11 monitor by following the procedure described in the RT-11 System Generation Manual supplied by DEC.

Note that the actual monitors (DYMNSJ.SYS or DYMNFB.SYS) must reside on side 0 in order to boot initially.

DOUBLE-SIDED SUPPORT UNDER RT-11 V3B

A. Nonsystem for side 1.

The file DYDSD.SYS on the diagnostic disk is an RT-11 V3B handler compatible with the distribution kit monitors that can be copied over to the winchester for use.

1. Boot RT-11 V3B on the 880 winchester.

2. Insert the diagnostic disk into DY0:.

3. Copy the RTV3B DY handler over to the winchester.

   .COPY DY0:DYDSD.*/SYS DL0:DY.*<CR>

4. Reboot the DL monitor.

   .BOOT DL0:<CR>

   This installs the double-sided handler.

3.8.7 DSD Monitor Patch Program RT-11 V3B

The monitor patch program takes a DYMNSJ or DYMNFB monitor from the DEC RT-11 V3B system distribution and replaces the DY handler currently in the distribution monitor with a double-sided DY handler. The new monitor has the same characteristics as the original monitor, such as batch support, 60 Hz line time clock, all handlers supported by the distribution monitor, and no error logging.

The monitor patch program would be used under the following conditions:

1. The distribution RT-11 V3B monitor provided by DEC is sufficient for the user's normal applications, except for not having double-sided support.

2. The user does not wish to perform a system generation.

3. The user has not changed the normal distribution monitor with customized patches, relating to the the user's system.

If these conditions are not met, a system generation may be required.

The DYMNSJ or DYMNFB monitor may be generated from the first or second release of RT-11 V3B. The distribution DYMNSJ or DYMNFB monitor that will be modified can be found on the distribution diskette as shown in the following:

First DX kit release of RT-11 V3B	Disk Label No.	Dated
DYMNSJ.SYS	AS-5781B-BC	11-Mar-78
DYMNFB.SYS	AS-5781B-BC	11-Mar-78

Second DX kit release of RT-11 V3B	Disk Label No.	Dated
DYMNSJ.SYS	AS-5783C-BC	27-Mar-79
DYMNFB.SYS	AS-5783C-BC	27-Mar-79

or either DY kit release may be used.

To use the DSD monitor patch procedure on the DSD 880:

1. Boot RT-11 V3B on the 880 winchester. Note that the default device DK: should be the system device floppy.

2. Copy the desired DY monitors from the DEC floppy distribution kit onto the 880 winchester (DYMNSJ.SYS and DYMNFB.SYS).

3. Copy the PAT files from the DSD diagnostic diskette onto the winchester.

   Insert the diagnostic diskette and type:

   .@DY0:PATSET<CR>

4. Put a blank diskette in DY0: and set to double-density. Note that the DEC format program only supports the standard device addresses. Use DSDFMT if an alternate address is to be used.

   ```
   .R FORMAT
   *<CTRL C>
   ```

5. Determine which double-sided monitor is to be generated. Type:

   .@PATSJ<CR>

   to put a single-job monitor on DY0:, or type:

   .@PATFB<CR>

   to put a foreground/background monitor on DY0:.

   Note: Both steps 4 and 5 should be repeated if both double-sided monitors are to be created.

   This procedure will copy a minimal system over to the floppy in DY0:, then patch and boot that monitor. This diskette then contains the selected RT-11 V3B monitor with double-sided support and should be used as a master for generating other double-sided bootable diskettes.

NOTE

RT-11 V3B will not boot a floppy with the selected monitor on the second side.

3.8.8 Double-Sided Floppy Support Under RT-11 V4

A command documentation file DYV4DS.DOC is provided which applies the diference to the DEC distribution DY.MAC given in DYV4DS.DIF.

To update the RT-11 V4 DY handler:

1. Boot RT-11 on the 880 winchester.

2. Copy DY.MAC from the DEC distribution kit onto the winchester.

3. Copy DYV4DS.* from the DSD diagnostic disk onto the winchester.

4. Type @DYV4DS.DOC.

3.8.9 Extended Mode Winchester Support

The DSD 880 operates in either RL01 compatible mode or extended mode which is a subset of an RL02. No changes to the RT-11 system are required to use the RL01. compatible mode. By using the RT-11 FILE.BAD capability to mask the unavailable disk area, extended mode can also be used without any operating system patches. If the FILE.BAD approach is not acceptable, then only a few minor patches to the DL handler will allow the extended mode operations.

Command file DLV388.DOC contains the procedures to be followed to enable extended support for RT-11 Version 3B. For RT-11 Version 4, command files DLV488.DIF and DLV488.DOC contain the changes to be applied to the RT-11 distribution handler sources using the SLP program.

3.8.10 Transfer of RSX-11M to DSD 880 Winchester

In order to bring up RSX-11M on the DSD 880, a host machine capable of reading the DEC distribution kit is required. There are several methods of transfer from this machine/disk onto the DSD 880:

1. SYSGEN with DSD 880 attached as an RL01/02 to the host machine.

2. SYSGEN with floppy drive and RL01/02 attached to the host machine.

3. SYSGEN on host system with only floppy drive available as an intermediary device.

The following paragraphs describe these methods in more detail.

SYSGEN with DSD 880 attached as an RL01/02 to the host machine

This is the most convenient method in that standard SYSGEN procedures can be followed for generating a target system. If the DSD 880 is in extended mode, the host monitor device-size table should be updated before running BAD and INI on the DSD 880. (See RSX-11M extended support section for details.)

SYSGEN with floppy drive and RL01/02 attached to host machine

Perform a SYSGEN with the RL01/02 as the target device. If only an RL01 is available and the end target is an extended DSD 880, run BAD while in RL01 mode, then apply the device-size updates before performing INI. Set the directory to the middle (default case) or beginning of the volume. After performing the software boot, change the device size in the new monitor before doing the SAV/WB. Alternatively, use DSC for the final expansion to the final device size as above. THE RSX image can be transferrd to the DSD 880 using either of the methods described below.

If RT-11 Version 4 is available, generate a system supporting the RL01/02 and floppy drive. Bring up this RT-11 system. Copy the RSX system image (on RL01/02) out onto multiple floppies then onto the DSD 880 using the RT-11 V4 indirect command files provided (88XFLP.COM and FLPX88.COM) on the DSD distribution disk. After these disks are copied onto the DSD 880, the DSD 880 will contain an image copy of the original RL01/02 and can be hardware booted into RSX-11M.

If the DSD copy utility is available, copy the RSX system image onto multiple floppies. These floppies can then be loaded onto the DSD 880 by using the DSD 880 restore mode of operation. AFter these disks are loaded, the DSD 880 will contain an image copy of the original RL01/02 and can be hardware booted into RSX-11M.

SYSGEN on host system with only floppy drive available

This method requires generating a floppy disk containing an RSX-11M system which is then booted using the DSD 880 floppy drive to produce an operational floppy based RSX-11M nucleus. The DSD 880 winchester drive is then setup from this nucleus and booted. The floppy can then be used to transfer the remaining files onto the winchester.

This procedure is most easily done in one SYSGEN if both floppy drive and RL01/02 handlers are set as loadable. This allows the final usable RSX11M.SYS images to be brought up by simply interchanging the LOA DL: and LOA DY: commands to VMR.

NOTE

The handler for the physical volume to be VMR'd upon must be the first file structured handler to be loaded. If not, then when tasks are to be installed, the message

INSTALL DEVICE NOT LBO:

will be output independent of any assignment command.

This procedure requires either a double-sided, double-density disk on the DSD 880, or two single-sided, double-density disk drives.

The following are the minimum complement of files required for DL volume initialization (602. blocks total):

RSX11M.SYS	258. blocks
FCPMD1.TASK	62.
COT.TSK	24.
INI.TSK	34.
BAD.TSK	50.
UFD.TSK	7.
MOU.TSK	24.
MCR.TSK	28.
LOA.TSK	29.
PIP.TSK	69.
DYDRV.TSK	5.
DYDRV.STB	1.
DLDRV.TSK	4.
DLDRV.STB	1.

The following files are required for the VMR phase and can be copied over individually as necessary.

RSX11M.STB	11.blocks
RSX11M.TSK	130.
LDR.TSK	5.
TTDRV.TSK	18.
TTDRV.STB	5.
SAV.TSK	65.
BOO.TSK	22.
INS.TSK	27.
VMR.TSK	142.
IND.TSK	101.

Appendix A contains a directory listing of a double-sided, double-density floppy disk that includes all files needed for both booting from the DY: and the final VMR of the DSD 880 winchester.

Once the DL volume is initialized and UFDs have been created, aditional files can be transferred from the floppy to the winchester as necessary. Appendix B contains a command file listing to perform this transfer (DLRSX.CMD, DYRSX.CMD).

When the files are transferred onto the winchester, install VMR and IND, then perform the final VMR phase. Appendix B contains command files to setup and perform the VMR (DLSYSV.CMD, DYSYSV.CMD).

After the VMR is complete, the system image can be booted and run.

3.8.11 RSX-11M Double-Sided Floppy Support

RSX-11M, as distributed, has almost all the support needed for RX03 floppy systems. There are, however, some glitches which are detailed below and in command file RSX11M.DOC.

1. BUG in extended memory cross field transfers. This is documented in June 1980 SOFTWARE DISPATCH. The correction is also contained in the file RSX11M.DOC on the DSD distribution disk.

2. BUG in track/sector calculation algorithm in [11,10] DYDRV.MAC and [12,10] SAVSPC.MAC used in [1,20] or [1,24] SAV.OLB. This causes a hard error return from the handler whenever block numbers (double-density, double-sided) greater than 1664 are accessed. A fix for the handler is included in file RSX11M.DOC on the DSD distribution disk. If the SAV.OLB is not rebuilt prior to SYSGEN, then any tasks the SAV accesses when saving the RSX-11M system image must reside below block 1663. If not, it will be impossible to make a floppy bootable RSX-11M system image.

3.8.12 RSX-11M, DSD 880x/8 in Extended Mode

The DSD 880 operates in either RL01 compatible mode or extended mode which is a subset of an RL02. The RSX-11 monitor must be informed of this difference in device sizes in RL02/extended mode. The device length must be specified just before initializing (INI) the winchester. Once the disk is initialized, all length specific attributes are contained in the file structure, and the device length in the monitor device table is no longer used. The device length value will be reset to the RL01 value each time the system is booted, however, once the disk has been initialized it does not matter that the monitor device table value has been reset. The following paragraphs explain this process in more detail.

The monitor device table must first be initialized with the correct device size. The size of each logical unit is contained in the fixed offset words U.CW2 (high 8 bits) and U.CW3 (low 16 bits) in the unit control block (UCB) of the monitor device tables (SYSTB.MAC). Thus, if DL0: is the DSD 880 winchester, look up .DL0 in RSX11M.MAP, which is the start of the UCB. In a typical multi-user SYSGEN .DL0 is at 42304 (octal). To this add the offset U.CW3 (in this case it is 14) which gives the address in memory for this entry. Use the privileged MCR OPEN command to change this location to 35600 (octal). This location should contain 24000 if the 880 was in normal mode (RL01) when last booted, or 50000 if the 880 was in extended mode (RL02). This location cannot be patched permanently by ZAP since all variable-sized units are dynamically sized and lengths reloaded at boot time. (This offset location for all RL units is modified at RSX-11 boot time by SAV.TSK to contain either 24000 or 50000 by the routine DLSET called by SIZDSK from $TSTDV found in [12,10] SAVSUB.MAC).

After the device length in U.CS3 is set to 35600, the device should be initialized by using BAD and INI (See RSX-11M Utilities Manual).

```
>ALL DL0:            !Allocate this unit for single user.

>BAD DL0:/L1         !Do a bad block scan of entire unit.
                     !Note that the last 20.blocks will be
                     !reserved as bad to protect the "Manufacturer's
                     !Last Track Bad Block Map."

>INI DL0:DLRSX       !Init the unit to a Files-11 file
                     !structure.
```

At this time all length specific quanitites for the 880 winchester are fixed by the file structure in the available block and BAD block-bit map files (BITMAP.SYS and BADBLK.SYS). The only further uses of the U.CW3 value are when another BAD or INI function is performed on that particular logical unit or when a DEV query is requested. If the value at U.CW3 is 24000, DEV returns RL01 as the device type; otherwise, for any other value, it returns RL02 as the device type. PIP/FR uses the BITMAP files to determine available space.

To put a bootable RSX-11M system on the 880 in extended mode, proceed by copying the desired files onto the winchester as in DLRSX.CMD and perform the final VMR phase of the SYSGEN as in DLSYSV.CMD on the DSD 880 distribution disk.

Note that to use the same SYSGEN for DY, DL and any other genned device, all file structured handlers must be loadable and the desired system device must be the first file structured handler loaded under VMR. When VMR is complete

>BOOT DLO:RSX11M<CR>

~system will come up part way and type out any offline devices and print the ">" prompt.

>SAV/WB<CR>

~system will finish the SAV process and come up all the way.

3.8.13 Quick Copies of RSX Using RT-11

Users who are replacing an RL01 with a DSD 880 and running RSX are faced with the problem of getting the RL01 copied onto the DSD 880. This is easily accomplished if both the DSD 880 and RL01 can be connected at the same time. However, this may require a new RSX SYSGEN to support the two RL controllers. If the user has RT-11 V4 available, it can be utilized to perform the volume copy of an RL01 RSX-11M disk onto the DSD 880. This eliminates the necessity of a custom SYSGEN to support two RL01 controllers.

The procedure requires disabling the RT-11 bad block remapping support, since this support uses Block 1 for its remap information. RSX-11 RSX-11 uses Block 1 as its home block, and it contains information that is very misleading to an unmodified RL01 handler. Thus, starting with a distribution RT-11 V4 system on floppy, set up two new modified handlers (say DF.SYS and DZ.SYS) for standard and alternate addresses.

```
.COPY/SYS DL.SYS DR.SYS<CR>
.COPY/SYS DL.SYS DZ.SYS<CR>
.R PATCH <CR>
```

File name -

```
*DF.SYS<CR>
 2500/ 177777<LF>
*2502/ 177777 0<CR>          ;Patch 2nd word of -1 value
                             ;to zero. This indicates
                             ;that the bad block map is
                             ;loaded into the handler and
                             ;contains no bad blocks
                             ;on unit 0.

*2554/ 177777<LF>
 2556/ 177777 0<CR>          ;Do same for DF1 unit.

*F
```

File name -

```
*DZ.SYS<CR>                  ;Set DZ handler for DEVICE
                             ;174410 and interrupt vector 150.
 1000/ 160 150<DR>           ;Patch to new interrupt
                             ;vector value.

*174400;S                    ;Find all occurences of
                             ;RLCS value.

  176/   174400 A=?? X=93X
 1020/   174400
 3104/   174400
 3452/   174400

 *176/   174400 174410<CR>
*1020/   174400 174410<CR>
*3104/   174400 174410<CR>
*3452/   174400 174410<CR>
*E
```

!Make sure that RT-11 V4 is booted on the floppy.

```
.INS DF:          !Install the new handlers so that RT
.INS DZ:          !knows about them.
```

Put the RSX disk to be copied in the standard address controller unit 0 to be accessed by the DF handler.

Copy the RL01 disk (device DF0:) onto the DSD 880 (device DZ0:) via:

```
.COPY/DEV  DF0:  DZ0:NOQ
```

Note that the DF and DZ handlers should only be used when bad block mapping is to be disabled.

4.0 OPERATION

4.1 General Information

This chapter provides information on the operation of the DSD 880x/8 Data Storage System. Included are operating parameters, mode/class selection, system initialization, bootstrapping, diskette formatting, and backup operation.

4.2 Power On Self-Tests

When power is applied to the DSD 880, the controller automatically performs four self-tests:

- ALU Test
- CRC Logic Test
- Internal RAM Memory Test
- PLL Test

If any of these tests fail, an error code will be displayed on the HyperDiagnostics panel identifying the failure. If the tests are successfully passed, both the floppy disk and winchester drives are homed. The winchester drive will be write protected for two minutes following a power on to allow thermal stabilization. The winchester ready light will flash during this period. It is possible to read or boot from the winchester drive during this period.

4.3 Mode and Class Selection

DSD 880x/8 mode and class of operation selection is made on the HyperDiagnostics panel. To gain access to the HyperDiagnostics panel, remove the front bezel by grasping the bezel on each side and pulling forward. Figure 4-1 shows the HyperDiagnostics control panel switches and indicators, and their location and function. Table 4-1 provides a summary of the indicators on the DSD 880x/8 HyperDiagnostics panel and their purpose. Table 4-2 provides a summary of the mode and class settings available on the DSD 880.

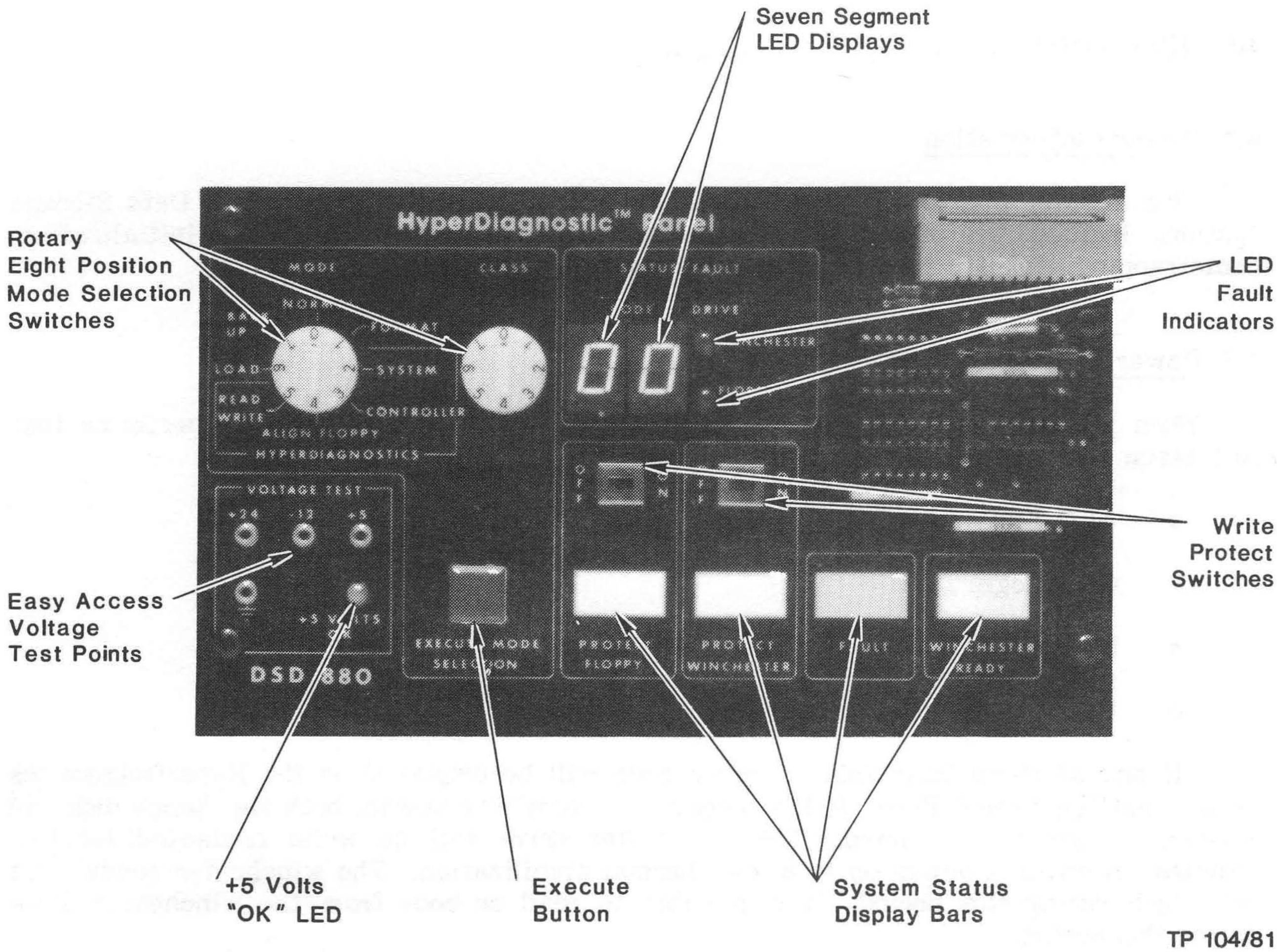


Figure 4-1. DSD 880x/8 HyperDiagnostic Panel

Table 4-1. DSD 880x/8 Indicators

Indicator	Purpose
Floppy Activity LED: (Located on the floppy disk drive front bezel)	This indicator illuminates whenever the head of the floppy disk drive is loaded. If the drive has a door lock mechanism, the door will be locked when the head is loaded.
Winchester Drive Ready LED: (Visible without removal of front bezel)	This indicator has several modes of operation. a. The indicator will flash for approximately two minutes after power is applied to the DSD 880x/8. During this time, the winchester drive will be write protected. This time is required to allow the media and drive to thermally stabilize.

Table 4-1. DSD 880x/8 Indicators (Cont)

Indicator	Purpose
	b. Approximately two minutes after power is applied to the unit the indicator will stop flashing and remain illuminated, if the bad track map has been read successfully, indicating that the drive is fully operational.
	c. Each time the winchester drive is accessed via a read or write command the indicator will flicker, indicating that the drive is busy (not ready).
	d. If a drive fault occurs which causes the winchester disk drive to be inoperative, the indicator will be extinguished until the fault is cleared.
Floppy Write Protect LED: (Visible without removal of front bezel)	This indicator is illuminated whenever the floppy disk drive is write protected, either by the write protect switch on the front panel, or by the presence of a write protected floppy disk.
Winchester Drive Write Protect LED: (Visible without removal of front bezel)	This indicator is illuminated whenever the winchester disk drive is write protected by the write protect switch on the front panel.
Fault LED: (Visible without removal of front bezel)	This indicator flashes for approximately one minute after an error occurs during the execution of a command. After approximately one minute, the indicator will cease flashing and illuminate steadily until the current error is cleared. If another error occurs before the original error is cleared, the indicator light will again flash for approximately one minute from the occurrence of that error. The indicator will be immediately extinguished by a bus initialize from the host processor.
Floppy Error LED:	This indicator flashes whenever the error being displayed by the seven segment displays occurred on the floppy disk drive.
Winchester Error LED:	This indicator flashes whenever the error being displayed by the seven segment displays occurred on the winchester disk drive.

Table 4-1. DSD 880x/8 Indicators (Cont)

Indicator	Purpose
Seven Segment Error Display (2):	These indicators flash the definitive error code for the most recent error. The error is flashed approximately one minute after the error is cleared. A bus initialize from the host processor will immediately clear all errors. When there are no errors present, the code 00 will be displayed. NOTE During HyperDiagnostics tests, the selected test code will be displayed will be displayed until either the test completes without error (00 displayed), or an error occurs (definitive error code flashing). If errors exist on both winchester and floppy drives, the seven segment error displays will indicate the most recent error, and the appropriate floppy or winchester error LED will flash. The other (earlier) error LED will be on continuously. If the "most recent" error is cleared, the seven segment error displays will begin to flash the error for the other drive.
5 Volts OK LED:	This indicator will be illuminated when the main 5 volt power supply of the DSD 880x/8 is operating within specification.

Table 4-2. DSD 880x/8 Mode and Class Options

Switch Settings: Mode	Switch Settings: Class	Descriptions
0	0	In this mode, called the normal mode, the winchester drive emulates a single RL01. User has access to 5.3 Mbytes of storage on the winchester drive. The RL01 emulation is totally software compatible with DEC.
0	1	In this mode, called the extended mode, the winchester drive emulates a diminished RL02. User has access to 7.8 Mbytes of storage on the winchester drive. For required modification to DEC software, see Section 5 of this manual. NOTE In both the above modes, the flexible disk drive emulates an RX02. Double-sided operation may be activated by incorporating the procedures outlined in Section 5 of this manual.
0	2	This mode is called the direct access mode and allows the user to access each physical track on the winchester drive. This mode is used for maintenance purposes.
1	0	FORMAT DOUBLE-DENSITY - formats entire floppy disk in DEC double-density.
1	1	FORMAT SINGLE-DENSITY - formats entire floppy disk in DEC/IBM single-density.
1	2	SET MEDIA DOUBLE-DENSITY - sets the floppy media to double-density.
1	3	SET MEDIA SINGLE-DENSITY - sets the floppy media to single-density.
1	4	SET MEDIA DOUBLE-DENSITY AND SCAN - writes all data feeds in DEC double-density format, then scans the disk looking for errors.
1	5	SET MEDIA SINGLE-DENSITY AND SCAN - writes all data fields in DEC/IBM single-density format, then scans the disk looking for errors.

Table 4-2. DSD 880x/8 Mode and Class Options (Cont)

Switch Settings		Descriptions
Mode	Class	
2	0	FLOPPY DISK EXERCISER WITH WRITE FORMAT - runs the following sequence of HyperDiagnostics tests on the floppy drive only: a. Hardware Self-Tests b. Single-Density Write Format c. Sequential Scan All Sectors d. Butterfly Read Headers e. Sequential Write/Read All Sectors f. Set Media Double-Density g. Sequential Scan All Sectors h. Butterfly Read Headers i. Sequential Write/Read All Sectors j. Set Media Single-Density
2	1	FLOPPY DISK EXERCISER WITHOUT WRITE FORMAT - runs the same sequence of tests as the floppy disk exerciser described previously with the exception of the single-density write format.
2	2	FIXED DISK EXERCISER - runs the following sequence of HyperDiagnostics tests on the fixed disk drive only: a. Hardware Self-Tests b. Sequential Scan All Sectors c. Butterfly Read Headers d. Sequential Write/Read All Sectors
2	3	GENERAL EXERCISER WITH FLOPPY DISK WRITE FORMAT - runs the floppy disk general exerciser then runs the fixed disk exerciser tests.
2	4	SINGLE PASS GENERAL EXERCISER WITH FLOPPY WRITE FORMAT - runs a single pass of the floppy and fixed disk exercisers.
2	5	SINGLE PASS GENERAL EXERCISER WITHOUT FLOPPY WRITE FORMAT - runs a single pass of the floppy and fixed disk exercisers without formatting the floppy disk.
2	6	GENERAL EXERCISER WITHOUT FLOPPY WRITE FORMAT AND FIXED READ/WRITE TESTS - runs the floppy disk general exerciser without formatting the floppy disk, then runs the fixed disk exerciser without executing the sequential write/read tests.

Table 4-2. DSD 880x/8 Mode and Class Options (Cont)

Switch Settings: Mode	Class	Descriptions
2	7	FIXED DISK EXERCISER WRITE ENABLE - permits sequential write operations on the winchester disk. (For tests 2, 3, 4, and 5)
3	0	CONTROLLER SWITCH AND INDICATOR TEST - tests the various controller switches and indicators.
3	1	GENERAL CONTROLLER HARDWARE TEST - runs the following controller hardware diagnostics: a. ALU test b. RAM test c. CRC logic test d. PLL test
3	2	ALU LOGIC TEST - tests the operation of the arithmetic - logic unit.
3	3	MEMORY TEST - tests the operation of the RAM buffer memory.
3	4	CRC LOGIC TEST - tests the operation of the CRC logic.
3	5	PLL TEST - tests the operation of the phase-locked-loop.
3	6	MICROCODE VERSION - displays the microcode version number.
3	7	Not Defined.
		NOTE The following floppy disk drive alignment tests can be run without media in the floppy drive.
4	0	FLOPPY DISK TRACK 00 DETECTOR ADJUSTMENT - loads floppy head and repeatedly seeks between track 00 and 01 every 100 ms.
4	1	FLOPPY DISK SEEK TRACK 01 AND LOAD HEAD - seeks floppy head to track 01 and loads it.
4	2	FLOPPY DISK SEEK TRACK 02 AND LOAD HEAD - seeks floppy head to track 02 and loads it.

Table 4-2. DSD 880x/8 Mode and Class Options (Cont)

Switch Settings Mode	Class	Descriptions
4	3	FLOPPY DISK SEEK TRACK 38 AND LOAD HEAD - seeks floppy head to track 38 and loads it.
4	4	FLOPPY DISK SEEK TRACK 76 AND LOAD HEAD - seeks floppy head to track 76 and loads it.
4	5	FLOPPY DISK HEAD LOAD TIMING ADJUSTMENT - seeks floppy head to track 00 then alternately loads and unloads head every 100 ms.
5	0	SINGLE PASS SEQUENTIAL SCAN FLOPPY DISK - scans entire floppy disk for CRC errors and valid disk headers only once.
5	1	BUTTERFLY SEEK TEST FLOPPY DISK DRIVE - steps head of floppy disk drive using butterfly pattern then seeks tack 00. Note that this test can be run without media in the floppy drive.
5	2	BUTTERFLY READ HEADERS ON FLOPPY DISK - steps head of floppy disk driving using butterfly pattern, checking for correct disk headers.
5	3	SEQUENTIAL WRITE/READ FLOPPY DISK - sequentially writes then reads the entire floppy disk checking for data or header errors.
5	4	SEQUENTIAL SCAN FIXED DISK - scans entire fixed disk for CRC errors and valid disk headers.
5	5	BUTTERFLY READ HEADERS ON FIXED DISK - steps head of fixed disk drive using butterfly pattern, checking for correct disk headers.
5	6	SEQUENTIAL WRITE/READ FIXED DISK - sequentially writes then reads the entire winchester disk.
5	7	FIXED DISK WRITE ENABLE - permits sequential write operation on the winchester disk. (For test 6)
6	0	RELOAD WINCHESTER FROM BACKUP FLOPPY DISKS - copies the data from valid backup floppy disks onto the winchester disk.

Table 4-2. DSD 880x/8 Mode and Class Options (Cont)

Switch Settings		Descriptions
Mode	Class	
6	1	RELOAD AND VERIFY WINCHESTER FROM BACKUP FLOPPY DISKS - copies data from valid backup floppy disks onto the winchester disk. Verifies each backup disk was copied correctly.
7	0	BACKUP WINCHESTER ONTO FLOPPY DISKS - copies the data on the winchester disk onto backup floppy disks.
7	1	BACKUP WITH VERIFY WINCHESTER ONTO FLOPPY DISKS - copies data on the winchester disk onto backup floppy disks. Verifies data was written correctly onto each floppy disk.
7	2	BACKUP WINCHESTER ONTO FLOPPY DISKS WITH DOUBLE-DENSITY FORMAT - formats the floppy disk in double-density, then copies the data on the winchester disk onto the floppy disk.
7	3	BACKUP WINCHESTER ONTO FLOPPY DISKS WITH DOUBLE-DENSITY FORMAT AND VERIFY - formats the floppy disks in double-density, copies the winchester data onto the floppy disks, then verifies that the data was written correctly onto each floppy disk.
7	4	BACKUP WITH SINGLE-DENSITY FORMAT - formats floppy disks in single-density Copies winchester disk data onto the floppy disk.
7	5	BACKUP WITH SINGLE-DENSITY FORMAT AND VERIFY - formats the floppy disks in single-density, copies the winchester data onto the floppy disks, then verifies that the data was written correctly onto each floppy disk.
7	6	Not Defined
7	7	FLOPPY-TYPE FLAG SET ON BAD TRACK MAP-maintenance only. See below: 1. Enter 77 and press EXECUTE. 2. System displays 47 asking for confirmation. 3. Enter 22 to confirm, press EXECUTE. 4. Enter: Mode / Class 0 / 0 = No floppy 0 / 1 = SA800 0 / 2 = SA850 0 / 3 = Normal backup/restore 0 / 4 = 8841-02 restore (see note, para. 7.4.9) and press EXECUTE pushbutton.

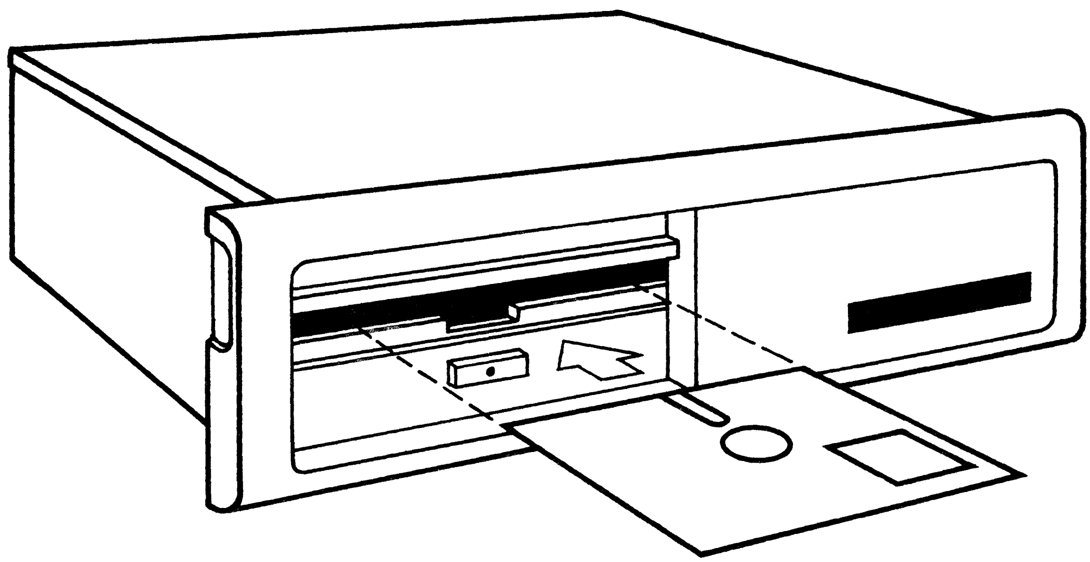
4.4 Normal Operation

Prior to placing the DSD 880x/8 into operation, insert a diskette into the floppy disk drive. Ensure the diskette is a soft sectored, eight-inch diskette (see Figure 4-2 for diskette orientation for insertion). Close the drive door. Select mode of operation that matches the operating system parameters (refer to Table 4-2 for DSD 880x/8 mode and class option).

CAUTION

If the DSD 880x/8 is not in the normal mode at the time a bus initialize is generated by the host processor, the DSD 880x/8 controller will terminate any HyperDiagnostic test which may be occurring, force the mode and class to 0, and initialize (home) the floppy and winchester disk drives.

If the mode is 0 (normal) at the time of a bus initialize, the DSD 880x/8 controller will determine if the class is a valid normal class (0-2). If the class is invalid, the controller will force the class to be 0.



TP 105/81

Figure 4-2. Proper Orientation of Diskette for Insertion

4.4.1 System Bootstrapping

A hardware bootstrap is built into the DSD 880x/8 LSI-11 and PDP-11 interfaces, eliminating the need to buy the expensive DEC bootstrap options (BDV11 bootstrap card or MXV11 multifunction card for LSI-11 or LSI 11/23 systems, MR11EA bootstrap for PDP-11 systems).

The 880x/8 system can boot using either the winchester drive or the flexible disk drive. For LSI-11 systems, the 880x/8 system can be bootstrapped in either power up mode.

In power up mode 1, the LSI-11 processor enters console ODT immediately on power up. The user may select the bootstrap device by entering the appropriate starting address at the console. For example, if the standard bootstrap base address is used, bootstrapping on the winchester may be initiated by entering 773000G. Entering 7730020G initiates bootstrap on the floppy disk.

In power up mode 2, the LSI-11 program counter is automatically set at 173000 on power up. Hence, the sytem automatically attemps to boot on the winchester. If the winchester is not bootable, the system loops at 773210 to 773274. The user may force bootstrapping on the floppy disk by entering the appropriate address at the console. For the LSI-11/23, the mode 2 power up address is user programmable. The DSD 880x/8 hardware bootstrap automatically performs certain operations and conducts tests to verify correct operation of the interface, the controller and the processor memory. The Shugart SA1000 drive requires a two minute delay for thermal stabilization after power on before writing should be done. This is provided for on the DSD 800x/8 by returning a drive write protected status error to the computer, if a write is attempted before then. If a user wishes to auto boot their system using power up mode 2 on the LSI-11 processors, it is useful to be able to wait for the drive to become write enabled. The program DL WAIT will do this, if run by a startup command file, and then continue executing the remainder of the startup file when the disk is write enabled. A version of this program is currently provided for RT-11.

The DSD880x/8 bootstrap program consists of three or four procedures, depending on the device to be booted:

Determines the selected bootstrap device (RL or RX).

Sizes memory, then checks memory for failing data or address bits.

RL BOOT - Reads block 0 from RL unit 0, then starts at location 0.

RX BOOT - Performs fill/empty test on DSD880x/8 RX02 device which verifies operation of available DMA address lines and RX02 sector buffer.

RX BOOT - Reads block 0 from RX unit 0, then starts at location 0.

Table 4-3 provides a listing of the DSD 880x/8 interface bootstrap program, starting address, and device addresses.

After completion of a successful system bootstrap, the DSD 880x/8 will have completed an initialization sequence, assumed the mode of operation selected, and be ready to complete data storage and retrieval tasks as directed by the host computer.

Table 4-3. DSD 880x/8 Interface Bootstrap Program Starting Addresses and Device Addresses

Bootstrap Offset	Standard Bootstrap Address		Bootstrap Device	Device Address
	PDP-11	LSI-11		
+ 0	771000	773000	Winchester	774400
+ 10	771010	773010	Winchester	774440
+ 20	771020	773020	Floppy	777170
+ 30	771030	773030	Floppy	777150
+ 36	771036	773036	— User Defined —	

4.5 Bootstrap Failure Procedure

At each stage in the bootstrap there are locations where failures will cause the bootstrap routine either to halt, or loop waiting for an action to occur.

Processor Halts - The processor run indicator will be extinguished on PDP-11 and LSI-11 front panels.

On processors with ODT and a console terminal, there will be an ODT prompt on the console.

Program Loops - The processor run indicator will be illuminated on PDP-11 and LSI-11 front panels.

Program loops can be halted by typing break on the console terminal, if ODT is available, and halt on break is enabled. On PDP-11s without ODT enter control halt from the front panel.

4.5.1 Troubleshooting Bootstrap Failures

If the program is stuck in a loop (not halted though not booted after approximately 30 seconds), manually halt the program via the console or front panel. Note the address at which the program halts and any error reported by the DSD 880x/8 front panel. Tables 4-4 and 4-5 provide a listing of bootstrap halt locations, the possible cause of the halt, and procedures for solving the problem. These tables are applicable to 880x/8 systems using the 8832 LSI-11 interface card. Table 4-6 and 4-7 provide the same information applicable to systems using the 8830 PDP-11 interface card, and 8836 LSI-11 interface card. If you are unable to manually halt the program, or a PDP-11 bus error occurs:

- Verify the DSD 880x/8 interface jumper configuration.
- Verify the backplane jumpers for DMA and interrupt grants.
- Verify correct installation of the DSD 880x/8 interface in the backplane.

Table 4-4. Program Halt Locations (8832 Interface)
(Halt location offset from bootstrap base address)

Location		Description
XXX002	Fault:	Bootstrap does not respond
	Possible cause:	DSD 880x/8 bootstrap not enabled Bootstrap starting address incorrectly configured Defective DSD 880x/8 interface Memory address range extends into bootstrap area
	Troubleshooting:	Verify configuration of DSD 880x/8 interface jumpers Verify ability to access bootstrap starting address without error (should contain 12737)
XXX244	Fault:	RL device reported error following read sector operation
	Possible cause:	Unable to read sector from RL Defective DSD 880x/8 controller Defective winchester disk drive
	Troubleshooting:	Verify integrity of DSD 880x/8 winchester bad track map Replace DSD 880x/8 controller card assembly Service winchester disk drive assembly
XXX276	Fault:	Processor memory error (at location R4 , read R0 expected R4)
	Possible cause:	Defective host processor memory Defective host processor Refresh for dynamic memory board defective
	Troubleshooting:	Verify ability to access failing memory location Verify dynamic memory refresh (deposit 125252, wait two minutes, verify contents unchanged) Use DEC memory diagnostics to verify failure Replace failing memory module
XXX324	Fault:	Processor memory error (at location -2 R4 , read R0 expected 0)
	Possible cause:	Defective host processor memory Defective host processor Refresh for dynamic memory board defective
	Troubleshooting:	Verify ability to access failing memory location Verify dynamic memory refresh (deposit 125252, wait two mintures, verify contents unchanged) Use DEC memory diagnostics to verify failure Replace failing memory module

Table 4-4. Program Halt Locations (8832 Interface) (Cont)
(Halt location offset from bootstrap base address)

Location		Description
XXX372	Fault:	Processor memory error (if R5 - boot base address + 112, R6 = 5002)
		Fill/empty error (if R5 = boot base address + 522, R6 = 5000)
	Possible cause:	KD11-F processor is being used to refresh external RAM Defective host processor memory Defective DSD 880x/8 controller if fill/empty error
	Troubleshooting:	If KD11-F uses REV-11 or on board memory refresh, use DEC memory diagnostics to verify failure Replace failing memory module Replace DSD 880x/8 controller PCB assembly
XXX436	Fault:	Error flag in RXCS set following bus initialize
	Possible cause:	Interface cable not properly installed AC power to DSD 880x/8 chassis not turned on Unable to read sector from floppy disk DSD 880x/8 controller failed initialize test sequence Defective DSD 880x/8 controller
	Troubleshooting:	Verify installation of DSD 880x/8 interface cable Verify ac power to DSD 880x/8 chassis Verify controller passes initialize test sequence Verify that floppy drive is properly configured for operating voltage and frequency Replace floppy disk media Replace DSD 880x/8 controller PCB assembly
XXX452	Fault:	RXCS does not latch appropriate bits (5460)
	Possible cause:	Interface defective
	Troubleshooting:	Replace interface PCB assembly
XXX474	Fault:	RXDB does not latch appropriate bits (1420, 173767)
	Possible cause:	Interface defective
	Troubleshooting:	Replace interface PCB assembly

Table 4-4. Program Halt Locations (8832 Interface) (Cont)
(Halt location offset from bootstrap base address)

Location		Description
XXX614	Fault:	RX02 device reported error following read sector operation (Definitive error code in R6)
	Possible cause:	Disk not inserted in floppy drive Floppy disk door open Double-sided floppy disk in single-sided drive Defective floppy disk media Incorrectly configured floppy disk drive (ac voltage and frequency) Defective floppy disk drive Defective DSD 880x/8 controller
	Troubleshooting:	Verify installation of floppy disk media in drive Replace floppy disk media Verify drive configuration Service floppy disk drive Replace DSD 880x/8 controller PCB assembly

Table 4-5. Program Loops (8832 Interface)
(Program loop addresses offset from bootstrap base address)

Location		Description
XXX152-156	Fault:	RL controller not ready following bus initialize
	Possible cause:	Interface cable not properly installed DSD 880x/8 controller failed initialization test sequence AC power to DSD 880x/8 chassis not turned on
	Troubleshooting:	Verify installation of DSD 880x/8 interface cable Verify ac power to DSD 880x/8 chassis Verify controller passes intialization test sequence Replace DSD 880x/8 controller PCB assembly
XXX154	Fault:	Interface does not respond to RLCS address
	Possible cause:	Incorrectly configured RL device address jumpers Incorrectly specified bootstrap starting address Defective interface
	Troubleshooting:	Verify interface jumper configuration Verify interface response at expected device addresses Replace interface PCB assembly

Table 4-5. Program Loops (8832 Interface) (Cont)
(Program loop addresses offset from bootstrap base address)

Location		Description
XXX172-174	Fault:	RL controller not ready following get status command
	Possible cause:	Defective DSD 880x/8 controller Defective interface
	Troubleshooting:	Replace DSD 880x/8 controller PCB assembly Replace interface PBC assembly
XXX210-212	Fault:	RL controller not ready following seek command
	Possible cause:	Defective DSD 880x/8 controller Defective interface
	Troubleshooting:	Replace DSD 880x/8 controller PCB assembly Replace interface PBC assembly
XXX232-234	Fault:	RL controller not ready following read sector command
	Possible cause:	Defective DSD 880x/8 controller Defective interface
	Troubleshooting:	Replace DSD 880x/8 controller PCB assembly Replace interface PBC assembly
XXX426	Fault:	Interface does not respond to RXCS address
	Possible cause:	Incorrectly configured RX device address jumpers Incorrectly specified bootstrap starting address Defective interface
	Troubleshooting:	Verify interface jumper configuration Verify interface response at expected device address Replace interface PCB assembly
XXX506-510 XXX514-516	Fault:	Transfer request error during RX02 fill buffer test
	Possible cause:	Defective interface Defective DSD 880x/8 controller
	Troubleshooting:	Replace interface PCB assembly Replace DSD 880x/8 controller PCB assembly

Table 4-5. Program Loops (8832 Interface) (Cont)
(Program loop addresses offset from bootstrap base address)

Location		Description
XXX536-540 XXX544-546	Fault:	Transfer request error during RX02 empty buffer test
	Possible cause:	Defective interface Defective DSD 880x/8 controller
	Troubleshooting:	Replace interface PCB assembly Replace DSD 880x/8 controller PCB assembly
XXX646-650 XXX654-656	Fault:	Transfer request error during RX02 read sector command
	Possible cause:	Defective interface Defective DSD 880x/8 controller
	Troubleshooting:	Replace interface PCB assembly Replace DSD 880x/8 controller PCB assembly
XXX724-726 XXX734-736	Fault:	Transfer request error during RX02 empty buffer command
	Possible cause:	Defective interface Defective DSD 880x/8 controller
	Troubleshooting:	Replace interface PCB assembly Replace DSD 880x/8 controller PCB assembly
XXX770-774	Fault:	Done flag error during RX02 command
	Possible cause:	Defective interface Defective DSD 880x/8 controller
	Troubleshooting:	Replace interface PCB assembly Replace DSD 880x/8 controller PCB assembly

Table 4-6. Program Halt Locations (8830 and 8836 Interfaces)
(Halt location offset from bootstrap base address)

Location		Description
XXX002	Fault:	Bootstrap does not respond
	Possible cause:	DSD 880x/8 bootstrap not enabled Bootstrap starting address incorrectly configured Defective DSD 880x/8 interface Memory address range extends into bootstrap area Processor halt switch is in HALT position

Table 4-6. Program Halt Locations (8830 and 8836 Interfaces) (Cont)
(Halt location offset from bootstrap base address)

Location		Description
XXX002 (Cont)	Troubleshooting:	Verify configuration of DSD 880x/8 interface jumpers Verify ability to access bootstrap starting address without error (should contain 12737) Verify halt switch is in RUN position
XXX246	Fault:	RL device reported error following read sector operation
	Possible cause:	Unable to read sector from RL Defective DSD 880x/8 controller Defective winchester disk drive Invalid RL logical unit specified as boot device
	Troubleshooting:	Verify integrity of DSD 880x/8 winchester bad track map Replace DSD 880x/8 controller PCB assembly Replace winchester disk drive assembly Reboot using correct logical unit
XXX300	Fault:	Processor memory error at location R4 (contents of memory did not equal value of R4)
	Possible cause:	Defective host processor memory Defective host processor Refresh for dynamic memory board defective
	Troubleshooting:	Verify ability to access failing memory location Verify dynamic memory refresh (deposit 125252, wait two minutes, verify contents unchanged) Use DEC memory diagnostics to verify failure Replace failing memory module
XXX324	Fault:	Processor memory error (at location -2, R4, Read R0 expected 0)
	Possible cause:	Defective host processor memory Defective host processor Refresh for dynamic memory board defective
	Troubleshooting:	Verify ability to access failing memory location Verify dynamic memory refresh (deposit 125252, wait two minutes, verify contents unchanged) Use DEC memory diagnostics to verify failure Replace failing memory module
XXX374	Fault:	Processor memory error (if R5 = boot base address + 116, R6 = 5002) Fill/empty error (if R5 = boot base address + 622, R6 = 5000)

Table 4-6. Program Halt Locations (8830 and 8836 Interfaces) (Cont)
(Halt location offset from bootstrap base address)

Location		Description
XXX374 (Cont)	Possible cause:	KD11-F processor is being used to refresh external RAM Defective host processor memory Defective DSD 880x/8 controller if, fill/empty error
	Troubleshooting:	If KD11-F uses REV-11 or on-board memory refresh Use DEC memory diagnostics to verify failure Replace failing memory module Replace DSD 880x/8 controller PCB assembly
XXX476	Fault:	Error flag in RXCS set following initialize
	Possible cause:	Interface cable not properly installed AC power to DSD 880x/8 chassis not turned on Unable to read sector from floppy disk DSD 880x/8 controller failed initialize test sequence Defective DSD 880x/8 controller
	Troubleshooting:	Verify installation of DSD 880x/8 interface cable Verify ac power to DSD 880x/8 chassis Verify controller passes initialize test sequence Verify that floppy drive is properly configured for operating voltage and frequency Replace floppy disk media Replace DSD 880x/8 controller PCB assembly
XXX524	Fault:	RXCS does not latch appropriate bits (5460)
	Possible cause:	Interface defective
	Troubleshooting:	Replace interface PCB assembly
	Fault:	RX02 device reported error following read sector operation (definitive error code at memory location 0)
	Possible cause:	Disk not inserted in floppy drive Floppy disk door open Double-sided floppy disk in single-sided drive Defective floppy disk media Incorrectly configured floppy disk drive (ac voltage and frequency) Defective floppy disk drive Defective DSD 880x/8 controller Incorrect logical unit specified as boot address

Table 4-6. Program Halt Locations (8830 and 8836 Interfaces) (Cont)
(Halt location offset from bootstrap base address)

Location		Description
XXX424 (Cont)	Troubleshooting:	Verify installation of floppy disk media in drive Replace floppy disk media Verify drive configuration Replace floppy disk drive Replace DSD 880x/8 controller PCB assembly Reboot using RX logical unit 0

Table 4-7. Program Loops (8830 and 8836 Interfaces)
(Program loop addresses offset from bootstrap base address)

Location		Description
XXX146-152	Fault:	RL controller not ready following bus initialize
	Possible cause:	Interface cable not properly installed DSD 880x/8 controller failed initialization test sequence AC power to DSD 880x/8 chassis not turned on
	Troubleshooting:	Verify installation of DSD 880x/8 interface cable Verify ac power to DSD 880x/8 chassis Verify controller passes intialization test sequence Replace DSD 880x/8 controller PCB assembly
XXX150	Fault:	Interface does not respond to RLCS address
	Possible cause:	Incorrectly configured RL device address jumpers Incorrectly specified bootstrap starting address Defective interface
	Troubleshooting:	Verify interface jumper configuration Verify interface response at expected device addresses Replace interface PCB assembly
XXX166-170	Fault:	RL controller not ready following get status command
	Possible cause:	Defective DSD 880x/8 controller Defective interface
	Troubleshooting:	Replace DSD 880x/8 controller PCB assembly Replace interface PBC assembly

Table 4-7. Program Loops (8830 and 8836 Interfaces)
(Program loop addresses offset from bootstrap base address)

Location		Description
XXX212-214	Fault:	RL controller not ready following seek command
	Possible cause:	Defective DSD 880x/8 controller Defective interface
	Troubleshooting:	Replace DSD 880x/8 controller PCB assembly Replace interface PBC assembly
XXX236-240	Fault:	RL controller not ready following read sector command
	Possible cause:	Defective DSD 880x/8 controller Defective interface
	Troubleshooting:	Replace DSD 880x/8 controller PCB assembly Replace interface PBC assembly
XXX416-420	Fault:	Transfer request error during read definitive error status command
	Possible Causes:	Defective interface Defective DSD 880x/8 controller
	Troubleshooting:	Replace interface PCB assembly Replace DDS 880x/8 controller PCB assembly
XXX470	Fault:	Interface does not respond to RXCS address
	Possible cause:	Incorrectly configured RX device address jumpers Incorrectly specified bootstrap starting address Defective interface
	Troubleshooting:	Verify interface jumper configuration Verify interface response at expected device address Replace interface PCB assembly
XXX534-536 XXX544-546	Fault:	Transfer request error during RX02 fill buffer test
	Possible cause:	Defective interface Defective DSD 880x/8 controller
	Troubleshooting:	Replace interface PCB assembly Replace DSD 880x/8 controller PCB assembly
XXX566-570 XXX576-600	Fault:	Transfer request error during RX02 empty buffer test
	Possible cause:	Defective interface Defective DSD 880x/8 controller

Table 4-7. Program Loops (8830 and 8836 Interfaces) (Cont)
(Program loop addresses offset from bootstrap base address)

Location		Description
XXX576-600 (Cont)	Troubleshooting:	Replace interface PCB assembly Replace DSD 880x/8 controller PCB assembly
XXX652-654 XXX660-662	Fault:	Transfer request error during RX02 read sector command
	Possible cause:	Defective interface Defective DSD 880x/8 controller
	Troubleshooting:	Replace interface PCB assembly Replace DSD 880x/8 controller PCB assembly
XXX730-732 XXX736-740	Fault:	Transfer request error during RX02 empty buffer command
	Possible cause:	Defective interface Defective DSD 880x/8 controller
	Troubleshooting:	Replace interface PCB assembly Replace DSD 880x/8 controller PCB assembly
XXX770-774	Fault:	Done flag or transfer request error during RX02 command
	Possible cause:	Defective interface Defective DSD 880x/8 controller
	Troubleshooting:	Replace interface PCB assembly Replace DSD 880x/8 controller PCB assembly

4.5.2 Bootstrap Program Flow Diagram

Figure 4-3 provides a flow diagram of the bootstrap program.

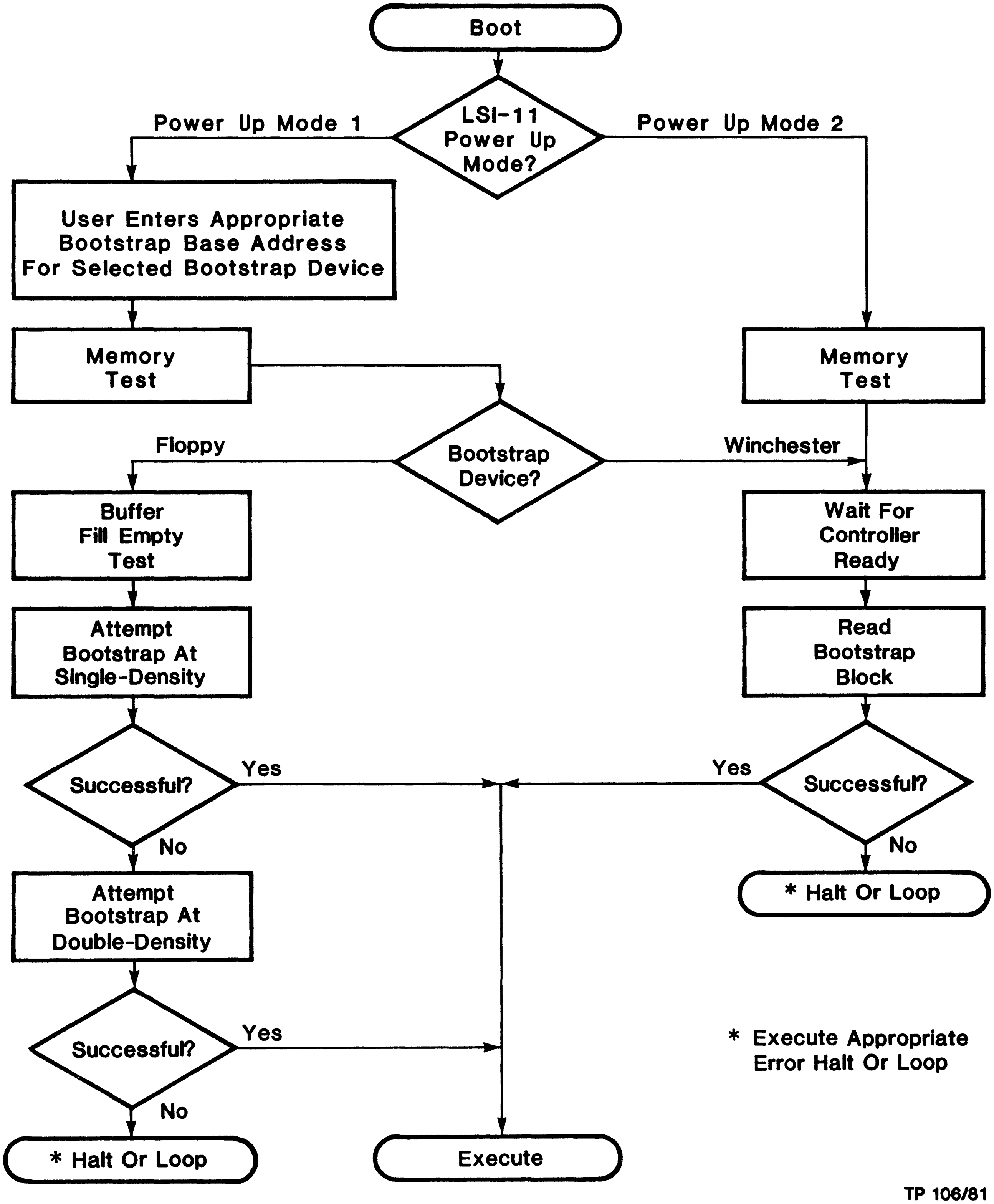


Figure 4-3. Bootstrap Flow Diagram

4.5.3 Bootstrap Program Listing (LSI-11 8832 Interface)

Table 4-8 provides the DSD 880x/8 bootstrap program listing for use with the 8832 LSI-11 interface printed circuit board.

Table 4-8. DSD 880x/8 Bootstrap Program Listing (LSI-11 8832 Interface)

```
DSD 880 BOOTSTRAP PROM  MACRO V04.00  23-OCT-80 11:15:00
TABLE OF CONTENTS

    1-   8      LSI-11 VERSION
    3-   1      RL COMPATIBLE BOOT
```

Table 4-8. DSD 880x/8 Bootstrap Program Listing (Cont) (LSI-11 8832 Interface)

```
DSD 880 BOOTSTRAP PROM  MACRO V04.00  23-OCT-80 11:15:00 PAGE 1

                  .TITLE  DSD 880 BOOTSTRAP PROM
                  ; BOT880.MAC     30-JUL-80-1
 .SBTTL LSI-11 VERSION
; BOOTSTRAP FOR DSD880 FLOPPY / WINCHESTER DISK CONTROLLER
; BOOTS EITHER SINGLE OR DOUBLE DENSITY FLOPPIES
; NOTE - THE DISKETTE BEING BOOTED MUST HAVE THE CORRECT MONITOR
;       FOR THE EXISTING HARDWARE CONFIGURATION.
;**  NOTE ON BOOTING WHILE REAL TIME CLOCK IS ENABLED. **
; THIS BOOT CAN BE STARTED WITH A RUNNING REAL TIME CLOCK IN 2 WAYS.
; 1) ENSURING THAT THE STACK IS POINTING TO NON-EXISTANT MEMORY THUS
;    FORCING A DOUBLE BUS ERROR ON ANY INTERRUPT AND TYPING
;    "773000G"  AND TYPING "P" IF HALTS OCCUR DUE TO ATTEMPTED INTERRUPTS.
; 2)  BY SETTING THE PSW AHEAD OF TIME TO DISABLE INTERRUPTS BY TYPING
;      "$S/ 340<CR>" AND "R7/ 773000<CR>" AND HITTING "P".

; THE BOOTSTRAP PROCEEDS IN 4 STEPS
; 1)    SELECT DEVICE   DETERMINES DEVICE TO BE BOOTED
; 2)    RAM TEST        CHECKS ALL AVAILABLE MEMORY FOR STUCK BITS
;                       ON BOTH DATA AND ADDRESS LINES. <0-30K>
;                       DOES BOTH DATA = ADDRESS AND PATTERN TESTS
;                       1) CLEARS MEMORY TO 0'S AND SIZES MEMORY
;                       2) LOADS MEMORY = ADDRESS AND CHECKS
;                       3) LOADS MEMORY = ADDRESS COMPLEMENT, CHECKS
;                       4) LOADS MEMORY WITH THE REPEATING PATTERN OF
;                          131617, 154707, 166343, 173161, 175470
; 3-WINCHESTER  READ IN BLOCK 0, START AT LOC 0.
; 3-DY  FILL-EMPTY      CHECKS DSD880 - PROCESSOR DATA PATH FOR
;                       SYNTAX AND DATA ERRORS.  ALSO INSURE'S ALL
;                       AVAILABLE ADDRESS LINES TOGGLE UNDER DMA.
;                       CHECKS FILL-EMPTY WITH BUFFERS AT 774,
;                       17700, 37676, 77704, 137700 IF MEMORY EXISTS.
; 4-DY  BOOTSTRAP       READS IN BLOCK 0 FROM DISKETTE IN CORRECT
;                       DENSITY AND STARTS AT LOC 0

; ERROR HALTS OR HANG UP LOOPS  (ADDRESSES RELATIVE TO BOOT BASE ADDR)
; 152-6 LOOP  RL CONTROLLER NOT READY
; 154   HANG    RL CONTROLLER NOT RESPONDING AT ADDRESS
; 172-174, 210-212, 232-234       RL TYPE CONTROLLER HUNG
; 276     HALT  MEMORY ERROR AT LOC -2(R4), READ R0, EXPECT ZERO
; 324     HALT  MEMORY ERROR AT -2(R4), READ R0, EXPECT 0
; 372     HALT  1) FILL-EMPTY ERROR IF R5=BOOT+522, SP=5000
;               2) MEMORY ERR IF R5=BOOT+112, SP=5002
; 426     LOOP   DY DEVICE ADDRESS SELECTED FOR BOOTING DOESN'T RESPOND
; 436     HALT  ERROR FLAG IN RXCS SET AFTER INIT
; 452     HALT  RXCS INTERFACE REGISTER STUCK BIT PROBLEM
; 474     HALT  RXDB INTERFACE LATCH PROBLEM, NOTE C(RXDB)
; 506-510, 514-516     TRANSFER REQUEST HANGUP  (FILL-EMPTY)
; 536-540, 544-546     TRANSFER REQUEST HANGUP  (FILL-EMPTY)

; 614   HALT    FLOPPY READ ERROR, PROCEED TO RETRY
;               C(SP) = DEFINITIVE ERROR STATUS
;               C(R5) = SECTOR # WITH PROBLEM
;       THIS USUALLY HAPPENS WITH A BAD DISKETTE AND MAY OCCUR
;       IF AN UN-BOOTABLE DISKETTE IS IN DRIVE 0.
; 646-650, 654-656     TRANSFER REQUEST HANGUP  (BOOTSTRAP)
; 724-726, 734-736     TRANSFER REQUEST HANGUP  (BOOTSTRAP)
; 770-774  LOOP DSD880 FLAG WAIT ROUTINE HANGUP
```

Table 4-8. DSD 880x/8 Bootstrap Program Listing (Cont) (LSI-11 8832 Interface)

```
DSD 880 BOOTSTRAP PROM  MACRO V04.00  23-OCT-80 11:15:00 PAGE 1-1
LSI-11 VERSION


        ; START ADDRESSES
        ; BOOT+0    (TYPICALLY 173000)  BOOTS RL DEVICE WITH RLCS AT 174400
        ; BOOT+10   (TYPICALLY 173010)  BOOTS RL DEVICE WITH RLCS AT 174410
        ; BOOT+20   (TYPICALLY 173020)  BOOTS DY DEVICE WITH RXCS AT 177170
        ; BOOT+30   (TYPICALLY 173030)  BOOTS DY DEVICE WITH RXCS AT 177150
        ; BOOT+36   (TYPICALLY 173036)  GENERAL DEVICE ENTRANCE - USER
        ;               SET'S  LOCATION 0 = DESIRED RLCS OR RXCS
        ;               NOTE: THE BIT OF VALUE 1000 MUST BE SET FOR RX BOOTING
        ;                       IF REAL TIME CLOCK MUST BE LEFT ON THEN SET
        ;                       $S/ 340<CR> AND R7/ 173040<CR> AND PROCEED

        ; A "BOOT" ON AN 11/04 OR 11/34 PRINTS R0, R4, SP, R7 ON THE TERMINAL.
        ; IF AN ERROR HALT OCCURS AT BOOT+774 WHILE BOOTING THEN
        ; BOOTING AGAIN ON AN 11/04 OR /34 PRINTS OUT THE FOLLOWING.
        ;       R0 = CURRENT DRIVE # BEING BOOTED FROM.
        ;       R4 = LOAD ADDRESS WHERE ERROR OCCURRED
        ;       SP = DEFINITIVE STATUS OF ERROR
        ;       R7 = ERROR HALT ADDR+2

        ; NOTE - A HALT OR HANGUP OCCURRING BETWEEN 742-746 THAT WILL NOT
        ;    RESPOND TO BREAK OR HALT IS GENERALLY DUE TO LACK OF DMA GRANT
        ;    CONTINUITY ON THE BUS. USER SHOULD PUT DSD880 INTERFACE CARD
        ;    CLOSER TO THE PROCESSOR AND ENSURE GRANT CONTINUITY.

        ; DSD880 - RX02 REGISTER SYNTAX DEFS
        RXCS=   177170
        ; ERR   INI XM XM  X02 ?? SID  DEN TRQ IEN  DON UN1 FUN  FUN FUN GO
                ;       100000  ; ERR   ERROR FLAG
                ;        40000  ; INI   LOAD INTO RXCS TO INITIALIZE
                ;        30000  ; XM    EXTENDED MEMORY SELECT BITS
                ;        4000   ; X02   = 1 FOR RX02 MODE SYNTAX
                ;        400    ; DEN   SET = 1 FOR DOUBLE DENSITY
                ;        200    ; TRQ   TRANSFER REQUEST - DATA TO/FROM RXDB
                ;        16     ; FUN   FUNCTION <0-7> - SET "GO" TO EXEC

        RXDB=RXCS+2                 ; RXES ERROR BIT LAYOUT
        ;         NXM  WCV SID   DRV DRV DEL    DSK DEN ACL    INT SID CRC
        ;              OVF  #1    #1 RDY DAT    DEN ERR LOW    DON RDY ERR

        ; REGISTER USAGE IN BOT880 SECTION
        XCS=    %1        ; R1     POINTER TO RXCS
        XDB=    %2        ; R2     POINTER TO RXDB
                          ; R3     READ COMMAND VAL WITH DENSITY BIT
        LDP=    %4        ; R4     LOAD POINTER
        SCT=    %5        ; R5     CURRENT SECTOR # (1, 3, 5, 7)
                          ; (SP)   WORD COUNT FOR CURRENT DENSITY
```

Table 4-8. DSD 880x/8 Bootstrap Program Listing (Cont) (LSI-11 8832 Interface)

```
DSD 880 BOOTSTRAP PROM  MACRO V04.00  23-OCT-80 11:15:00 PAGE 2
LSI-11 VERSION

        ; RL01 / RL02 COMPATIBLE HARDWARE DEFS.
        RLCS=   174400  ; RL COMMAND STATUS REGISTER
        ; ERR  DE  NXM DLT  DCRC OPI DS1  DS0 CRDY IE  A17 A16 F2  F1 F0 DRDY
        ;              HNF  HCRC OPI
        ; RO   RO  RO  RO   RO   RO  R/W  R/W R/W  R/W R/W R/W RW  RW RW RO
        ; 15   14  13  12   11   10  09   08  07   06  05  04  03  02 01 00
        ;
        ; FUNCTIONS      0 0 0   00      NOOP
        ;                0 0 1   02      WRITE CHECK
        ;                0 1 0   04      GET STATUS
        ;                0 1 1   06      SEEK
        ;                1 0 0   10      READ HEADER
        ;                1 0 1   12      WRITE DATA
        ;                1 1 0   14      READ DATA
        ;                1 1 1   16      READ DATA - NO HEADER CHECK

        ; RLBA  = 174402        - BUS ADDRESS REGISTER
         .RLBA  = 2         ; OFFSET

        ; RLDA= 174404  - DISK ADDRESS REGISTER (SEEK)
         .RLDA= 4
        ; DF8  DF7 DF6 DF5  DF4 DF3 DF2  DF1 DF0 000  000 HS  000 DIR 000 001
        ;      DF7 - DF0       CYCLINDER DIFFERENCE TO SEEK
        ;      HS              SET = LOWER SIDE, CLEAR = UPPER
        ;      DIR             SET = SEEK INWARDS TOWARD SPINDLE
        ;                      CLR = SEEK OUTWARDS

        ; RLDA= 174404  - DISK ADDRESS DURING READ/WRITE DATA COMMANDS
        ; CA8  CA7 CA6 CA5  CA4 CA3 CA2  CA1 CA0 HS   SA5 SA4 SA3  SA2 SA1 SA0

        ;               - DISK ADDRESS DURING GET STATUS COMMAND
        ; 000  000 000 000  000 000 000  000 000 000  000 000 RST  000 000 000

        ; RLMP= 174406  - MULTI-PURPOSE REGISTER
         .RLMP= 6
        ; WDE  HCE WLK SKTO  SPE WGE VC  DSE 000 HS  CO  HO  BH   ST2 ST1 ST0


                                   ; START HERE FOR RL01 TYPE BOOT - @ 174400
   000000  012737  BOTW00: MOV     #RLCS, @#0       ; DO RL BOOT ON POWER UP
           174400
           000000
   000006  000413          BR      BOTENT

   000010  012737  BOTW10: MOV     #RLCS+10, @#0    ; DO ALTERNATIVE RL BOOT
           174410
           000000
   000016  000407          BR      BOTENT

   000020  012737  BOT170: MOV     #177170, @#0     ; DO STANDARD FLOPPY BOOT
           177170
           000000
   000026  000403          BR      BOTENT

   000030  012737  BOT150: MOV     #177150, @#0     ; DO ALTERNATIVE FLOPPY BOOT
           177150
           000000
```

Table 4-8. DSD 880x/8 Bootstrap Program Listing (Cont) (LSI-11 8832 Interface)

```
DSD 880 BOOTSTRAP PROM  MACRO V04.00  23-OCT-80 11:15:00 PAGE 2-1
LSI-11 VERSION

     54
     55 000036  011706  BOTENT: MOV     (PC), SP        ; SET STACK TO 12700
     56 000040  012700          MOV     #340, R0        ; LOCK OUT LINE TIME CLOCK
                000340
     58 000044  106400          MTPS    R0              ; BY SETTING TO PRIORITY 7.
     59 000046  000240          NOP                     ; SO SAME SIZE IN PDP-11 VERSION
     60                                                 ; ABOVE 2 WORDS BECOME
     61                                                 ; MOV   R0, @#177776
     65
     66 000050  004467          JSR     R4, MEMHGH      ; GET POINTER TO TRAP ROUTINE
                000010
     67
     68                 ; TRAP PROCESSOR FOR NON-EXISTANT MEMORY TIMEOUT
     69                 ; SETS CARRY AND RETURNS ON NON-EXISTANT MEMORY TRAP
     70
     71 000054  012766  TRAP4:  MOV     #341, 2(SP)     ; SETS CARRY ON TRAP TO 4
                000341
                000002
     72 000062  000002          RTI                     ; ALSO SETS CURRENT PRIORITY HIGH
     73
     74
     75                 ; NOW TEST FROM 10 TO TOP OF AVAILABLE CONTIGUOUS MEMORY
     76                 ; INIT VECTORS AND SET LOW TEST LIMIT TO 10
     77 000064  012701  MEMHGH: MOV     #4, R1          ; SET LOW MEM POINTER
                000004
     78 000070  010421          MOV     R4, (R1)+       ; LOAD TRAP VECTOR
     79 000072  010021          MOV     R0, (R1)+       ; LOAD TRAP PSW VALUE = 340
     80 000074  010102          MOV     R1, R2          ; INIT TO LOW MEMORY = 10
     81
     82                 ; FIND TOP OF AVAILABLE MEMORY
     83 000076  005022  2$:     CLR     (R2)+           ; FIND TOP OF MEMORY
     84 000100  103403           BCS     4$             ; CARRY SET BY TRAP TO 4
     85 000102  020227          CMP     R2, #160000     ; AT END OF PDP-11 ADDR SPACE?
                160000
     86 000106  103773           BLO    2$
     87 000110  005042  4$:     CLR     -(R2)           ; SET POINTER TO LAST LOCATION+2
     88 000112  004567          JSR     R5, MEMCHK      ; TEST TO TOP OF MEMORY
                000136
     89
     90 000116  005000          CLR     R0              ; INIT FOR LATER
     91 000120  011001          MOV     (R0), R1        ; GET R*CS POINTER
     92 000122  032701          BIT     #1000, R1       ; RX02 OR RL01 DEVICE?
                001000
     93 000126  001411           BEQ    BOTRL           ; BOOT VIA RX02 MODE IF 1000 BIT SET
     94
     95                 ; FILL EMPTY TEST - DONE AT MULTIPLE BUFFER ADDRESSES IN ORDER
     96                 ; TO TOGGLE ALL ADDRESS BITS IN SYSTEM MEMORY
     97 000130  004567          JSR     R5, FILEMP      ; DO FILL-EMPTY BUFFER TEST
                000252
     98 000134  000774          10+<5*100.>     ; START FILL AT BEGINNING OF
     99 000136  017700          10+<5*1624.>    ; PATTERN REPETITION LEFT BY RAM TEST
    100 000140  037676          10+<5*3262.>    ; DO DMA TEST ACROSS ALL ADDRESS BITS
    101 000142  077704          10+<5*6540.>    ; THAT CAN BE SET IN AVAILABLE MEMORY
    102                 ;       10+<5*9816.>    ; SO ALL BITS TOGGLE OK
    103 000144  000000           0      ; ADDRESS TERMINATOR
    104 000146  000167          JMP     BOT880          ; DO RX02 TYPE BOOT.
                000442
```

Table 4-8. DSD 880x/8 Bootstrap Program Listing (Cont) (LSI-11 8832 Interface)

```
DSD 880 BOOTSTRAP PROM  MACRO V04.00  23-OCT-80 11:15:00 PAGE 3
RL COMPATIBLE BOOT

     1                      .SBTTL  RL COMPATIBLE BOOT
     2                      ; BOOT USING RL01 PROTOCOL (UNOTE 063)
     3                      ; DISPATCH WITH R0 = 0, R1 = RLCS
     4
     5 000152  105711  BOTRL:  TSTB    (R1)              ; CHECK CONTROLLER READY
     6 000154  103777           BCS    .                 ; HANG IF NO BUS RESPONSE TO DEVICE
     7 000156  100375           BPL    BOTRL             ; ELSE WAIT FOR CONTROLLER RDY
     8 000160  012761          MOV     #13,   .RLDA(R1) ; DO RESET ON GET STATUS
               000013
               000004
     9 000166  012711          MOV     #4,    (R1)      ; LOAD GET STATUS FUNCTION
               000004
    10 000172  105711          TSTB           (R1)      ; WAIT FOR CONTROLLER READY
    11 000174  100376           BPL    .-2
    12 000176  012761          MOV     #177601,.RLDA(R1); SET MAXIMAL LENGTH SEEK OUTWARDS
               177601
               000004
    13 000204  012711          MOV     #6,    (R1)      ; LOAD RL01 SEEK COMMAND
               000006
    14 000210  105711          TSTB           (R1)      ; WAIT FOR CONTROLLER READY
    15 000212  100376           BPL    .-2
    16 000214  012761          MOV     #-400,.RLMP(R1)  ; SET WORDCOUNT FOR 1 BLOCK
               177400
               000006
    17 000222  010061          MOV     R0,    .RLDA(R1) ; LOAD A 0 INTO DISK ADDRESS REGISTER
               000004
    18 000226  012711          MOV     #14,   (R1)      ; ISSUE READ FUNCTION
               000014
    19 000232  105711          TSTB    (R1)             ; CONTROLLER READY?
    20 000234  100376           BPL    .-2
    21 000236  005711          TST     (R1)             ; ERROR?
    22 000240  100001           BPL    .+4
    23 000242  000000            HALT
    24 000244  021027          CMP     (R0), #240       ; LOC 0 MUST BE NOP
               000240
    25 000250  001340           BNE    BOTRL            ; JUST TRY AGAIN
    26 000252  005007          CLR     PC               ; DISPATCH TO LOC 0.
```

Table 4-8. DSD 880x/8 Bootstrap Program Listing (Cont) (LSI-11 8832 Interface)

```
DSD 880 BOOTSTRAP PROM  MACRO V04.00  23-OCT-80 11:15:00 PAGE 4
RL COMPATIBLE BOOT

                         ; ROUTINE TO TEST MEMORY FROM C(R1) = LOW LIMIT
                         ; TO C(R2) = UPPER LIMIT BEYOND TEST
                         ; IF ERROR FOUND HALTS WITH R4 POINTING TO ERROR LOC, OR 2 BEYOND.
                         ; R0 = DATA READ

000254  010104  MEMCHK: MOV    R1, R4          ; GET STARTING ADDRESS
000256  010400  2$:     MOV    R4, R0          ; KILL Z FLAG <MOV R4, (R4)+>
000260  010024          MOV    R0, (R4)+       ; LOAD CONTENTS = ADDRESS
000262  020402          CMP    R4, R2          ; AT END OF TEST?
000264  103774           BLO   2$
000266  024404  CHKADP: CMP    -(R4), R4       ; CHECK BACK DOWN TO START ADDR
000270  001402           BEQ   NCKADP
000272  011400            MOV  (R4), R0        ; DATA READ IN ERROR IN R0
000274  000000            HALT                 ; STUCK BIT IN DATA OR ADDRESS!!
000276  020401  NCKADP: CMP    R4, R1
000300  101372           BHI   CHKADP          ; CONTINUE TILL AT START ADDR

000302  005124  SETCOM: COM    (R4)+           ; MAKE LOC = ADDR COMPLEMENT
000304  020402          CMP    R4, R2          ; AT END OF TEST?
000306  103775           BLO   SETCOM

000310  010104          MOV    R1, R4          ; START AT BEGINNING
000312  060414  CHKCOM: ADD    R4, (R4)        ; SHOULD BE ALL 1'S
000314  005214          INC    (R4)
000316  012400          MOV    (R4)+, R0       ; DATA SHOULD = ALL ZEROES
000320  001401           BEQ    NCKCOM
000322  000000            HALT                 ; STUCK DATA BIT IF NO HALT AT +156
000324  020402  NCKCOM: CMP    R4, R2
000326  103771           BLO   CHKCOM

                         ; SET UP TO LEAVE A PATTERN OF 1 011 001 110 001 111 B ROTATED
                         ; RIGHT INTO 4 SUCCESSIVE WORDS
                         ; USED AS MEM BACKGROUND AND FILL-EMPTY DATA.

000330  010104          MOV    R1, R4          ; SET INITIAL ADDRESS
000332  012703  SETPAT: MOV    #131617, R3     ; SET INITIAL PATTERN
        131617
000336  020402  4$:     CMP    R4, R2          ; END OF ADDRESS RANGE?
000340  103004           BHIS  CHKPAT          ; GO CHECK DATA IF AT END
000342  010324          MOV    R3, (R4)+       ; CARRY SET BY CMP INSTRUCTION.
000344  006203          ASR    R3              ; ROTATE AND LOAD AGAIN
000346  103773           BCS   4$
000350  000770           BR    SETPAT

000352  010104  CHKPAT: MOV    R1, R4          ; SET INITIAL ADDRESS
000354  012703  CHKPTL: MOV    #131617, R3
        131617
000360  020324  3$:     CMP    R3, (R4)+       ; DATA OK?
000362  001403           BEQ   4$
000364  016400            MOV  -2(R4), R0      ; SET DATA READ FOR LOOKING
        177776
000370  000000            HALT                 ; PATTERN SENSITIVITY ERROR
000372  020402  4$:     CMP    R4, R2          ; AT END OF ADDRESS RANGE?
000374  103003           BHIS  FILEXT          ; YES - EXIT
000376  006203          ASR    R3              ; CARRY SET BY CMP INSTRUCTION
000400  103767           BCS   3$
000402  000764           BR    CHKPTL
000404  000205  FILEXT: RTS    R5
```

Table 4-8. DSD 880x/8 Bootstrap Program Listing (Cont) (LSI-11 8832 Interface)

```
DSD 880 BOOTSTRAP PROM  MACRO V04.00  23-OCT-80 11:15:00 PAGE 5
RL COMPATIBLE BOOT

                        ; FILL - EMPTY BUFFER TEST

 000406  012504  FILEMP: MOV     (R5)+, R4       ; GET BUFFER ADDRESS
 000410  001775          BEQ     FILEXT
 000412  005764          TST     404(R4)         ; DOES MEMORY EXIST?
         000404
 000416  103773          BCS     FILEMP          ; NO - STEP TO END OF LIST
 000420  010102          MOV     XCS, XDB        ; INIT FOR RXDB
 000422  004767          CALL    WTFLAG          ; WAIT FOR DONE FLAG UP
         000342
 000426  103777          BCS     .               ; LOOP IF NO BUS RESPONSE
 000430  005711          TST     (R1)            ; RX02 ERROR SET?
 000432  100001          BPL     .+4             ; HALT IF ERROR
 000434  000000          HALT                    ; INTERFACE SETUP ERROR

                        ; DSD880 - RX02 INTERFACE LATCHED BIT TEST

 000436  012722          MOV     #1420, (XDB)+   ; LOAD INTO RXCS
         001420
 000442  022711          CMP     #5460, (XCS)    ; DID THEY LATCH OK?
         005460
 000446  001401          BEQ     .+4
 000450  000000          HALT                    ; STUCK BITS IN RXCS
 000452  022712          CMP     #1420, (XDB)    ; LATCHED OK IN RXDB?
         001420
 000456  001005          BNE     RXHALT          ; NO - BAD INTERFACE.

 000460  012712  RXDBTS: MOV     #173767, (XDB)  ; CHECK RXDB LATCH
         173767
 000464  022712          CMP     #173767, (XDB)  ; DID THEY LATCH
         173767
 000470  001401          BEQ     .+4
 000472  000000  RXHALT: HALT                    ; HALT IF INCORRECT BIT LATCHUP

 000474  010102  RXFIEM: MOV     XCS, XDB        ; SET UP RXDB POINTER
 000476  012746          MOV     #200, -(SP)     ; SAVE THE WORD-COUNT
         000200
 000502  012722          MOV     #401, (XDB)+    ; DO FILL COMMAND
         000401
 000506  105711          TSTB    (XCS)           ; WAIT FOR TRREQ
 000510  100376          BPL     .-2
 000512  011612          MOV     (SP), (XDB)     ; WORDCOUNT (=200)
 000514  105711          TSTB    (XCS)           ; WAIT FOR TRREQ
 000516  100376          BPL     .-2
 000520  010412          MOV     R4, (XDB)       ; BUFFER ADDR
 000522  004767          CALL    WTFLAG          ; WAIT FOR DONE OR TRREQ
         000242

                        ; NOW EMPTY SECTOR BUFFER AND CHECK DATA VALIDITY

 000526  022424  EMPBFT: CMP     (R4)+, (R4)+    ; BUMP EMPTY BUFFER ADDR
                                                 ; SO ERROR IF NO DATA TRANSFER.
 000530  012711          MOV     #403, (XCS)     ; DO EMPTY BUFFER COMMAND
         000403
 000534  010403          MOV     R4, R3          ; SAVE BUFFER START ADDRESS
 000536  105711          TSTB    (XCS)           ; WAIT FOR TRREQ
 000540  100376          BPL     .-2
```

Table 4-8. DSD 880x/8 Bootstrap Program Listing (Cont) (LSI-11 8832 Interface)

```
DSD 880 BOOTSTRAP PROM  MACRO V04.00  23-OCT-80 11:15:00 PAGE 5-1
RL COMPATIBLE BOOT

   000542  011612           MOV     (SP), (XDB)       ; LOAD WORD COUNT
   000544  105711          TSTB     (XCS)             ; WAIT FOR TRREQ
   000546  100376           BPL     .-2
   000550  010412           MOV     R4, (XDB)         ; AND FILL BUFFER ADDR+2
   000552  004767          CALL     WTFLAG            ; WAIT FOR ERROR, DONE OR TRREQ
           000212

   000556  006316  CHKEMP: ASL      (SP)     ; MAKE WORD COUNT INTO BYTE COUNT
   000560  010402          MOV      R4, R2
   000562  062602          ADD      (SP)+, R2         ; SET R2 = END ADDR TO CHECK
   000564  004567          JSR      R5, CHKPTL        ; DO DATA CHECK
           177564
   000570  000706          BR       FILEMP            ; DO NEXT FILL-EMPTY
```

Table 4-8. DSD 880x/8 Bootstrap Program Listing (Cont) (LSI-11 8832 Interface)

```
DSD 880 BOOTSTRAP PROM  MACRO V04.00  23-OCT-80 11:15:00 PAGE 6
RL COMPATIBLE BOOT

                      ; BOOT THE DEVICE IN R1, REGISTERS USED AS INDICATED BELOW
                                      ; R0    SET TO 0
               000001 XCS=     %1     ; R1    POINTER TO RXCS
               000002 XDB=     %2     ; R2    POINTER TO RXDB
                                      ; R3    READ COMMAND VALUE WITH DENSITY BIT
               000004 LDP=     %4     ; R4    LOAD POINTER
               000005 SCT=     %5     ; R5    CURRENT SECTOR # (1, 3, 5, 7)
                                      ; (SP)  WORD COUNT FOR CURRENT DENSITY

                      ; LOADS DEFINITIVE ERROR CODE INTO STACK POINTER = SP
                      ; THEN HALTS.  A PROCEED WILL ATTEMPT TO BOOT THE NEXT DRIVE.

        000572 012711 DEFNST: MOV     #17, (XCS)      ; DO DEFINITIVE ERROR STATUS
               000017
        000576 105711 DEFNWT: TSTB    (XCS)           ; WAIT FOR TRREQ OR DONE
        000600 001776          BEQ    .-2
        000602 100002          BPL    DEFNRD
        000604 010412           MOV   LDP, (XDB)      ; STATUS UPWARDS FROM LOAD ADDR
        000606 000773           BR    DEFNWT

        000610 011406 DEFNRD: MOV     (LDP), SP       ; SHOW DEFINITIVE STATUS IN SP.
        000612 000000         HALT                    ; EXAMINE SP VALUE IF HERE

        000614 005000 BOT880: CLR     R0              ; USE AS 0
        000616 011706 BOOTR1: MOV     (PC), SP        ; INIT STACK POINTER
        000620 005004         CLR     LDP             ; INIT LOAD ADDRESS POINTER
        000622 012703         MOV     #7, R3          ; GET INITIAL COMMAND
               000007
        000626 012746         MOV     #100, -(SP)     ; SET LOW DENSITY WORDCOUNT
               000100
        000632 012705         MOV     #1, SCT         ; INIT SECTOR TO READ
               000001

        000636 004767 RDLP:   CALL    WTFLAG          ; WAIT FOR DONE FLAG SET?
               000126
        000642 010102         MOV     XCS, R2         ; COPY RXCS POINTER
        000644 010322         MOV     R3, (R2)+       ; LOAD READ COMMAND
        000646 105711         TSTB    (XCS)           ; WAIT FOR TRREQ
        000650 100376          BPL    .-2
        000652 010512         MOV     SCT, (XDB)      ; LOAD SECTOR
        000654 105711         TSTB    (XCS)
        000656 100376          BPL    .-2
        000660 012712         MOV     #1, (XDB)       ; LOAD TRACK
               000001
        000664 004767         CALL    WTFLAG          ; WAIT FOR DONE
               000100
        000670 005711         TST     (XCS)           ; CLUDGE SINCE DEC RX02 SETS ERROR
                                                      ; BEFORE IT SETS DONE
        000672 100010          BPL    EMPBUF          ; EMPTY IF NO ERROR
        000674 032712           BIT   #20, (XDB)      ; IS ERROR A DENSITY ERROR?
               000020
        000700 001734            BEQ  DEFNST          ; NO- DO DEFINITIVE STATUS
        000702 052703           BIS   #400, R3        ; SET COMMAND TO DOUBLE DENSITY
               000400
        000706 012716           MOV   #200, (SP)      ; SET TO D.D. WORD COUNT
               000200
        000712 000751           BR    RDLP            ; AND TRY READING AGAIN
```

Table 4-8. DSD 880x/8 Bootstrap Program Listing (Cont) (LSI-11 8832 Interface)

```
DSD 880 BOOTSTRAP PROM  MACRO V04.00  23-OCT-80 11:15:00 PAGE 6-1
RL COMPATIBLE BOOT

000714  010346  EMPBUF: MOV     R3, -(SP)       ; GET COMMAND COPY
000716  042716          BIC     #4, (SP)        ; MAKE INTO AN EMPTY BUFFER COMMAND
        000004
000722  012611          MOV     (SP)+, (XCS)    ; AND EXECUTE
000724  105711          TSTB    (XCS)           ; WAIT FOR FIRST TRREQ
000726  100376           BPL    .-2
000730  011612            MOV   (SP), (XDB)     ; LOAD THE WORD COUNT
000732  105711          TSTB    (XCS)
000734  100376           BPL    .-2
000736  010412            MOV   LDP, (XDB)      ; AND XFER ADDRESS
000740  004767          CALL    WTFLAG          ; WAIT FOR DONE OR TRREQ
        000024
000744  121027  EMPDON: CMPB    (R0), #240      ; INSURE FIRST INSTRUCT IS A NOP.
        000240
000750  001322           BNE    BOOTR1          ; NO - NOT VALID DATA AT LOC 0
                                                ; C(SP) = WORD COUNT
000752  061604          ADD     (SP), LDP       ; BUMP LOAD ADDRESS FOR NEXT SECT
000754  061604          ADD     (SP), LDP       ; ADD ACTUAL BYTE COUNT
000756  122525          CMPB    (SCT)+, (SCT)+  ; BUMP SECTOR # BY 2
000760  020427          CMP     LDP, #1000      ; FINISHED LOADING?
        001000
000764  002724           BLT    RDLP            ; READ NEXT SECTOR

000766  005007          CLR     PC              ; GO DISPATCH


                ; WAIT FOR FLOPPY FLAGS, DONE, ERROR, TRREQ

000770  032711  WTFLAG: BIT     #240, (XCS)     ; WAIT FOR DONE OR TRREQ
        000240
000774  001775           BEQ    WTFLAG          ; CAN'T TEST RX02 ERROR HERE
000776  000207          RETURN

001000          BOTLST::

        000020'         .END    BOT170
```

Table 4-8. DSD 880x/8 Bootstrap Program Listing (Cont) (LSI-11 8832 Interface)

```
DSD 880 BOOTSTRAP PROM  MACRO V04.00  23-OCT-80 11:15:00 PAGE 6-2
SYMBOL TABLE

BOOTR1   000616R      002 DEFNST   000572R      002 RXDB   = 177172
BOTENT   000036R      002 DEFNWT   000576R      002 RXDBTS  000460R      002
BOTLST   001000RG     002 EMPBFT   000526R      002 RXFIEM  000474R      002
BOTRL    000152R      002 EMPBUF   000714R      002 RXHALT  000472R      002
BOTW00   000000R      002 EMPDON   000744R      002 SCT    =%000005
BOTW10   000010R      002 FILEMP   000406R      002 SETCOM  000302R      002
BOT150   000030R      002 FILEXT   000404R      002 SETPAT  000332R      002
BOT170   000020R      002 LDP    =%000004           TRAP4   000054R      002
BOT880   000614R      002 MEMCHK   000254R      002 WTFLAG  000770R      002
CHKADP   000266R      002 MEMHGH   000064R      002 XCS    =%000001
CHKCOM   000312R      002 NCKADP   000276R      002 XDB    =%000002
CHKEMP   000556R      002 NCKCOM   000324R      002 .RLBA = 000002
CHKPAT   000352R      002 RDLP     000636R      002 .RLDA = 000004
CHKPTL   000354R      002 RLCS   = 174400           .RLMP = 000006
DEFNRD   000610R      002 RXCS   = 177170

. ABS.   000000      000
         000000      001
BOOT     001000      002
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  8192 WORDS  ( 32 PAGES)
DYNAMIC MEMORY AVAILABLE FOR  64 PAGES
DY0:BOT880,DY0:BOT880/L:TTM/C=BOT880
```

```
DSD 880 BOOTSTRAP PROM  MACRO V04.00   23-OCT-80 11:15:00 PAGE S-1
CROSS REFERENCE TABLE (CREF V04.00 )


$PDP11   1-8      1-9      2-57
.RLBA    2-18#
.RLDA    2-21#    3-8*     3-12*    3-17*
.RLMP    2-35#    3-16*
BOOTR1   6-24#    6-60
BOT150   2-53#
BOT170   2-50#    6-79
BOT880   2-104    6-23#
BOTENT   2-45     2-48     2-51     2-55#
BOTLST   6-77#
BOTRL    2-93     3-5#     3-7      3-25
BOTW00   2-44#
BOTW10   2-47#
CHKADP   4-11#    4-16
CHKCOM   4-23#    4-29
CHKEMP   5-53#
CHKPAT   4-38     4-44#
CHKPTL   4-45#    4-54     5-56
DEFNRD   6-16     6-20#
DEFNST   6-13#    6-44
DEFNWT   6-14#    6-18
EMPBFT   5-41#
EMPBUF   6-42     6-49#
EMPDON   6-59#
FILEMP   2-97     5-3#     5-6      5-57
FILEXT   4-51     4-55#    5-4
LDP      1-109#   6-6#     6-17     6-20     6-25*    6-57     6-62*
         6-63*    6-65
MEMCHK   2-88     4-6#
MEMHGH   2-66     2-77#
NCKADP   4-12     4-15#
NCKCOM   4-26     4-28#
RDLP     6-30#    6-47     6-66
RLCS     2-2#     2-44     2-47
RXCS     1-91#    1-101
RXDB     1-101#
RXDBTS   5-23#
RXFIEM   5-28#
RXHALT   5-21     5-26#
SCT      1-110#   6-7#     6-28*    6-35     6-64     6-64
SETCOM   4-18#    4-20
SETPAT   4-36#    4-42
TRAP4    2-71#
WTFLAG   5-8      5-37     5-51     6-30     6-39     6-58     6-73#
         6-74
XCS      1-106#   5-7      5-17     5-28     5-31     5-34     5-43*
         5-45     5-48     6-3#     6-13*    6-14     6-31     6-33
         6-36     6-40     6-51*    6-52     6-55     6-73
XDB      1-107#   5-7*     5-16*    5-20     5-23*    5-24     5-28*
         5-30*    5-33*    5-36*    5-47*    5-50*    6-4#     6-17*
         6-35*    6-38*    6-43     6-54*    6-57*
```

4.5.4 Bootstrap Program Listing (PDP-11 8830 and LSI-11 8836 Interface)

Table 4-9 contains a listing of the bootstrap program for DSD 880x/8 systems with 8830 and 8836 interface boards.

Table 4-9. DSD Bootstrap Program Listing (PDP-11 8830 and LSI-11 8836 Interface)

```
DSD 880/x/20/30 BOOTSTRAP PROM   MACRO M1113  05-OCT-81 19:26
TABLE OF CONTENTS

     1-    8    LSI-11 VERSION
     3-  214    RL COMPATIBLE BOOT
```

Table 4-9. DSD Bootstrap Program Listing (Cont) (PDP-11 8830 and LSI-11 8836 Interface)

```
DSD 880/x/20/30 BOOTSTRAP PROM   MACRO M1113  05-OCT-81 19:26  PAGE 1

                                 .TITLE  DSD 880-TAURUS BOOTSTRAP PROM
                                 ; BOT88T.MAC      8-JUL-81
 .SBTTL LSI-11 VERSION
; BOOTSTRAP FOR DSD880 FLOPPY / WINCHESTER DISK CONTROLLER
; BOOTS EITHER SINGLE OR DOUBLE DENSITY FLOPPIES
; ** NOTE ON BOOTING WHILE REAL TIME CLOCK IS ENABLED. **
; THIS BOOT CAN BE STARTED WITH A RUNNING REAL TIME CLOCK IN 2 WAYS.
; 1)  BY SETTING THE PSW AHEAD OF TIME TO DISABLE INTERRUPTS BY TYPING
;     "$S/ 340<CR>" AND "R7/ 773000<CR>" AND HITTING "P".
; 1) (LSI-11 /2 ONLY) BY ENSURING THAT THE STACK IS POINTING TO
;     NON-EXISTANT MEMORY THUS FORCING A DOUBLE BUS ERROR ON ANY
;     INTERRUPT AND BY TYPING
;    "773000G"  AND TYPING "P" IF HALTS OCCUR DUE TO ATTEMPTED INTERRUPTS.

; THE BOOTSTRAP PROCEEDS IN 4 STEPS
; 1)    SELECT DEVICE   DETERMINES DEVICE TO BE BOOTED
; 2)    RAM TEST        CHECKS ALL AVAILABLE MEMORY FOR STUCK BITS
;                       ON BOTH DATA AND ADDRESS LINES. <0-30K>
;                       DOES BOTH DATA = ADDRESS AND PATTERN TESTS
;                       1) CLEARS MEMORY TO 0'S AND SIZES MEMORY
;                       2) LOADS MEMORY = ADDRESS AND CHECKS
;                       3) LOADS MEMORY = ADDRESS COMPLEMENT, CHECKS
;                       4) LOADS MEMORY WITH THE REPEATING PATTERN OF
;                          131617, 154707, 166343, 173161, 175470
; CHECK FOR UNIT 0 OVERRIDE FROM KEYBOARD
; 3-WINCHESTER  READ IN BLOCK 0, START AT LOC 0
; 3-DY  FILL-EMPTY      CHECKS DSD880 - PROCESSOR DATA PATH FOR
;                       SYNTAX AND DATA ERRORS.  ALSO INSURE'S ALL
;                       AVAILABLE ADDRESS LINES TOGGLE UNDER DMA.
;                       CHECKS FILL-EMPTY WITH BUFFERS AT 774,
;                       17700, 37676, 77704, 137700 IF MEMORY EXISTS.
; 4-DY  BOOTSTRAP       READS IN BLOCK 0 FROM DISKETTE IN CORRECT
;                       DENSITY AND STARTS AT LOC 0

; ERROR HALTS OR HANG UP LOOPS  (ADDRESSES RELATIVE TO BOOT BASE ADDR)
; 146-52 LOOP  RL CONTROLLER NOT READY
; 150   HANG    RL CONTROLLER NOT RESPONDING AT ADDRESS
; 166-170, 212-214, 236-240      RL TYPE CONTROLLER HUNG
; 246   HALT    ERROR DURING READ BLOCK 0
; 300   HALT    MEMORY ERROR AT LOC (R4), CONTENTS SHOULD EQUAL ADDRESS
; 324      HALT  MEMORY ERROR AT -2(R4), EXPECT 0, CONTENTS=ADDRESS COMPLEMENTED
; 374      HALT  1) FILL-EMPTY ERROR IF R5=BOOT+622
;                2) MEMORY ERR IF R5=BOOT+116
; 424   HALT    ERROR ON FLOPPY, DEFINITIVE STATUS AT LOC 0 IN MEMORY
; 470      LOOP   DY DEVICE ADDRESS SELECTED FOR BOOTING DOESN'T RESPOND
; 500      HALT  ERROR FLAG IN RXCS SET AFTER INIT
; 524      HALT  RXCS OR RXDB INTERFACE REGISTER STUCK BIT PROBLEM
;               NOTE CONTENTS OF RXCS, RXDB <5460,1420>,<*,173767>
; 534-536, 544-546      TRANSFER REQUEST HANGUP  (FILL BUFFER)
; 566-570, 576-600      TRANSFER REQUEST HANGUP  (EMPTY BUFFER)

; 652-654, 660-662      TRANSFER REQUEST HANGUP  (DY-READ-BOOTSTRAP)
; 730-732, 740-742      TRANSFER REQUEST HANGUP  (DY-EMPTY-BOOTSTRAP)
; 770-774  LOOP DSD880 FLAG WAIT ROUTINE HANGUP


; START ADDRESSES
```

Table 4-9. DSD Bootstrap Program Listing (Cont) (PDP-11 8830 and LSI-11 8836 Interface)

```
DSD 880/x/20/30 BOOTSTRAP PROM   MACRO M1113  05-OCT-81 19:26  PAGE 1-1
LSI-11 VERSION

        ; BOOT+0    (TYPICALLY 173000)  BOOTS RL DEVICE WITH RLCS AT 174400
        ; BOOT+10   (TYPICALLY 173010)  BOOTS RL DEVICE WITH RLCS AT 174410
        ; BOOT+20   (TYPICALLY 173020)  BOOTS DY DEVICE WITH RXCS AT 177170
        ; BOOT+30   (TYPICALLY 173030)  BOOTS DY DEVICE WITH RXCS AT 177150
        ; BOOT+36   (TYPICALLY 173036)  GENERAL DEVICE ENTRANCE - USER
        ;                SET'S  LOCATION 0 = DESIRED RLCS OR RXCS
        ;                NOTE: THE BIT OF VALUE 1000 MUST BE SET FOR RX BOOTING
        ;                        IF REAL TIME CLOCK MUST BE LEFT ON THEN SET
        ;                        $S/ 340<CR> AND R7/ 173040<CR> AND PROCEED

        ; A "BOOT" ON AN 11/04 OR 11/34 PRINTS R0, R4, SP, R7 ON THE TERMINAL.
        ; IF AN ERROR HALT OCCURS AT BOOT+774 WHILE BOOTING THEN
        ; BOOTING AGAIN ON AN 11/04 OR /34 PRINTS OUT THE FOLLOWING.
        ;       R0 = CURRENT DRIVE # BEING BOOTED FROM.
        ;       R4 = LOAD ADDRESS WHERE ERROR OCCURRED
        ;       SP = DEFINITIVE STATUS OF ERROR
        ;       R7 = ERROR HALT ADDR+2

        ; NOTE - A HALT OR HANGUP OCCURRING BETWEEN 742-746 THAT WILL NOT
        ;    RESPOND TO BREAK OR HALT IS GENERALLY DUE TO LACK OF DMA GRANT
        ;    CONTINUITY ON THE BUS. USER SHOULD PUT DSD880 INTERFACE CARD
        ;    CLOSER TO THE PROCESSOR AND ENSURE GRANT CONTINUITY.

        ; DSD880 - RX02 REGISTER SYNTAX DEFS
        RXCS=   177170
        ; ERR   INI XM XM  X02 ?? SID  DEN TRQ IEN  DON UN1 FUN  FUN FUN GO
                ;       100000   ; ERR   ERROR FLAG
                ;        40000   ; INI   LOAD INTO RXCS TO INITIALIZE
                ;        30000   ; XM    EXTENDED MEMORY SELECT BITS
                ;        4000    ; X02   = 1 FOR RX02 MODE SYNTAX
                ;        400     ; DEN   SET = 1 FOR DOUBLE DENSITY
                ;        200     ; TRQ   TRANSFER REQUEST - DATA TO/FROM RXDB
                ;        16      ; FUN   FUNCTION <0-7> - SET "GO" TO EXEC

        RXDB=RXCS+2               ; RXES ERROR BIT LAYOUT
        ;          NXM  WCV SID   DRV DRV DEL   DSK DEN ACL   INT SID CRC
        ;               OVF  #1    #1 RDY DAT   DEN ERR LOW   DON RDY ERR

        ; REGISTER USAGE IN BOOTRX SECTION
        XCS=    %1        ; R1    POINTER TO RXCS
        XDB=    %2        ; R2    POINTER TO RXDB
                          ; R3    READ COMMAND VAL WITH DENSITY BIT
        LDP=    %4        ; R4    LOAD POINTER
        SCT=    %5        ; R5    CURRENT SECTOR # (1, 3, 5, 7)
                          ; (SP)  WORD COUNT FOR CURRENT DENSITY

        TKS=    177560    ; CONSOLE STATUS REGESTER DEFS FOR GETTING UNIT #
        TKB=    TKS+2
        TPS=    TKB+2
        TPB=    TPS+2
```

Table 4-9. DSD Bootstrap Program Listing (Cont) (PDP-11 8830 and LSI-11 8836 Interface)

```
DSD 880/x/20/30 BOOTSTRAP PROM   MACRO M1113  05-OCT-81 19:26  PAGE 2
LSI-11 VERSION

        ; RL01 / RL02 COMPATIBLE HARDWARE DEFS.
        RLCS=   174400  ; RL COMMAND STATUS REGISTER
        ; ERR  DE  NXM DLT  DCRC OPI DS1  DS0 CRDY IE  A17 A16 F2  F1 F0 DRDY
        ;              HNF  HCRC OPI
        ; RO   RO  RO  RO   RO   RO  R/W  R/W R/W  R/W R/W R/W RW  RW RW RO
        ; 15   14  13  12   11   10  09   08  07   06  05  04  03  02 01 00
        ;
        ; FUNCTIONS     0 0 0   00      NOOP
        ;               0 0 1   02      WRITE CHECK
        ;               0 1 0   04      GET STATUS
        ;               0 1 1   06      SEEK
        ;               1 0 0   10      READ HEADER
        ;               1 0 1   12      WRITE DATA
        ;               1 1 0   14      READ DATA
        ;               1 1 1   16      READ DATA - NO HEADER CHECK

        ; RLBA  = 174402        - BUS ADDRESS REGISTER
         .RLBA  = 2        ; OFFSET

        ; RLDA= 174404  - DISK ADDRESS REGISTER (SEEK)
         .RLDA= 4
        ; DF8  DF7 DF6 DF5  DF4 DF3 DF2  DF1 DF0 000  000 HS  000 DIR 000 001
        ;      DF7 - DF0       CYCLINDER DIFFERENCE TO SEEK
        ;      HS              SET = LOWER SIDE, CLEAR = UPPER
        ;      DIR             SET = SEEK INWARDS TOWARD SPINDLE
        ;                      CLR = SEEK OUTWARDS

        ; RLDA= 174404  - DISK ADDRESS DURING READ/WRITE DATA COMMANDS
        ; CA8  CA7 CA6 CA5  CA4 CA3 CA2  CA1 CA0 HS   SA5 SA4 SA3  SA2 SA1 SA0

        ;               - DISK ADDRESS DURING GET STATUS COMMAND
        ; 000  000 000 000  000 000 000  000 000 000  000 000 RST  000 000 000

        ; RLMP= 174406  - MULTI-PURPOSE REGISTER
         .RLMP= 6
        ; WDE  HCE WLK SKTO  SPE WGE VC  DSE 000 HS  CO  HO  BH   ST2 ST1 ST0


                                ; START HERE FOR RL01 TYPE BOOT - @ 174400
    000000  012737  BOTW00: MOV     #RLCS, @#0      ; DO RL BOOT ON POWER UP
            174400
            000000
    000006  000413          BR      BOTENT

    000010  012737  BOTW10: MOV     #RLCS+10, @#0   ; DO ALTERNATIVE RL BOOT
            174410
            000000
    000016  000407          BR      BOTENT

    000020  012737  BOT170: MOV     #177170, @#0    ; DO STANDARD FLOPPY BOOT
            177170
            000000
    000026  000403          BR      BOTENT

    000030  012737  BOT150  MOV     #177150, @#0    ; DO ALTERNATIVE FLOPPY BOOT
            177150
            000000
```

Table 4-9. DSD Bootstrap Program Listing (Cont)
(PDP-11 8830 and LSI-11 8836 Interface)

```
DSD 880/x/20/30 BOOTSTRAP PROM   MACRO M1113  05-OCT-81 19:26  PAGE 2-1
LSI-11 VERSION

    168
    169 000036  011706  BOTENT: MOV     (PC), SP        ; SET STACK TO 12700
    170 000040  012700          MOV     #340, RO        ; LOCK OUT LINE TIME CLOCK
                000340
    172 000044  106400          MTPS    RO              ; BY SETTING TO PRIORITY 7.
    173 000046  000240          NOP                     ; SO SAME SIZE IN PDP-11 VERSION
    174                                                 ; ABOVE 2 WORDS BECOME
    175                                                 ; MOV   RO, @#177776
    179
    180 000050  004467          JSR     R4, MEMHGH      ; GET POINTER TO TRAP ROUTINE
                000010
    181
    182                 ; TRAP PROCESSOR FOR NON-EXISTANT MEMORY TIMEOUT
    183                 ; SETS CARRY AND RETURNS ON NON-EXISTANT MEMORY TRAP
    184
    185 000054  012766  TRAP4:  MOV     #341, 2(SP)     ; SETS CARRY ON TRAP TO 4
                000341
                000002
    186 000062  000002          RTI                     ; ALSO SETS CURRENT PRIORITY HIGH
    187
    188
    189                 ; NOW TEST FROM 10 TO TOP OF AVAILABLE CONTIGUOUS MEMORY
    190                 , INIT VECTORS AND SET LOW TEST LIMIT TO 10
    191 000064  012701  MEMHGH: MOV     #4, R1          ; SET LOW MEM POINTER
                000004
    192 000070  010421          MOV     R4, (R1)+       ; LOAD TRAP VECTOR
    193 000072  010021          MOV     RO, (R1)+       ; LOAD TRAP PSW VALUE = 340
    194 000074  010102          MOV     R1, R2          ; INIT TO LOW MEMORY = 10
    195
    196                 ; FIND TOP OF AVAILABLE MEMORY
    197 000076  005022  2$:     CLR     (R2)+           ; FIND TOP OF MEMORY
    198 000100  103403           BCS     4$             ; CARRY SET BY TRAP TO 4
    199 000102  020227          CMP     R2, #160000     ; AT END OF PDP-11 ADDR SPACE?
                160000
    200 000106  103773           BLO    2$
    201 000110  005042  4$:     CLR     -(R2)           ; SET POINTER TO LAST LOCATION+2
    202 000112  004567          JSR     R5, MEMCHK      ; TEST TO TOP OF MEMORY
                000144
    203
    204 000116  005000  CHKDEV: CLR     RO              ; INIT UNIT # FOR LATER
    205 000120  011001          MOV     (RO), R1        ; GET R*CS POINTER
    206 000122  105737          TSTB    @#TKS           ; HAS A UNIT # BEEN TYPED
                177560
    207 000126  100004           BPL    10$             ; NO - DEFAULT TO 0
    208 000130  013700            MOV   @#TKB, RO       ; ELSE USE LOW 2 BITS AS UNIT #
                177562
    209                 ;         MOV   RO, @#TPB       ; ECHOE CHAR IF TYPED
    210 000134  042700            BIC   #^C3, RO        ; CLEAR ALL BUT UNIT BITS
                177774
    211 000140  032701  10$:    BIT     #1000, R1       ; RXO2 OR RLO1 DEVICE?
                001000
    212 000144  001130           BNE    RXFLEM          ; BOOT VIA RXO2 MODE IF 1000 BIT SET IN R*CS
```

Table 4-9. DSD Bootstrap Program Listing (Cont) (PDP-11 8830 and LSI-11 8836 Interface)

```
DSD 880/x/20/30 BOOTSTRAP PROM   MACRO M1113  05-OCT-81 19:26  PAGE 3
RL COMPATIBLE BOOT

                          .SBTTL  RL COMPATIBLE BOOT
                          ; BOOT USING RL01 PROTOCOL (UNOTE 063)
                          ; DISPATCH WITH R0 = UNIT #, R1 = RLCS

 000146   105711  BOOTRL: TSTB    (XCS)             ; CHECK CONTROLLER READY
 000150   103777          BCS     .                 ; HANG IF NO BUS RESPONSE TO DEVICE
 000152   100375          BPL     BOOTRL            ; ELSE WAIT FOR CONTROLLER RDY
 000154   012761          MOV     #13, .RLDA(XCS) ; DO RESET CONTROLLER ON GET STATUS
          000013
          000004
 000162   012711          MOV     #4,    (XCS)      ;RLCS - LOAD GET STATUS FUNCTION
          000004
 000166   105711          TSTB          (XCS)       ; WAIT FOR CONTROLLER READY
 000170   100376          BPL     .-2
 000172   012761          MOV     #177601,.RLDA(XCS) ; SET MAXIMAL LENGTH SEEK OUTWARDS
          177601
          000004
 000200   012703          MOV     #6*400, R3        ; SEEK COMMAND
          003000
 000204   050003          BIS     R0, R3            ; WITH UNIT BITS
 000206   000303          SWAB    R3                ; BACK TO UN  UN CR IE  DF DF FN  FN FN GO
 000210   010311          MOV     R3,    (XCS)      ; LOAD RL01 SEEK COMMAND
 000212   105711          TSTB          (XCS)       ; RLCS - WAIT FOR CONTROLLER READY
 000214   100376          BPL     .-2
 000216   012761          MOV     #-400,.RLMP(R1) ;RLWC - SET WORDCOUNT FOR 1 BLOCK
          177400
          000006
 000224   005061          CLR     .RLDA(R1)         ; LOAD A ZERO INTO DISK ADDRESS REG
          000004
 000230   062703          ADD     #6, R3            ; MAKE SEEK INTO A READ COMMAND
          000006
 000234   010311          MOV     R3,  (XCS)        ; ISSUE READ FUNCTION
 000236   105711          TSTB    (XCS)             ; CONTROLLER READY?
 000240   100376          BPL     .-2

 000242   005711  CHKNOP: TST     (XCS)             ; ERROR?
 000244   100001          BPL     .+4
 000246   000000          HALT
 000250   023727          CMP     @#0, #240         ; LOC 0 MUST BE NOP
          000000
          000240
 000256   001317          BNE     CHKDEV            ; CHECK IF DIFFERENT UNIT #
 000260   005007          CLR     PC                ; DISPATCH TO LOC 0.
```

Table 4-9. DSD Bootstrap Program Listing (Cont) (PDP-11 8830 and LSI-11 8836 Interface)

```
DSD 880/x/20/30 BOOTSTRAP PROM   MACRO M1113  05-OCT-81 19:26  PAGE 4
RL COMPATIBLE BOOT

                         ; ROUTINE TO TEST MEMORY FROM C(R1) = LOW LIMIT
                         ; TO C(R2) = UPPER LIMIT BEYOND TEST
                         ; IF ERROR FOUND HALTS WITH R4 POINTING TO ERROR LOC, OR 2 BEYOND.
                         ; R0 = UNIT #  (UNCHANGED)

 000262  010104  MEMCHK: MOV     R1, R4          ; GET STARTING ADDRESS
 000264  010403  2$:     MOV     R4, R3          ; KILL Z FLAG <MOV R4, (R4)+>
 000266  010324          MOV     R3, (R4)+       ; LOAD CONTENTS = ADDRESS
 000270  020402          CMP     R4, R2          ; AT END OF TEST?
 000272  103774           BLO    2$
 000274  024404  CHKADP: CMP     -(R4), R4       ; CHECK BACK DOWN TO START ADDR
 000276  001401           BEQ    NCKADP
 000300  000000            HALT                  ; STUCK BIT IN DATA OR ADDRESS!!
 000302  020401  NCKADP: CMP     R4, R1
 000304  101373           BHI    CHKADP          ; CONTINUE TILL AT START ADDR

 000306  005124  SETCOM: COM     (R4)+           ; MAKE LOC = ADDR COMPLEMENT
 000310  020402          CMP     R4, R2          ; AT END OF TEST?
 000312  103775           BLO    SETCOM

 000314  010104          MOV     R1, R4          ; START AT BEGINNING
 000316  060414  CHKCOM: ADD     R4, (R4)        ; SHOULD BE ALL 1'S
 000320  005224          INC     (R4)+           ; DATA SHOULD = ALL ZEROES
 000322  001401           BEQ     NCKCOM
 000324  000000            HALT                  ; STUCK DATA BIT IF NO HALT AT +156
 000326  020402  NCKCOM: CMP     R4, R2
 000330  103772           BLO    CHKCOM

 000332  012737          MOV     # D, @#TPB      ; PRINT A "D" AS PROMPT
         000104
         177566
                         ; SET UP TO LEAVE A PATTERN OF 1 011 001 110 001 111 B ROTATED
                         ; RIGHT INTO 4 SUCCESSIVE WORDS
                         , USED AS MEM BACKGROUND AND FILL-EMPTY DATA.

 000340  010104          MOV     R1, R4          ; SET INITIAL ADDRESS
 000342  012703  SETPAT: MOV     #131617, R3     ; SET INITIAL PATTERN
         131617
 000346  020402  4$:     CMP     R4, R2          ; END OF ADDRESS RANGE?
 000350  103004           BHIS   CHKPAT          ; GO CHECK DATA IF AT END
 000352  010324          MOV     R3, (R4)+       ; CARRY SET BY CMP INSTRUCTION.
 000354  006203          ASR     R3              ; ROTATE AND LOAD AGAIN
 000356  103773          BCS     4$
 000360  000770          BR      SETPAT

 000362  010104  CHKPAT: MOV     R1, R4          ; SET INITIAL ADDRESS
 000364  012703  CHKPTL: MOV     #131617, R3
         131617
 000370  020324  3$:     CMP     R3, (R4)+       ; DATA OK?
 000372  001401           BEQ    4$
 000374  000000            HALT                  ; PATTERN SENSITIVITY ERROR
 000376  020402  4$:     CMP     R4, R2          ; AT END OF ADDRESS RANGE?
 000400  103003           BHIS   MEMEXT          ; YES - EXIT
 000402  006203          ASR     R3              ; CARRY SET BY CMP INSTRUCTION
 000404  103771           BCS    3$
 000406  000766           BR     CHKPTL
 000410  000205  MEMEXT: RTS     R5
```

Table 4-9. DSD Bootstrap Program Listing (Cont)
(PDP-11 8830 and LSI-11 8836 Interface)

```
DSD 880/x/20/30 BOOTSTRAP PROM   MACRO M1113  05-OCT-81 19:26  PAGE 5
RL COMPATIBLE BOOT

                        ; DEFNST - DISPLAY RX02 DEFINITIVE STATUS STARTING AT LOC 0
                        ; PROCEDE WILL RETRY

     000412  012711  DEFNST  MOV     #17, (XCS)        ; DO DEFINITIVE ERROR STATUS
             000017
     000416  105711  DEFNWT  TSTB    (XCS)             ; WAIT FOR TRREQ OR DONE
     000420  100376           BPL    DEFNWT            ; WAIT FOR TRANSFER REQUEST
     000422  005012            CLR   (XDB)             ; STATUS UPWARDS FROM LOAD ADDR
     000424  000000          HALT                      ; DEFINITIVE STATUS AT LOC 0
                        ;    BR      CHKDEV            ; ACCEPT UNIT AGAIN ON PROCEDE


                        ; FILL EMPTY TEST - DONE AT MULTIPLE BUFFER ADDRESSES IN ORDER
                        ; TO TOGGLE ALL ADDRESS BITS IN SYSTEM MEMORY
     000426  004567  RXFLEM  JSR     R5, FILEMP        ; DO FILL-EMPTY BUFFER TEST
             000016
     000432  000034          10+<5*4>         ; START FILL AT BEGINNING OF
     000434  017700          10+<5*1624.>     ; PATTERN REPETITION LEFT BY RAM TEST
     000436  037676          10+<5*3262.>     ; DO DMA TEST ACROSS ALL ADDRESS BITS
     000440  077704          10+<5*6540.>     ; THAT CAN BE SET IN AVAILABLE MEMORY
     000442  000000           0       ; ADDRESS TERMINATOR
     000444     007           BYTE   7, 27    ; COMMAND SET BITS FOR UNIT 0,1,2,3
     000445     027
     000446     047          .BYTE   47,67    ; PROTECT AGAINST HIGH UNITS
     000447     067

                        ; NOTE - FILEMP DOES NOT RETURN BUT FLOWS THROUGH INTO BOOTRX

                        ; FILL - EMPTY BUFFER TEST

     000450  012504  FILEMP  MOV     (R5)+, R4         ; GET BUFFER ADDRESS
     000452  001464           BEQ    BOOTRX            ; GO BOOT UNIT IN R0
     000454  005764          TST     404(R4)           ; DOES MEMORY EXIST?
             000404
     000460  103773           BCS    FILEMP            ; NO - STEP TO END OF LIST
     000462  010102          MOV     XCS, XDB          ; INIT FOR RXDB
     000464  004767          CALL    WTFLAG            ; WAIT FOR DONE FLAG UP
             000300
     000470  103777           BCS    .                 ; LOOP IF NO BUS RESPONSE
     000472  005711          TST     (R1)              ; RX02 ERROR SET?
     000474  100001           BPL    .+4               ; HALT IF ERROR
     000476  000000            HALT                    ; INTERFACE SETUP ERROR

                        ; DSD880 - RX02 INTERFACE LATCHED BIT TEST

     000500  012722          MOV     #1420, (XDB)+     ; LOAD INTO RXCS
             001420
     000504  022711          CMP     #5460, (XCS)      ; DID THEY LATCH OK?
             005460
     000510  001005           BNE    RXHALT            ; LATCHED BIT COMPARE ERROR
                        ;    CMP     #1420, (XDB)      ; LATCHED OK IN RXDB?
                        ;     BNE    RXHALT            ; NO - BAD INTERFACE

     000512  012712  RXDBTS  MOV     #173767, (XDB)    ; CHECK RXDB LATCH
             173767
     000516  022712          CMP     #173767, (XDB)    ; DID THEY LATCH
             173767
```

Table 4-9. DSD Bootstrap Program Listing (Cont) (PDP-11 8830 and LSI-11 8836 Interface)

```
DSD 880/x/20/30 BOOTSTRAP PROM   MACRO M1113  05-OCT-81 19:26  PAGE 5-1
RL COMPATIBLE BOOT

000522  001401            BEQ    RXFIEM          ; DO FILL-EMPTY DATA TEST
000524  000000  RXHALT:   HALT                   ; HALT IF INCORRECT BIT LATCHUP

000526  010102  RXFIEM:  MOV     XCS, XDB        ; SET UP RXDB POINTER
000530  012722           MOV     #401, (XDB)+    ; DO FILL COMMAND
        000401
000534  105711           TSTB    (XCS)           ; WAIT FOR TRREQ
000536  100376            BPL    .-2
000540  012712             MOV   #200, (XDB)     ; WORDCOUNT (=200)
        000200
000544  105711           TSTB    (XCS)           ; WAIT FOR TRREQ
000546  100376            BPL    .-2
000550  010412             MOV   R4, (XDB)       ; BUFFER ADDR
000552  004767           CALL    WTFLAG          ; WAIT FOR DONE OR TRREQ
        000212

                ; NOW EMPTY SECTOR BUFFER AND CHECK DATA VALIDITY

000556  022424  EMPBFT:  CMP     (R4)+, (R4)+    ; BUMP EMPTY BUFFER ADDR
                                                 ; SO ERROR IF NO DATA TRANSFER.
000560  012711           MOV     #403, (XCS)     ; DO EMPTY BUFFER COMMAND
        000403
000564  010403           MOV     R4, R3          ; SAVE BUFFER START ADDRESS
000566  105711           TSTB    (XCS)           ; WAIT FOR TRREQ
000570  100376            BPL    .-2
000572  012712             MOV   #200, (XDB)     ; LOAD WORD COUNT
        000200
000576  105711           TSTB    (XCS)           ; WAIT FOR TRREQ
000600  100376            BPL    .-2
000602  010412             MOV   R4, (XDB)       ; AND FILL BUFFER ADDR+2
000604  004767           CALL    WTFLAG          ; WAIT FOR ERROR, DONE OR TRREQ
        000160

000610  010402  CHKEMP:  MOV     R4, R2          ; SET UP UPPER CHECK LIMIT
000612  062702           ADD     #400, R2        ; SET R2 = END ADDR TO CHECK
        000400
000616  004567           JSR     R5, CHKPTL      ; DO DATA CHECK
        177542
000622  000712           BR      FILEMP          ; DO NEXT FILL-EMPTY
```

Table 4-9. DSD Bootstrap Program Listing (Cont) (PDP-11 8830 and LSI-11 8836 Interface)

```
DSD 880/x/20/30 BOOTSTRAP PROM   MACRO M1113  05-OCT-81 19:26  PAGE 6
RL COMPATIBLE BOOT

                          ; BOOT THE DEVICE IN R1, REGISTERS USED AS INDICATED BELOW
                                    ; R0     LOGICAL UNIT #
            000001  XCS=    %1      ; R1    POINTER TO RXCS
            000002  XDB=    %2      ; R2    POINTER TO RXDB
                                    ; R3    READ COMMAND VALUE WITH DENSITY BIT
            000004  LDP=    %4      ; R4    LOAD POINTER
            000005  SCT=    %5      ; R5    CURRENT SECTOR # (1, 3, 5, 7)
                                    ; (SP)  WORD COUNT FOR CURRENT DENSITY

    000624  060005  BOOTRX: ADD     R0, R5          ; PTR TO READ UNIT N COMMAND
    000626  111503          MOVB    (R5), R3        ; GET COMMAND FOR UNIT
    000630  005004          CLR     LDP             ; INIT LOAD ADDRESS POINTER
    000632  012746          MOV     #100, -(SP)     ; SET LOW DENSITY WORDCOUNT
            000100
    000636  012705          MOV     #1, SCT         ; INIT SECTOR TO READ
            000001

    000642  004767  RDLP:   CALL    WTFLAG          ; WAIT FOR DONE FLAG SET?
            000122
    000646  010102          MOV     XCS, XDB        ; COPY RXCS POINTER
    000650  010322          MOV     R3, (XDB)+      ; LOAD READ COMMAND AND BUMP XDB TO RXDB
    000652  105711          TSTB    (XCS)           ; WAIT FOR TRREQ
    000654  100376           BPL    .-2
    000656  010512          MOV     SCT, (XDB)      ; LOAD SECTOR
    000660  105711          TSTB    (XCS)
    000662  100376           BPL    .-2
    000664  012712          MOV     #1, (XDB)       ; LOAD TRACK
            000001
    000670  004767          CALL    WTFLAG          ; WAIT FOR DONE
            000074
    000674  005711          TST     (XCS)           ; CLUDGE SINCE DEC RX02 SETS ERROR
                                                    ; BEFORE IT SETS DONE
    000676  100010           BPL    EMPBUF          ; EMPTY IF NO ERROR
    000700  032712            BIT   #20, (XDB)      ; IS ERROR A DENSITY ERROR?
            000020
    000704  001642              BEQ  DEFNST         ; NO- DO DEFINITIVE STATUS
    000706  052703               BIS #400, R3       ; SET COMMAND TO DOUBLE DENSITY
            000400
    000712  012716               MOV #200, (SP)     ; SET TO D.D. WORD COUNT
            000200
    000716  000751                BR RDLP           ; AND TRY READING AGAIN

    000720  010346  EMPBUF: MOV     R3, -(SP)       ; GET COMMAND COPY
    000722  042716          BIC     #4, (SP)        ; MAKE INTO AN EMPTY BUFFER COMMAND
            000004
    000726  012611          MOV     (SP)+, (XCS)    ; AND EXECUTE
    000730  105711          TSTB    (XCS)           ; WAIT FOR FIRST TRREQ
    000732  100376           BPL    .-2
    000734  011612            MOV   (SP), (XDB)     ; LOAD THE WORD COUNT
    000736  105711          TSTB    (XCS)
    000740  100376           BPL    .-2
    000742  010412            MOV   LDP, (XDB)      ; AND XFER ADDRESS
    000744  004767          CALL    WTFLAG          ; WAIT FOR DONE OR TRREQ
            000020
    000750  061604          ADD     (SP), LDP       ; BUMP LOAD ADDRESS FOR NEXT SECT
    000752  061604          ADD     (SP), LDP       ; ADD ACTUAL BYTE COUNT
    000754  122525          CMPB    (SCT)+, (SCT)+  ; BUMP SECTOR # BY 2
```

Table 4-9. DSD Bootstrap Program Listing (Cont) (PDP-11 8830 and LSI-11 8836 Interface)

```
DSD 880/x/20/30 BOOTSTRAP PROM   MACRO M1113  05-OCT-81 19:26  PAGE 6-1
RL COMPATIBLE BOOT

000756  020427           CMP    LDP, #1000       ; FINISHED LOADING?
        001000
000762  002727            BLT   RDLP             ; READ NEXT SECTOR

000764  000167           JMP    CHKNOP           ; CHECK LOC 0 = NOP AND DISPATCH
        177252


                 ; WAIT FOR FLOPPY FLAGS, DONE, ERROR, TRREQ

000770  032711  WTFLAG: BIT     #240, (XCS)      ; WAIT FOR DONE OR TRREQ
        000240
000774  001775            BEQ   WTFLAG           ; CAN'T TEST RX02 ERROR HERE
000776  000207           RETURN

001000          BOTLST::

        000020'          .END   BOT170
```

Table 4-9. DSD Bootstrap Program Listing (Cont) (PDP-11 8830 and LSI-11 8836 Interface)

```
DSD 880/x/20/30 BOOTSTRAP PROM    MACRO M1113  05-OCT-81 19:26  PAGE 6-2
SYMBOL TABLE

BOOTRL  000146R      002 DEFNWT  000416R      002 RXFLEM  000426R      002
BOOTRX  000624R      002 EMPBFT  000556R      002 RXHALT  000524R      002
BOTENT  000036R      002 EMPBUF  000720R      002 SCT    =%000005
BOTLST  001000RG     002 FILEMP  000450R      002 SETCOM  000306R      002
BOTW00  000000R      002 LDP    =%000004          SETPAT  000342R      002
BOTW10  000010R      002 MEMCHK  000262R      002 TKB    = 177562
BOT150  000030R      002 MEMEXT  000410R      002 TKS    = 177560
BOT170  000020R      002 MEMHGH  000064R      002 TPB    = 177566
CHKADP  000274R      002 NCKADP  000302R      002 TPS    = 177564
CHKCOM  000316R      002 NCKCOM  000326R      002 TRAP4   000054R      002
CHKDEV  000116R      002 RDLP    000642R      002 WTFLAG  000770R      002
CHKEMP  000610R      002 RLCS   = 174400          XCS    =%000001
CHKNOP  000242R      002 RXCS   = 177170          XDB    =%000002
CHKPAT  000362R      002 RXDB   = 177172          .RLBA  = 000002
CHKPTL  000364R      002 RXDBTS  000512R      002 .RLDA  = 000004
DEFNST  000412R      002 RXFIEM  000526R      002 .RLMP  = 000006

  ABS.  000000     000
        000000     001
BOOT    001000     002
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  331 WORDS  ( 2 PAGES)
DYNAMIC MEMORY:  2822 WORDS  ( 10 PAGES)
ELAPSED TIME:  00:00:12
,BOT88T/CR=BOT88T
```

```
BOTBBT      CREATED BY  MACRO  ON 5-OCT-81 AT 19 26        PAGE 1

SYMBOL CROSS REFERENCE                                     CREF

SYMBOL   VALUE              REFERENCES

BOOTRL     000146 R      #3-218     3-220
BOOTRX     000624 R       5-227    #6-388
BOTENT     000036 R       2-159     2-162      2-165     #2-169
BOTLST     001000 RG     #6-438     6-439
BOTW00     000000 R      #2-158     6-439
BOTW10     000010 R      #2-161
BOT1SC     000030 R      #2-167
BOT170     000020 R      #2-164     6-441
CHKADP     000274 R      #4-256     4-260
CHKCOM     000316 R      #4-267     4-272
CHKDEV     000116 R      #2-204     3-243
CHKEMP     000610 R      #5-374
CHKNOP     000242 R      #3-239     6-429
CHKPAT     000362 R       4-282    #4-288
CHKPTL     000364 R      #4-289     4-297      5-376
DEFNST     000412 R      #5-303     6-408
DEFNWT     000416 R      #5-304     5-305
EMPBFT     000556 R      #5-362
EMPBUF     000720 R       6-406    #6-413
FILEMP     000450 R       5-313    #5-326      5-329      5-377
LDP      =%000004        #1-106    #6-384     *6-390      6-421     *6-423
                         *6-424     6-426
MEMCHK     000262 R       2-202    #4-251
MEMEXT     000410 R       4-294    #4-298
MEMHGH     000064 R       2-180    #2-191
NCKADP     000302 R       4-257    #4-259
NCKCOM     000326 R       4-269    #4-271
RDLP       000642 R      #6-394     6-411      6-427
RLCS     = 174400        #2-116     2-158      2-161
RXCS     = 177170        #1-88      1-98
RXDB     = 177172        #1-98
RXDBTS     000512 R      #5-345
RXFIEM     000526 R       5-347    #5-350
RXFLEM     000426 R       2-212    #5-313
RXHALT     000524 R       5-341    #5-348
SCT      =%000005        #1-107    #6-385     *6-392      6-399     *6-425
                         *6-425
SETCOM     000306 R      #4-262     4-264
SETPAT     000342 R      #4-280     4-286
TKB      = 177562        #1-111     1-112      2-208
TKS      = 177560        #1-110     1-111      2-206
TPB      = 177566        #1-113    *4-274
TPS      = 177564        #1-112     1-113
TRAP4      000054 R      #2-185
WTFLAG     000770 R       5-331     5-358      5-372      6-394      6-403
                          6-422    #6-434      6-435
XCS      =%000001        #1-103     3-218      3-221      3-222      3-223
                          3-225     3-229      3-230      3-235      3-236
                          3-239     5-303      5-304      5-330      5-340
                          5-350     5-352      5-355      5-364      5-366
                          5-369    #6-381      6-395      6-397      6-400
                          6-404     6-415      6-416      6-419      6-434
```

Table 4-9. DSD Bootstrap Program Listing (Cont) (PDP-11 8830 and LSI-11 8836 Interface)

```
BOTBBT      CREATED BY  MACRO  ON 5-OCT-81 AT 19.26        PAGE 2

SYMBOL CROSS REFERENCE                                     CREF

SYMBOL   VALUE            REFERENCES

XDB      =%000002         #1-104      5-306      #5-330     #5-339     5-345
                           5-346     #5-350      #5-351      5-354     5-357
                           5-368      5-371      #6-382     #6-395    #6-396
                           6-399      6-402       6-407      6-418     6-421
$PDP11   =  ******         1-8        1-9         2-171
.RLBA    =  000002        #2-132
.RLDA    =  000004        #2-135     #3-221      #3-225     #3-233
.RLMP    =  000006        #2-149     #3-232
```

4.5.5 Bootstrap Program Listing (PDP 11/45 Modification Version)

Table 4-10 contains the bootstrap program listing for use with the PDP 11/45 modification version.

Table 4-10. DSD 880x/8 Bootstrap Program Listing (PDP 11/45 Modification Version)

```
                                          DSD 880-TAURUS BOOTSTRAP PROM   MACRO M1113  06-JAN-82 15:59
TABLE OF CONTENTS

     1-   10    PDP-11 VERSION
     3-  216    RL COMPATIBLE BOOT
```

Table 4-10. DSD 880x/8 Bootstrap Program Listing (Cont) (PDP 11/45 Modification Version)

```
DSD 880-TAURUS BOOTSTRAP PROM   MACRO M1113  06-JAN-82 15:59  PAGE 1

                                      .TITLE  DSD 880-TAURUS BOOTSTRAP PROM
            000001                    $PDP11=1          ; 11/45 MODIFICATION VERSION  16-NOV-81
                                      ; BOT88T.MAC     16-NOV-81
  .SBTTL PDP-11 VERSION
  ; BOOTSTRAP FOR DSD880 FLOPPY / WINCHESTER DISK CONTROLLER
  ; BOOTS EITHER SINGLE OR DOUBLE DENSITY FLOPPIES
  ; ** NOTE ON BOOTING WHILE REAL TIME CLOCK IS ENABLED. **
  ; THIS BOOT CAN BE STARTED WITH A RUNNING REAL TIME CLOCK IN 2 WAYS.
  ; 1)   BY SETTING THE PSW AHEAD OF TIME TO DISABLE INTERRUPTS BY TYPING
  ;      "$S/ 340<CR>" AND "R7/ 773000<CR>" AND HITTING "P".
  ; 1)  (LSI-11 /2 ONLY) BY ENSURING THAT THE STACK IS POINTING TO
  ;      NON-EXISTANT MEMORY THUS FORCING A DOUBLE BUS ERROR ON ANY
  ;      INTERRUPT AND BY TYPING
  ;     "773000G"  AND TYPING "P" IF HALTS OCCUR DUE TO ATTEMPTED INTERRUPTS.

  ; THE BOOTSTRAP PROCEEDS IN 4 STEPS
  ; 1)      SELECT DEVICE    DETERMINES DEVICE TO BE BOOTED
  ; 2)      RAM TEST         CHECKS ALL AVAILABLE MEMORY FOR STUCK BITS
  ;                          ON BOTH DATA AND ADDRESS LINES. <0-30K>
  ;                          DOES BOTH DATA = ADDRESS AND PATTERN TESTS
  ;                          1) CLEARS MEMORY TO 0'S AND SIZES MEMORY
  ;                          2) LOADS MEMORY = ADDRESS AND CHECKS
  ;                          3) LOADS MEMORY = ADDRESS COMPLEMENT, CHECKS
  ;                          4) LOADS MEMORY WITH THE REPEATING PATTERN OF
  ;                             131617, 154707, 166343, 173161, 175470
  ; CHECK FOR UNIT 0 OVERRIDE FROM KEYBOARD
  ; 3-WINCHESTER  READ IN BLOCK 0, START AT LOC 0.
  ; 3-DY  FILL-EMPTY         CHECKS DSD880 - PROCESSOR DATA PATH FOR
  ;                          SYNTAX AND DATA ERRORS.  ALSO INSURE'S ALL
  ;                          AVAILABLE ADDRESS LINES TOGGLE UNDER DMA.
  ;                          CHECKS FILL-EMPTY WITH BUFFERS AT 774,
  ;                          17700, 37676, 77704, 137700 IF MEMORY EXISTS.
  ; 4-DY  BOOTSTRAP          READS IN BLOCK 0 FROM DISKETTE IN CORRECT
  ;                          DENSITY AND STARTS AT LOC 0

  ; ERROR HALTS OR HANG UP LOOPS  (ADDRESSES RELATIVE TO BOOT BASE ADDR)
  ; 146-52 LOOP  RL CONTROLLER NOT READY
  ; 150    HANG    RL CONTROLLER NOT RESPONDING AT ADDRESS
  ; 166-170, 212-214, 236-240      RL TYPE CONTROLLER HUNG
  ; 246    HALT    ERROR DURING READ BLOCK 0
  ; 300    HALT    MEMORY ERROR AT LOC (R4), CONTENTS SHOULD EQUAL ADDRESS
  ; 324      HALT  MEMORY ERROR AT -2(R4), EXPECT 0, CONTENTS=ADDRESS COMPLEMENTED
  ; 374      HALT  1) FILL-EMPTY ERROR IF R5=BOOT+622
  ;                2) MEMORY ERR IF R5=BOOT+116
  ; 424    HALT    ERROR ON FLOPPY, DEFINITIVE STATUS AT LOC 0 IN MEMORY
  ; 470      LOOP   DY DEVICE ADDRESS SELECTED FOR BOOTING DOESN'T RESPOND
  ; 500      HALT  ERROR FLAG IN RXCS SET AFTER INIT
  ; 524      HALT  RXCS OR RXDB INTERFACE REGISTER STUCK BIT PROBLEM
  ;                NOTE CONTENTS OF RXCS, RXDB <5460,1420>,<*,173767>
  ; 534-536, 544-546       TRANSFER REQUEST HANGUP  (FILL BUFFER)
  ; 566-570, 576-600       TRANSFER REQUEST HANGUP  (EMPTY BUFFER)

  ; 652-654, 660-662       TRANSFER REQUEST HANGUP  (DY-READ-BOOTSTRAP)
  ; 730-732, 740-742       TRANSFER REQUEST HANGUP  (DY-EMPTY-BOOTSTRAP)
  ; 770-774  LOOP DSD880 FLAG WAIT ROUTINE HANGUP
```

Table 4-10. DSD 880x/8 Bootstrap Program Listing (Cont) (PDP 11/45 Modification Version)

```
DSD 880-TAURUS BOOTSTRAP PROM   MACRO M1113  06-JAN-82 15:59  PAGE 1-1
PDP-11 VERSION

        ; START ADDRESSES
        ; BOOT+0    (TYPICALLY 173000)  BOOTS RL DEVICE WITH RLCS AT 174400
        ; BOOT+10   (TYPICALLY 173010)  BOOTS RL DEVICE WITH RLCS AT 174410
        ; BOOT+20   (TYPICALLY 173020)  BOOTS DY DEVICE WITH RXCS AT 177170
        ; BOOT+30   (TYPICALLY 173030)  BOOTS DY DEVICE WITH RXCS AT 177150
        ; BOOT+36   (TYPICALLY 173036)  GENERAL DEVICE ENTRANCE - USER
        ;                SET'S  LOCATION 0 = DESIRED RLCS OR RXCS
        ;                NOTE: THE BIT OF VALUE 1000 MUST BE SET FOR RX BOOTING
        ;                        IF REAL TIME CLOCK MUST BE LEFT ON THEN SET
        ;                        $S/ 340<CR> AND R7/ 173040<CR> AND PROCEED

        ; A "BOOT" ON AN 11/04 OR 11/34 PRINTS R0, R4, SP, R7 ON THE TERMINAL.
        ; IF AN ERROR HALT OCCURS AT BOOT+774 WHILE BOOTING THEN
        ; BOOTING AGAIN ON AN 11/04 OR /34 PRINTS OUT THE FOLLOWING.
        ;       R0 = CURRENT DRIVE # BEING BOOTED FROM.
        ;       R4 = LOAD ADDRESS WHERE ERROR OCCURRED
        ;       SP = DEFINITIVE STATUS OF ERROR
        ;       R7 = ERROR HALT ADDR+2

        ; NOTE - A HALT OR HANGUP OCCURRING BETWEEN 742-746 THAT WILL NOT
        ;    RESPOND TO BREAK OR HALT IS GENERALLY DUE TO LACK OF DMA GRANT
        ;    CONTINUITY ON THE BUS. USER SHOULD PUT DSD880 INTERFACE CARD
        ;    CLOSER TO THE PROCESSOR AND ENSURE GRANT CONTINUITY.

        ; DSD880 - RX02 REGISTER SYNTAX DEFS
        RXCS=   177170
        ; ERR  INI XM XM  X02 ?? SID  DEN TRQ IEN  DON UN1 FUN  FUN FUN GO
                ;      100000   ; ERR   ERROR FLAG
                ;       40000   ; INI   LOAD INTO RXCS TO INITIALIZE
                ;       30000   ; XM    EXTENDED MEMORY SELECT BITS
                ;       4000    ; X02   = 1 FOR RX02 MODE SYNTAX
                ;       400     ; DEN   SET = 1 FOR DOUBLE DENSITY
                ;       200     ; TRQ   TRANSFER REQUEST - DATA TO/FROM RXDB
                ;       16      ; FUN   FUNCTION <0-7> - SET "GO" TO EXEC

        RXDB=RXCS+2              ; RXES ERROR BIT LAYOUT
        ;        NXM  WCV SID   DRV DRV DEL   DSK DEN ACL   INT SID CRC
        ;             OVF  #1    #1 RDY DAT   DEN ERR LOW   DON RDY ERR

        ; REGISTER USAGE IN BOOTRX SECTION
        XCS=    %1      ; R1    POINTER TO RXCS
        XDB=    %2      ; R2    POINTER TO RXDB
                        ; R3    READ COMMAND VAL WITH DENSITY BIT
        LDP=    %4      ; R4    LOAD POINTER
        SCT=    %5      ; R5    CURRENT SECTOR # (1, 3, 5, 7)
                        ; (SP)  WORD COUNT FOR CURRENT DENSITY

        TKS=    177560  ; CONSOLE STATUS REGESTER DEFS FOR GETTING UNIT #
        TKB=    TKS+2
        TPS=    TKB+2
        TPB=    TPS+2
```

Table 4-10. DSD 880x/8 Bootstrap Program Listing (Cont) (PDP 11/45 Modification Version)

```
DSD 880-TAURUS BOOTSTRAP PROM   MACRO M1113  06-JAN-82 15:59  PAGE 2
PDP-11 VERSION

        ; RL01 / RL02 COMPATIBLE HARDWARE DEFS.
        RLCS=   174400  ; RL COMMAND STATUS REGISTER
        ; ERR  DE  NXM DLT  DCRC OPI DS1  DS0 CRDY IE  A17 A16 F2  F1 F0 DRDY
        ;              HNF  HCRC OPI
        ; RO   RO  RO  RO   RO   RO  R/W  R/W R/W  R/W R/W R/W RW  RW RW RO
        ; 15   14  13  12   11   10  09   08  07   06  05  04  03  02 01 00
        ;
        ; FUNCTIONS      0 0 0   00      NOOP
        ;                0 0 1   02      WRITE CHECK
        ;                0 1 0   04      GET STATUS
        ;                0 1 1   06      SEEK
        ;                1 0 0   10      READ HEADER
        ;                1 0 1   12      WRITE DATA
        ;                1 1 0   14      READ DATA
        ;                1 1 1   16      READ DATA - NO HEADER CHECK

        ; RLBA  = 174402        - BUS ADDRESS REGISTER
         .RLBA  = 2        ; OFFSET

        ; RLDA= 174404  - DISK ADDRESS REGISTER (SEEK)
         .RLDA= 4
        ; DF8  DF7 DF6 DF5  DF4 DF3 DF2  DF1 DF0 000  000 HS  000 DIR 000 001
        ;      DF7 - DF0        CYCLINDER DIFFERENCE TO SEEK
        ;      HS               SET = LOWER SIDE, CLEAR = UPPER
        ;      DIR              SET = SEEK INWARDS TOWARD SPINDLE
        ;                       CLR = SEEK OUTWARDS

        ; RLDA= 174404  - DISK ADDRESS DURING READ/WRITE DATA COMMANDS
        ; CA8  CA7 CA6 CA5  CA4 CA3 CA2  CA1 CA0 HS   SA5 SA4 SA3  SA2 SA1 SA0

        ;               - DISK ADDRESS DURING GET STATUS COMMAND
        ; 000  000 000 000  000 000 000  000 000 000  000 000 RST  000 000 000

        ; RLMP= 174406  - MULTI-PURPOSE REGISTER
         .RLMP= 6
        ; WDE  HCE WLK SKTO  SPE WGE VC  DSE 000 HS  CO  HO  BH   ST2 ST1 STO


                                  ; START HERE FOR RL01 TYPE BOOT - @ 174400
    000000  012737  BOTW00: MOV     #RLCS, @#0      ; DO RL BOOT ON POWER UP
            174400
            000000
    000006  000413          BR      BOTENT

    000010  012737  BOTW10: MOV     #RLCS+10, @#0   ; DO ALTERNATIVE RL BOOT
            174410
            000000
    000016  000407          BR      BOTENT

    000020  012737  BOT170: MOV     #177170, @#0    ; DO STANDARD FLOPPY BOOT
            177170
            000000
    000026  000403          BR      BOTENT

    000030  012737  BOT150: MOV     #177150, @#0    ; DO ALTERNATIVE FLOPPY BOOT
            177150
            000000
```

```
DSD 880-TAURUS BOOTSTRAP PROM   MACRO M1113  06-JAN-82 15:59  PAGE 2-1
PDP-11 VERSION

    169
    170 000036  011706  BOTENT: MOV     (PC), SP         ; SET STACK TO 12700
    171 000040  012700          MOV     #340, R0         ; LOCK OUT LINE TIME CLOCK
                000340
    178 000044  005002          CLR     R2               ; SET UP PTR FOR PSW FOR 11/45
    179 000046  010042          MOV     R0, -(R2)        ; LOAD PSW EXPLICITLY
    181
    182 000050  004467          JSR     R4, MEMHGH       ; GET POINTER TO TRAP ROUTINE
                000010
    183
    184                 ; TRAP PROCESSOR FOR NON-EXISTANT MEMORY TIMEOUT
    185                 ; SETS CARRY AND RETURNS ON NON-EXISTANT MEMORY TRAP
    186
    187 000054  012766  TRAP4:  MOV     #341, 2(SP)      ; SETS CARRY ON TRAP TO 4
                000341
                000002
    188 000062  000002          RTI                      ; ALSO SETS CURRENT PRIORITY HIGH
    189
    190
    191                 ; NOW TEST FROM 10 TO TOP OF AVAILABLE CONTIGUOUS MEMORY
    192                 ; INIT VECTORS AND SET LOW TEST LIMIT TO 10
    193 000064  012701  MEMHGH: MOV     #4, R1           ; SET LOW MEM POINTER
                000004
    194 000070  010421          MOV     R4, (R1)+        ; LOAD TRAP VECTOR
    195 000072  010021          MOV     R0, (R1)+        ; LOAD TRAP PSW VALUE = 340
    196 000074  010102          MOV     R1, R2           ; INIT TO LOW MEMORY = 10
    197
    198                 ; FIND TOP OF AVAILABLE MEMORY
    199 000076  005022  2$:     CLR     (R2)+            ; FIND TOP OF MEMORY
    200 000100  103403           BCS     4$              ; CARRY SET BY TRAP TO 4
    201 000102  020227          CMP     R2, #160000      ; AT END OF PDP-11 ADDR SPACE?
                160000
    202 000106  103773           BLO    2$
    203 000110  005042  4$:     CLR     -(R2)            ; SET POINTER TO LAST LOCATION+2
    204 000112  004567          JSR     R5, MEMCHK       ; TEST TO TOP OF MEMORY
                000144
    205
    206 000116  005000  CHKDEV: CLR     R0               ; INIT UNIT # FOR LATER
    207 000120  011001          MOV     (R0), R1         ; GET R*CS POINTER
    208 000122  105737          TSTB    @#TKS            ; HAS A UNIT # BEEN TYPED
                177560
    209 000126  100004           BPL    10$              ; NO - DEFAULT TO 0
    210 000130  013700            MOV   @#TKB, R0        ; ELSE USE LOW 2 BITS AS UNIT #
                177562
    211                 ;         MOV   R0, @#TPB        ; ECHOE CHAR IF TYPED
    212 000134  042700            BIC   #^C3, R0         ; CLEAR ALL BUT UNIT BITS
                177774
    213 000140  032701  10$:    BIT     #1000, R1        ; RX02 OR RL01 DEVICE?
                001000
    214 000144  001130           BNE    RXFLEM           ; BOOT VIA RX02 MODE IF 1000 BIT SET IN R*CS
```

Table 4-10. DSD 880x/8 Bootstrap Program Listing (Cont) (PDP 11/45 Modification Version)

```
DSD 880-TAURUS BOOTSTRAP PROM   MACRO M1113  06-JAN-82 15:59  PAGE 3
RL COMPATIBLE BOOT

                        .SBTTL  RL COMPATIBLE BOOT
                        ; BOOT USING RL01 PROTOCOL (UNOTE 063)
                        ; DISPATCH WITH R0 = UNIT #, R1 = RLCS

000146  105711  BOOTRL: TSTB    (XCS)               ; CHECK CONTROLLER READY
000150  103777          BCS     .                   ; HANG IF NO BUS RESPONSE TO DEVICE
000152  100375          BPL     BOOTRL              ; ELSE WAIT FOR CONTROLLER RDY
000154  012761          MOV     #13, .RLDA(XCS) ; DO RESET CONTROLLER ON GET STATUS
        000013
        000004
000162  012711          MOV     #4,    (XCS)        ;RLCS - LOAD GET STATUS FUNCTION
        000004
000166  105711          TSTB           (XCS)        ; WAIT FOR CONTROLLER READY
000170  100376          BPL     .-2
000172  012761          MOV     #177601,.RLDA(XCS) ; SET MAXIMAL LENGTH SEEK OUTWARDS
        177601
        000004
000200  012703          MOV     #6*400, R3          ; SEEK COMMAND
        003000
000204  050003          BIS     R0, R3              ; WITH UNIT BITS
000206  000303          SWAB    R3                  ; BACK TO UN  UN CR IE  DF DF FN  FN FN (
000210  010311          MOV     R3,    (XCS)        ; LOAD RL01 SEEK COMMAND
000212  105711          TSTB           (XCS)        ; RLCS - WAIT FOR CONTROLLER READY
000214  100376          BPL     .-2
000216  012761          MOV     #-400,.RLMP(R1) ;RLWC - SET WORDCOUNT FOR 1 BLOCK
        177400
        000006
000224  005061          CLR     .RLDA(R1)           ; LOAD A ZERO INTO DISK ADDRESS REG
        000004
000230  062703          ADD     #6, R3              ; MAKE SEEK INTO A READ COMMAND
        000006
000234  010311          MOV     R3,  (XCS)          ; ISSUE READ FUNCTION
000236  105711          TSTB    (XCS)               ; CONTROLLER READY?
000240  100376          BPL     .-2

000242  005711  CHKNOP: TST     (XCS)               ; ERROR?
000244  100001          BPL     .+4
000246  000000          HALT
000250  023727          CMP     @#0, #240           ; LOC 0 MUST BE NOP
        000000
        000240
000256  001317          BNE     CHKDEV              ; CHECK IF DIFFERENT UNIT #
000260  005007          CLR     PC                  ; DISPATCH TO LOC 0.
```

Table 4-10. DSD 880x/8 Bootstrap Program Listing (Cont) (PDP 11/45 Modification Version)

```
DSD 880-TAURUS BOOTSTRAP PROM   MACRO M1113  06-JAN-82 15:59  PAGE 4
RL COMPATIBLE BOOT

                        ; ROUTINE TO TEST MEMORY FROM C(R1) = LOW LIMIT
                        ; TO C(R2) = UPPER LIMIT BEYOND TEST
                        ; IF ERROR FOUND HALTS WITH R4 POINTING TO ERROR LOC, OR 2 BEYOND.
                        ; R0 = UNIT #  (UNCHANGED)

 000262  010104  MEMCHK: MOV     R1, R4          ; GET STARTING ADDRESS
 000264  010403  2$:     MOV     R4, R3          ; KILL Z FLAG <MOV R4, (R4)+>
 000266  010324          MOV     R3, (R4)+       ; LOAD CONTENTS = ADDRESS
 000270  020402          CMP     R4, R2          ; AT END OF TEST?
 000272  103774           BLO    2$
 000274  024404  CHKADP: CMP     -(R4), R4       ; CHECK BACK DOWN TO START ADDR
 000276  001401           BEQ    NCKADP
 000300  000000            HALT                  ; STUCK BIT IN DATA OR ADDRESS!!
 000302  020401  NCKADP: CMP     R4, R1
 000304  101373           BHI    CHKADP          ; CONTINUE TILL AT START ADDR

 000306  005124  SETCOM: COM     (R4)+           ; MAKE LOC = ADDR COMPLEMENT
 000310  020402          CMP     R4, R2          ; AT END OF TEST?
 000312  103775           BLO    SETCOM

 000314  010104          MOV     R1, R4          ; START AT BEGINNING
 000316  060414  CHKCOM: ADD     R4, (R4)        ; SHOULD BE ALL 1'S
 000320  005224          INC     (R4)+           ; DATA SHOULD = ALL ZEROES
 000322  001401           BEQ     NCKCOM
 000324  000000            HALT                  ; STUCK DATA BIT IF NO HALT AT +156
 000326  020402  NCKCOM: CMP     R4, R2
 000330  103772           BLO    CHKCOM

 000332  012737          MOV     #'D, @#TPB      ; PRINT A "D" AS PROMPT
         000104
         177566
                        ; SET UP TO LEAVE A PATTERN OF 1 011 001 110 001 111 B ROTATED
                        ; RIGHT INTO 4 SUCCESSIVE WORDS
                        ; USED AS MEM BACKGROUND AND FILL-EMPTY DATA.

 000340  010104          MOV     R1, R4          ; SET INITIAL ADDRESS
 000342  012703  SETPAT: MOV     #131617, R3     ; SET INITIAL PATTERN
         131617
 000346  020402  4$:     CMP     R4, R2          ; END OF ADDRESS RANGE?
 000350  103004           BHIS   CHKPAT          ; GO CHECK DATA IF AT END
 000352  010324          MOV     R3, (R4)+       ; CARRY SET BY CMP INSTRUCTION.
 000354  006203          ASR     R3              ; ROTATE AND LOAD AGAIN
 000356  103773           BCS    4$
 000360  000770           BR     SETPAT

 000362  010104  CHKPAT: MOV     R1, R4          ; SET INITIAL ADDRESS
 000364  012703  CHKPTL: MOV     #131617, R3
         131617
 000370  020324  3$:     CMP     R3, (R4)+       ; DATA OK?
 000372  001401           BEQ    4$
 000374  000000            HALT                  ; PATTERN SENSITIVITY ERROR
 000376  020402  4$:     CMP     R4, R2          ; AT END OF ADDRESS RANGE?
 000400  103003           BHIS   MEMEXT          ; YES - EXIT
 000402  006203          ASR     R3              ; CARRY SET BY CMP INSTRUCTION
 000404  103771           BCS    3$
 000406  000766           BR     CHKPTL
 000410  000205  MEMEXT: RTS     R5
```

Table 4-10. DSD 880x/8 Bootstrap Program Listing (Cont) (PDP 11/45 Modification Version)

```
DSD 880-TAURUS BOOTSTRAP PROM   MACRO M1113  06-JAN-82 15:59  PAGE 5
RL COMPATIBLE BOOT

                       ; DEFNST - DISPLAY RX02 DEFINITIVE STATUS STARTING AT LOC 0
                       ; PROCEDE WILL RETRY

    000412  012711  DEFNST: MOV     #17, (XCS)       ; DO DEFINITIVE ERROR STATUS
            000017
    000416  105711  DEFNWT: TSTB    (XCS)            ; WAIT FOR TRREQ OR DONE
    000420  100376           BPL    DEFNWT           ; WAIT FOR TRANSFER REQUEST
    000422  005012            CLR   (XDB)            ; STATUS UPWARDS FROM LOAD ADDR
    000424  000000          HALT                     ; DEFINITIVE STATUS AT LOC 0
                    ;        BR     CHKDEV           ; ACCEPT UNIT AGAIN ON PROCEDE


                       ; FILL EMPTY TEST - DONE AT MULTIPLE BUFFER ADDRESSES IN ORDER
                       ; TO TOGGLE ALL ADDRESS BITS IN SYSTEM MEMORY
    000426  004567  RXFLEM: JSR     R5, FILEMP       ; DO FILL-EMPTY BUFFER TEST
            000016
    000432  000034          10+<5*4>         ; START FILL AT BEGINNING OF
    000434  017700          10+<5*1624.>     ; PATTERN REPETITION LEFT BY RAM TEST
    000436  037676          10+<5*3262.>     ; DO DMA TEST ACROSS ALL ADDRESS BITS
    000440  077704          10+<5*6540.>     ; THAT CAN BE SET IN AVAILABLE MEMORY
    000442  000000           0    ; ADDRESS TERMINATOR
    000444     007          .BYTE   7, 27    ; COMMAND SET BITS FOR UNIT 0,1,2,3
    000445     027
    000446     047          .BYTE   47,67    ; PROTECT AGAINST HIGH UNITS
    000447     067

                       ; NOTE - FILEMP DOES NOT RETURN BUT FLOWS THROUGH INTO BOOTRX

                       ; FILL - EMPTY BUFFER TEST

    000450  012504  FILEMP: MOV     (R5)+, R4        ; GET BUFFER ADDRESS
    000452  001464           BEQ    BOOTRX           ; GO BOOT UNIT IN R0
    000454  005764          TST     404(R4)          ; DOES MEMORY EXIST?
            000404
    000460  103773           BCS    FILEMP           ; NO - STEP TO END OF LIST
    000462  010102          MOV     XCS, XDB         ; INIT FOR RXDB
    000464  004767          CALL    WTFLAG           ; WAIT FOR DONE FLAG UP
            000300
    000470  103777           BCS    .                ; LOOP IF NO BUS RESPONSE
    000472  005711          TST     (R1)             ; RX02 ERROR SET?
    000474  100001           BPL    .+4              ; HALT IF ERROR
    000476  000000            HALT                   ; INTERFACE SETUP ERROR

                       ; DSD880 - RX02 INTERFACE LATCHED BIT TEST

    000500  012722          MOV     #1420, (XDB)+    ; LOAD INTO RXCS
            001420
    000504  022711          CMP     #5460, (XCS)     ; DID THEY LATCH OK?
            005460
    000510  001005           BNE    RXHALT           ; LATCHED BIT COMPARE ERROR
                    ;       CMP     #1420, (XDB)     ; LATCHED OK IN RXDB?
                    ;        BNE    RXHALT           ; NO - BAD INTERFACE.

    000512  012712  RXDBTS: MOV     #173767, (XDB)   ; CHECK RXDB LATCH
            173767
    000516  022712          CMP     #173767, (XDB)   ; DID THEY LATCH
            173767
```

Table 4-10. DSD 880x/8 Bootstrap Program Listing (Cont) (PDP 11/45 Modification Version)

```
DSD 880-TAURUS BOOTSTRAP PROM   MACRO M1113  06-JAN-82 15:59  PAGE 5-1
RL COMPATIBLE BOOT

000522  001401             BEQ    RXFIEM           ; DO FILL-EMPTY DATA TEST
000524  000000  RXHALT:    HALT                    ; HALT IF INCORRECT BIT LATCHUP

000526  010102  RXFIEM: MOV     XCS, XDB         ; SET UP RXDB POINTER
000530  012722          MOV     #401, (XDB)+     ; DO FILL COMMAND
        000401
000534  105711          TSTB    (XCS)            ; WAIT FOR TRREQ
000536  100376           BPL    .-2
000540  012712            MOV   #200, (XDB)      ; WORDCOUNT (=200)
        000200
000544  105711          TSTB    (XCS)            ; WAIT FOR TRREQ
000546  100376           BPL    .-2
000550  010412            MOV   R4, (XDB)        ; BUFFER ADDR
000552  004767          CALL    WTFLAG           ; WAIT FOR DONE OR TRREQ
        000212

                ; NOW EMPTY SECTOR BUFFER AND CHECK DATA VALIDITY

000556  022424  EMPBFT: CMP     (R4)+, (R4)+     ; BUMP EMPTY BUFFER ADDR
                                                 ; SO ERROR IF NO DATA TRANSFER.
000560  012711          MOV     #403, (XCS)      ; DO EMPTY BUFFER COMMAND
        000403
000564  010403          MOV     R4, R3           ; SAVE BUFFER START ADDRESS
000566  105711          TSTB    (XCS)            ; WAIT FOR TRREQ
000570  100376           BPL    .-2
000572  012712            MOV   #200, (XDB)      ; LOAD WORD COUNT
        000200
000576  105711          TSTB    (XCS)            ; WAIT FOR TRREQ
000600  100376           BPL    .-2
000602  010412            MOV   R4, (XDB)        ; AND FILL BUFFER ADDR+2
000604  004767          CALL    WTFLAG           ; WAIT FOR ERROR, DONE OR TRREQ
        000160

000610  010402  CHKEMP: MOV     R4, R2           ; SET UP UPPER CHECK LIMIT
000612  062702          ADD     #400, R2         ; SET R2 = END ADDR TO CHECK
        000400
000616  004567          JSR     R5, CHKPTL       ; DO DATA CHECK
        177542
000622  000712          BR      FILEMP           ; DO NEXT FILL-EMPTY
```

Table 4-10. DSD 880x/8 Bootstrap Program Listing (Cont) (PDP 11/45 Modification Version)

```
DSD 880-TAURUS BOOTSTRAP PROM   MACRO M1113  06-JAN-82 15:59  PAGE 6
RL COMPATIBLE BOOT

                          ; BOOT THE DEVICE IN R1, REGISTERS USED AS INDICATED BELOW
                                         ; R0      LOGICAL UNIT #
               000001  XCS=     %1       ; R1     POINTER TO RXCS
               000002  XDB=     %2       ; R2     POINTER TO RXDB
                                         ; R3     READ COMMAND VALUE WITH DENSITY BIT
               000004  LDP=     %4       ; R4     LOAD POINTER
               000005  SCT=     %5       ; R5     CURRENT SECTOR # (1, 3, 5, 7)
                                         ; (SP)   WORD COUNT FOR CURRENT DENSITY

        000624 060005  BOOTRX: ADD     R0, R5            ; PTR TO READ UNIT N COMMAND
        000626 111503          MOVB    (R5), R3          ; GET COMMAND FOR UNIT
        000630 005004          CLR     LDP               ; INIT LOAD ADDRESS POINTER
        000632 012746          MOV     #100, -(SP)       ; SET LOW DENSITY WORDCOUNT
               000100
        000636 012705          MOV     #1, SCT           ; INIT SECTOR TO READ
               000001

        000642 004767  RDLP:   CALL    WTFLAG            ; WAIT FOR DONE FLAG SET?
               000122
        000646 010102          MOV     XCS, XDB          ; COPY RXCS POINTER
        000650 010322          MOV     R3, (XDB)+        ; LOAD READ COMMAND AND BUMP XDB TO RXDB
        000652 105711          TSTB    (XCS)             ; WAIT FOR TRREQ
        000654 100376           BPL    .-2
        000656 010512          MOV     SCT, (XDB)        ; LOAD SECTOR
        000660 105711          TSTB    (XCS)
        000662 100376           BPL    .-2
        000664 012712          MOV     #1, (XDB)         ; LOAD TRACK
               000001
        000670 004767          CALL    WTFLAG            ; WAIT FOR DONE
               000074
        000674 005711          TST     (XCS)             ; CLUDGE SINCE DEC RX02 SETS ERROR
                                                         ; BEFORE IT SETS DONE
        000676 100010           BPL    EMPBUF            ; EMPTY IF NO ERROR
        000700 032712            BIT   #20, (XDB)        ; IS ERROR A DENSITY ERROR?
               000020
        000704 001642             BEQ  DEFNST            ; NO- DO DEFINITIVE STATUS
        000706 052703              BIS #400, R3          ; SET COMMAND TO DOUBLE DENSITY
               000400
        000712 012716              MOV #200, (SP)        ; SET TO D.D. WORD COUNT
               000200
        000716 000751               BR RDLP              ; AND TRY READING AGAIN

        000720 010346  EMPBUF: MOV     R3, -(SP)         ; GET COMMAND COPY
        000722 042716          BIC     #4, (SP)          ; MAKE INTO AN EMPTY BUFFER COMMAND
               000004
        000726 012611          MOV     (SP)+, (XCS)      ; AND EXECUTE
        000730 105711          TSTB    (XCS)             ; WAIT FOR FIRST TRREQ
        000732 100376           BPL    .-2
        000734 011612            MOV   (SP), (XDB)       ; LOAD THE WORD COUNT
        000736 105711          TSTB    (XCS)
        000740 100376           BPL    .-2
        000742 010412            MOV   LDP, (XDB)        ; AND XFER ADDRESS
        000744 004767          CALL    WTFLAG            ; WAIT FOR DONE OR TRREQ
               000020
        000750 061604          ADD     (SP), LDP         ; BUMP LOAD ADDRESS FOR NEXT SECT
        000752 061604          ADD     (SP), LDP         ; ADD ACTUAL BYTE COUNT
        000754 122525          CMPB    (SCT)+, (SCT)+    ; BUMP SECTOR # BY 2
```

Table 4-10. DSD 880x/8 Bootstrap Program Listing (Cont) (PDP 11/45 Modification Version)

```
DSD 880-TAURUS BOOTSTRAP PROM   MACRO M1113  06-JAN-82 15:59  PAGE 6-1
RL COMPATIBLE BOOT

    000756  020427          CMP     LDP, #1000      ; FINISHED LOADING?
            001000
    000762  002727           BLT    RDLP            ; READ NEXT SECTOR

    000764  000167          JMP     CHKNOP          ; CHECK LOC 0 = NOP AND DISPATCH
            177252


                    ; WAIT FOR FLOPPY FLAGS, DONE, ERROR, TRREQ

    000770  032711  WTFLAG: BIT     #240, (XCS)     ; WAIT FOR DONE OR TRREQ
            000240
    000774  001775           BEQ    WTFLAG          ; CAN'T TEST RX02 ERROR HERE
    000776  000207          RETURN

    001000          BOTLST::

            000020'         .END    BOT170
```

Table 4-10. DSD 880x/8 Bootstrap Program Listing (Cont) (PDP 11/45 Modification Version)

```
DSD 880-TAURUS BOOTSTRAP PROM   MACRO M1113  06-JAN-82 15:59  PAGE 6-2
SYMBOL TABLE

BOOTRL  000146R     002 EMPBFT  000556R     002 RXHALT  000524R     002
BOOTRX  000624R     002 EMPBUF  000720R     002 SCT    =%000005
BOTENT  000036R     002 FILEMP  000450R     002 SETCOM  000306R     002
BOTLST  001000RG    002 LDP    =%000004         SETPAT  000342R     002
BOTW00  000000R     002 MEMCHK  000262R     002 TKB    = 177562
BOTW10  000010R     002 MEMEXT  000410R     002 TKS    = 177560
BOT150  000030R     002 MEMHGH  000064R     002 TPB    = 177566
BOT170  000020R     002 NCKADP  000302R     002 TPS    = 177564
CHKADP  000274R     002 NCKCOM  000326R     002 TRAP4   000054R     002
CHKCOM  000316R     002 RDLP    000642R     002 WTFLAG  000770R     002
CHKDEV  000116R     002 RLCS   = 174400         XCS    =%000001
CHKEMP  000610R     002 RXCS   = 177170         XDB    =%000002
CHKNOP  000242R     002 RXDB   = 177172         $PDP11= 000001
CHKPAT  000362R     002 RXDBTS  000512R     002 .RLBA = 000002
CHKPTL  000364R     002 RXFIEM  000526R     002 .RLDA = 000004
DEFNST  000412R     002 RXFLEM  000426R     002 .RLMP = 000006
DEFNWT  000416R     002

. ABS.  000000     000
        000000     001
BOOT    001000     002
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  336 WORDS  ( 2 PAGES)
DYNAMIC MEMORY:  2822 WORDS  ( 10 PAGES)
ELAPSED TIME:  00:00:17
BOT88T,BOT88T=BOT88T
```

Table 4-11. DSD 880x/8 Bootstrap Program Listing (PDP-11/23+ with 22 bit addressing)

```
DSD 880X8,20,30  Q22-BOOTSTRAP  MACRO M1113  22-APR-82 15:41
TABLE OF CONTENTS
```

Table 4-11. DSD 880x/8 Bootstrap Program Listing (Cont) (PDP-11/23+ with 22 bit addressing)

```
DSD 880X8,20,30  Q22-BOOTSTRAP  MACRO M1113  22-APR-82 15:41  PAGE 1

                .TITLE  DSD 880X8,20,30  Q22-BOOTSTRAP  PROM
                ; $PDP11=1       ; DEFINE FOR UNIBUS VERSION
                ; BOT88T.MAC     5-APR-82          ; Q22 AND VT103 UPGRADE
 .SBTTL LSI-11 VERSION
  ; BOOTSTRAP FOR DSD880 FLOPPY / WINCHESTER DISK CONTROLLER
  ; BOOTS EITHER SINGLE OR DOUBLE DENSITY FLOPPIES
  ; ** NOTE ON BOOTING WHILE REAL TIME CLOCK IS ENABLED. **
  ; THIS BOOT CAN BE STARTED WITH A RUNNING REAL TIME CLOCK IN 2 WAYS.
  ; 1)  BY SETTING THE PSW AHEAD OF TIME TO DISABLE INTERRUPTS BY TYPING
  ;     "$S/ 340<CR>" AND "R7/ 773000<CR>" AND HITTING "P".
  ; 1)  (LSI-11 /2 ONLY) BY ENSURING THAT THE STACK IS POINTING TO
  ;     NON-EXISTANT MEMORY THUS FORCING A DOUBLE BUS ERROR ON ANY
  ;     INTERRUPT AND BY TYPING
  ;    "773000G"  AND TYPING "P" IF HALTS OCCUR DUE TO ATTEMPTED INTERRUPTS.

  ; THE BOOTSTRAP PROCEEDS IN 4 STEPS
  ; 1)     SELECT DEVICE   DETERMINES DEVICE TO BE BOOTED
  ; 2)     RAM TEST        CHECKS ALL AVAILABLE MEMORY FOR STUCK BITS
  ;                        ON BOTH DATA AND ADDRESS LINES. <0-30K>
  ;                        DOES BOTH DATA = ADDRESS AND PATTERN TESTS
  ;                        1) CLEARS MEMORY TO 0'S AND SIZES MEMORY
  ;                        2) LOADS MEMORY = ADDRESS AND CHECKS
  ;                        3) LOADS MEMORY = ADDRESS COMPLEMENT, CHECKS
  ;                        4) LOADS MEMORY WITH THE REPEATING PATTERN OF
  ;                           131617, 154707, 166343, 173161, 175470
  ; CHECK FOR UNIT 0 OVERRIDE FROM KEYBOARD
  ; 3-WINCHESTER  READ IN BLOCK 0, START AT LOC 0.
  ; 3-DY  FILL-EMPTY       CHECKS DSD880 - PROCESSOR DATA PATH FOR
  ;                        SYNTAX AND DATA ERRORS.  ALSO INSURE'S ALL
  ;                        AVAILABLE ADDRESS LINES TOGGLE UNDER DMA.
  ;                        CHECKS FILL-EMPTY WITH BUFFERS AT 774,
  ;                        17700, 37676, 77704, 137700 IF MEMORY EXISTS.
  ; 4-DY  BOOTSTRAP        READS IN BLOCK 0 FROM DISKETTE IN CORRECT
  ;                        DENSITY AND STARTS AT LOC 0

  ; ERROR HALTS OR HANG UP LOOPS  (ADDRESSES RELATIVE TO BOOT BASE ADDR)
  ; 154-160 LOOP  RL CONTROLLER NOT READY
  ; 156    HANG   RL CONTROLLER NOT RESPONDING AT ADDRESS
  ; 174-176, 220-222, 244-246      RL TYPE CONTROLLER HUNG
  ; 254   HALT    ERROR DURING READ BLOCK 0
  ; 306   HALT    MEMORY ERROR AT LOC (R4), CONTENTS SHOULD EQUAL ADDRESS
  ; 332      HALT  MEMORY ERROR AT -2(R4), EXPECT 0, CONTENTS=ADDRESS COMPLEMENTED
  ; 374      HALT  1) FILL-EMPTY ERROR IF R5=BOOT+622
  ;                2) MEMORY ERR IF R5=BOOT+116
  ; 424   HALT    ERROR ON FLOPPY, DEFINITIVE STATUS AT LOC 0 IN MEMORY
  ; 470      LOOP   DY DEVICE ADDRESS SELECTED FOR BOOTING DOESN'T RESPOND
  ; 500      HALT  ERROR FLAG IN RXCS SET AFTER INIT
  ; 524      HALT  RXCS OR RXDB INTERFACE REGISTER STUCK BIT PROBLEM
  ;                NOTE CONTENTS OF RXCS, RXDB <5460,1420>,<*,173767>
  ; 534-536, 544-546      TRANSFER REQUEST HANGUP  (FILL BUFFER)
  ; 566-570, 576-600      TRANSFER REQUEST HANGUP  (EMPTY BUFFER)

  ; 652-654, 660-662      TRANSFER REQUEST HANGUP  (DY-READ-BOOTSTRAP)
  ; 730-732, 740-742      TRANSFER REQUEST HANGUP  (DY-EMPTY-BOOTSTRAP)
  ; 770-774  LOOP DSD880 FLAG WAIT ROUTINE HANGUP
```

Table 4-11. DSD 880x/8 Bootstrap Program Listing (Cont) (PDP-11/23+ with 22 bit addressing)

```
DSD 880X8,20,30  Q22-BOOTSTRAP  MACRO M1113  22-APR-82 15:41  PAGE 1-1
LSI-11 VERSION

        ; START ADDRESSES
        ; BOOT+0    (TYPICALLY 173000)  BOOTS RL DEVICE WITH RLCS AT 174400
        ; BOOT+10   (TYPICALLY 173010)  BOOTS RL DEVICE WITH RLCS AT 174420
        ; BOOT+20   (TYPICALLY 173020)  BOOTS DY DEVICE WITH RXCS AT 177170
        ; BOOT+30   (TYPICALLY 173030)  BOOTS DY DEVICE WITH RXCS AT 177150
        ; BOOT+36   (TYPICALLY 173036)  GENERAL DEVICE ENTRANCE - USER
        ;                SET'S  LOCATION 0 = DESIRED RLCS OR RXCS
        ;                NOTE: THE BIT OF VALUE 1000 MUST BE SET FOR RX BOOTING
        ;                        IF REAL TIME CLOCK MUST BE LEFT ON THEN SET
        ;                        $S/ 340<CR> AND R7/ 1730B0<CR> AND PROCEED

        ; A "BOOT" ON AN 11/04 OR 11/34 PRINTS R0, R4, SP, R7 ON THE TERMINAL.
        ; IF AN ERROR HALT OCCURS AT BOOT+424 WHILE BOOTING THEN
        ; BOOTING AGAIN ON AN 11/04 OR /34 PRINTS OUT THE FOLLOWING.
        ;       R0 = CURRENT DRIVE # BEING BOOTED FROM.
        ;       R4 = LOAD ADDRESS WHERE ERROR OCCURRED
        ;       SP = DEFINITIVE STATUS OF ERROR
        ;       R7 = ERROR HALT ADDR+2

        ; NOTE - A HALT OR HANGUP OCCURRING BETWEEN 770-774 THAT WILL NOT
        ;    RESPOND TO BREAK OR HALT IS GENERALLY DUE TO LACK OF DMA GRANT
        ;    CONTINUITY ON THE BUS. USER SHOULD PUT DSD880 INTERFACE CARD
        ;    CLOSER TO THE PROCESSOR AND ENSURE GRANT CONTINUITY.

        ; DSD880 - RX02 REGISTER SYNTAX DEFS
        RXCS=   177170
        ; ERR  INI XM XM  X02 ?? SID  DEN TRQ IEN  DON UN1 FUN  FUN FUN GO
               ;       100000   ; ERR   ERROR FLAG
               ;        40000   ; INI   LOAD INTO RXCS TO INITIALIZE
               ;        30000   ; XM    EXTENDED MEMORY SELECT BITS
               ;        4000    ; X02   = 1 FOR RX02 MODE SYNTAX
               ;        400     ; DEN   SET = 1 FOR DOUBLE DENSITY
               ;        200     ; TRQ   TRANSFER REQUEST - DATA TO/FROM RXDB
               ;        16      ; FUN   FUNCTION <0-7> - SET "GO" TO EXEC

        RXDB=RXCS+2              ; RXES ERROR BIT LAYOUT
        ;         NXM  WCV SID   DRV DRV DEL   DSK DEN ACL   INT SID CRC
        ;              OVF  #1    #1 RDY DAT   DEN ERR LOW   DON RDY ERR

        ; REGISTER USAGE IN BOOTRX SECTION
        XCS=    %1      ; R1    POINTER TO RXCS
        XDB=    %2      ; R2    POINTER TO RXDB
                        ; R3    READ COMMAND VAL WITH DENSITY BIT
        LDP=    %4      ; R4    LOAD POINTER
        SCT=    %5      ; R5    CURRENT SECTOR # (1, 3, 5, 7)
                        ; (SP)  WORD COUNT FOR CURRENT DENSITY

        TKS=    177560  ; CONSOLE STATUS REGESTER DEFS FOR GETTING UNIT #
        TKB=    TKS+2
```

Table 4-11. DSD 880x/8 Bootstrap Program Listing (Cont) (PDP-11/23+ with 22 bit addressing)

```
DSD 880X8,20,30  Q22-BOOTSTRAP  MACRO M1113  22-APR-82 15:41  PAGE 2
LSI-11 VERSION

        ; RL01 / RL02 COMPATIBLE HARDWARE DEFS.
        RLCS=    174400   ; RL COMMAND STATUS REGISTER
        ; ERR  DE  NXM DLT  DCRC OPI DS1  DS0 CRDY IE  A17 A16 F2  F1 F0 DRDY
        ;              HNF  HCRC OPI
        ; RO   RO  RO  RO   RO   RO  R/W  R/W R/W  R/W R/W R/W RW  RW RW RO
        ; 15   14  13  12   11   10  09   08  07   06  05  04  03  02 01 00
        ;
        ; FUNCTIONS      0 0 0   00      NOOP
        ;                0 0 1   02      WRITE CHECK
        ;                0 1 0   04      GET STATUS
        ;                0 1 1   06      SEEK
        ;                1 0 0   10      READ HEADER
        ;                1 0 1   12      WRITE DATA
        ;                1 1 0   14      READ DATA
        ;                1 1 1   16      READ DATA - NO HEADER CHECK

        ; RLBA  = 174402        - BUS ADDRESS REGISTER
         .RLBA  = 2        ; OFFSET

        ; RLDA= 174404  - DISK ADDRESS REGISTER (SEEK)
         .RLDA= 4
        ; DF8  DF7 DF6 DF5  DF4 DF3 DF2  DF1 DF0 000  000 HS  000 DIR 000 001
        ;      DF7 - DF0       CYCLINDER DIFFERENCE TO SEEK
        ;      HS              SET = LOWER SIDE, CLEAR = UPPER
        ;      DIR             SET = SEEK INWARDS TOWARD SPINDLE
        ;                      CLR = SEEK OUTWARDS

        ; RLDA= 174404  - DISK ADDRESS DURING READ/WRITE DATA COMMANDS
        ; CA8  CA7 CA6 CA5  CA4 CA3 CA2  CA1 CA0 HS   SA5 SA4 SA3  SA2 SA1 SA0

        ;               - DISK ADDRESS DURING GET STATUS COMMAND
        ; 000  000 000 000  000 000 000  000 000 000  000 000 RST  000 000 000

        ; RLMP= 174406  - MULTI-PURPOSE REGISTER
         .RLMP= 6
        ; WDE  HCE WLK SKTO  SPE WGE VC  DSE 000 HS  CO  HO  BH   ST2 ST1 ST0


                                    ; START HERE FOR RL01 TYPE BOOT - @ 174400
    000000  012737  BOTW00: MOV     #RLCS, @#0        ; DO RL BOOT ON POWER UP
            174400
            000000
    000006  000413          BR      BOTENT

    000010  012737  BOTW10: MOV     #RLCS+20, @#0     ; DO ALTERNATIVE RL BOOT
            174420
            000000
    000016  000407          BR      BOTENT

    000020  012737  BOT170: MOV     #177170, @#0      ; DO STANDARD FLOPPY BOOT
            177170
            000000
    000026  000403          BR      BOTENT

    000030  012737  BOT150: MOV     #177150, @#0      ; DO ALTERNATIVE FLOPPY BOOT
            177150
            000000
```

Table 4-11. DSD 880x/8 Bootstrap Program Listing (Cont) (PDP-11/23+ with 22 bit addressing)

```
DSD 880X8,20,30  Q22-BOOTSTRAP  MACRO M1113  22-APR-82 15:41  PAGE 2-1
LSI-11 VERSION


 000036  011706  BOTENT: MOV     (PC), SP          ; SET STACK TO 12700
 000040  012700          MOV     #340, R0          ; LOCK OUT LINE TIME CLOCK
         000340
 000044  106400          MTPS    R0                ; BY SETTING TO PRIORITY 7.
 000046  000240          NOP                       ; SO SAME SIZE IN PDP-11 VERSION
                                                   ; ABOVE 2 WORDS BECOME
                                                   ; MOV   R0, @#177776

 000050  004467          JSR     R4, MEMHGH        ; GET POINTER TO TRAP ROUTINE
         000010

                 ; TRAP PROCESSOR FOR NON-EXISTANT MEMORY TIMEOUT
                 ; SETS CARRY AND RETURNS ON NON-EXISTANT MEMORY TRAP

 000054  012766  TRAP4:  MOV     #341, 2(SP)       ; SETS CARRY ON TRAP TO 4
         000341
         000002
 000062  000002          RTI                       ; ALSO SETS CURRENT PRIORITY HIGH


                 ; NOW TEST FROM 10 TO TOP OF AVAILABLE CONTIGUOUS MEMORY
                 ; INIT VECTORS AND SET LOW TEST LIMIT TO 10
 000064  012701  MEMHGH: MOV     #4, R1            ; SET LOW MEM POINTER
         000004
 000070  010421          MOV     R4, (R1)+         ; LOAD TRAP VECTOR
 000072  010021          MOV     R0, (R1)+         ; LOAD TRAP PSW VALUE = 340
 000074  010102          MOV     R1, R2            ; INIT TO LOW MEMORY = 10

                 ; FIND TOP OF AVAILABLE MEMORY
 000076  005022  2$:     CLR     (R2)+             ; FIND TOP OF MEMORY
 000100  103403           BCS     4$               ; CARRY SET BY TRAP TO 4
 000102  020227          CMP     R2, #160000       ; AT END OF PDP-11 ADDR SPACE?
         160000
 000106  103773           BLO    2$
 000110  005042  4$:     CLR     -(R2)             ; SET POINTER TO LAST LOCATION+2
 000112  004567          JSR     R5, MEMCHK        ; TEST TO TOP OF MEMORY
         000152

 000116  005000  CHKDEV: CLR     R0                ; INIT UNIT # FOR LATER
 000120  011001          MOV     (R0), R1          ; GET R*CS POINTER
 000122  105737          TSTB    @#TKS             ; HAS A UNIT # BEEN TYPED
         177560
 000126  100007           BPL    10$               ; NO - DEFAULT TO 0
 000130  013700            MOV   @#TKB, R0         ; ELSE USE LOW 2 BITS AS UNIT #
         177562
 000134  042700            BIC   #314, R0          ; CLEAR PARITY,EXTRA BITS.
         000314
 000140  162700            SUB   #'0, R0           ; CHECK FOR VALID UNIT #
         000060
 000144  002764             BLT  CHKDEV            ; IGNORE IF NOT  X X  1 1 X  X U U
 000146  032701  10$:    BIT     #1000, R1         ; RX02 OR RL01 DEVICE?
         001000
 000152  001125           BNE    RXFLEM            ; BOOT VIA RX02 MODE IF 1000 BIT SET IN R*CS
```

Table 4-11. DSD 880x/8 Bootstrap Program Listing (Cont) (PDP-11/23+ with 22 bit addressing)

```
DSD 880X8,20,30  Q22-BOOTSTRAP  MACRO M1113  22-APR-82 15:41  PAGE 3
RL COMPATIBLE BOOT

                         .SBTTL  RL COMPATIBLE BOOT
                         ; BOOT USING RL01 PROTOCOL (UNOTE 063)
                         ; DISPATCH WITH R0 = UNIT #, R1 = RLCS

000154  105711  BOOTRL:  TSTB    (XCS)               ; CHECK CONTROLLER READY
000156  103777           BCS     .                   ; HANG IF NO BUS RESPONSE TO DEVICE
000160  100375           BPL     BOOTRL              ; ELSE WAIT FOR CONTROLLER RDY
000162  012761           MOV     #13, .RLDA(XCS)     ; DO RESET CONTROLLER ON GET STATUS
        000013
        000004
000170  012711           MOV     #4,    (XCS)        ;RLCS - LOAD GET STATUS FUNCTION
        000004
000174  105711           TSTB           (XCS)        ; WAIT FOR CONTROLLER READY
000176  100376           BPL     .-2
000200  012761           MOV     #177601,.RLDA(XCS)  ; SET MAXIMAL LENGTH SEEK OUTWARDS
        177601
        000004
000206  012703           MOV     #6*400, R3          ; SEEK COMMAND
        003000
000212  050003           BIS     R0, R3              ; WITH UNIT BITS
000214  000303           SWAB    R3                  ; BACK TO UN  UN CR IE  DF DF FN  FN FN GO
000216  010311           MOV     R3,    (XCS)        ; LOAD RL01 SEEK COMMAND
000220  105711           TSTB           (XCS)        ; RLCS - WAIT FOR CONTROLLER READY
000222  100376           BPL     .-2
000224  012761           MOV     #-400,.RLMP(R1)     ;RLWC - SET WORDCOUNT FOR 1 BLOCK
        177400
        000006
000232  005061           CLR     .RLDA(R1)           ; LOAD A ZERO INTO DISK ADDRESS REG
        000004
000236  062703           ADD     #6, R3              ; MAKE SEEK INTO A READ COMMAND
        000006
000242  010311           MOV     R3,  (XCS)          ; ISSUE READ FUNCTION
000244  105711           TSTB    (XCS)               ; CONTROLLER READY?
000246  100376           BPL     .-2

000250  005711  CHKNOP:  TST     (XCS)               ; ERROR?
000252  100001           BPL     .+4
000254  000000           HALT
000256  023727           CMP     @#0, #240           ; LOC 0 MUST BE NOP
        000000
        000240
000264  001314           BNE     CHKDEV              ; CHECK IF DIFFERENT UNIT #
000266  005007           CLR     PC                  ; DISPATCH TO LOC 0.
```

Table 4-11. DSD 880x/8 Bootstrap Program Listing (Cont) (PDP-11/23+ with 22 bit addressing)

```
DSD 880X8,20,30  Q22-BOOTSTRAP  MACRO M1113  22-APR-82 15:41  PAGE 4
RL COMPATIBLE BOOT

                        ; ROUTINE TO TEST MEMORY FROM C(R1) = LOW LIMIT
                        ; TO C(R2) = UPPER LIMIT BEYOND TEST
                        ; IF ERROR FOUND HALTS WITH R4 POINTING TO ERROR LOC, OR 2 BEYOND.
                        ; R0 = UNIT #  (UNCHANGED)
000270  010104  MEMCHK: MOV     R1, R4          ; GET STARTING ADDRESS
000272  010403  2$:     MOV     R4, R3          ; KILL Z FLAG <MOV R4, (R4)+>
000274  010324          MOV     R3, (R4)+       ; LOAD CONTENTS = ADDRESS
000276  020402          CMP     R4, R2          ; AT END OF TEST?
000300  103774           BLO    2$
000302  024404  CHKADP: CMP     -(R4), R4       ; CHECK BACK DOWN TO START ADDR
000304  001401           BEQ    NCKADP
000306  000000            HALT                  ; STUCK BIT IN DATA OR ADDRESS!!
000310  020401  NCKADP: CMP     R4, R1
000312  101373           BHI    CHKADP          ; CONTINUE TILL AT START ADDR
000314  005124  SETCOM: COM     (R4)+           ; MAKE LOC = ADDR COMPLEMENT
000316  020402          CMP     R4, R2          ; AT END OF TEST?
000320  103775           BLO    SETCOM
000322  010104          MOV     R1, R4          ; START AT BEGINNING
000324  060414  CHKCOM: ADD     R4, (R4)        ; SHOULD BE ALL 1'S
000326  005224          INC     (R4)+           ; DATA SHOULD = ALL ZEROES
000330  001401           BEQ     NCKCOM
000332  000000            HALT                  ; STUCK DATA BIT IF NO HALT AT +156
000334  020402  NCKCOM: CMP     R4, R2
000336  103772           BLO    CHKCOM
                        ;       MOV     #'D, @#TPB      ; PRINT A "D" AS PROMPT
                        ; SET UP TO LEAVE A PATTERN OF 1 011 001 110 001 111 B ROTATED
                        ; RIGHT INTO 4 SUCCESSIVE WORDS
                        ; USED AS MEM BACKGROUND AND FILL-EMPTY DATA.
000340  010104          MOV     R1, R4          ; SET INITIAL ADDRESS
000342  012703  SETPAT: MOV     #131617, R3     ; SET INITIAL PATTERN
        131617
000346  020402  4$:     CMP     R4, R2          ; END OF ADDRESS RANGE?
000350  103004           BHIS   CHKPAT          ; GO CHECK DATA IF AT END
000352  010324          MOV     R3, (R4)+       ; CARRY SET BY CMP INSTRUCTION.
000354  006203          ASR     R3              ; ROTATE AND LOAD AGAIN
000356  103773           BCS    4$
000360  000770           BR     SETPAT
000362  010104  CHKPAT: MOV     R1, R4          ; SET INITIAL ADDRESS
000364  012703  CHKPTL: MOV     #131617, R3
        131617
000370  020324  3$:     CMP     R3, (R4)+       ; DATA OK?
000372  001401           BEQ    4$
000374  000000            HALT                  ; PATTERN SENSITIVITY ERROR
000376  020402  4$:     CMP     R4, R2          ; AT END OF ADDRESS RANGE?
000400  103003           BHIS   MEMEXT          ; YES - EXIT
000402  006203          ASR     R3              ; CARRY SET BY CMP INSTRUCTION
000404  103771           BCS    3$
000406  000766           BR     CHKPTL
000410  000205  MEMEXT: RTS     R5
```

Table 4-11. DSD 880x/8 Bootstrap Program Listing (Cont)
(PDP-11/23+ with 22 bit addressing)

```
DSD 880X8,20,30  Q22-BOOTSTRAP  MACRO M1113  22-APR-82 15:41  PAGE 5
RL COMPATIBLE BOOT

                        ; DEFNST - DISPLAY RX02 DEFINITIVE STATUS STARTING AT LOC 0
                        ; PROCEDE WILL RETRY

    000412  012711  DEFNST: MOV     #17, (XCS)      ; DO DEFINITIVE ERROR STATUS
            000017
    000416  105711  DEFNWT: TSTB    (XCS)           ; WAIT FOR TRREQ OR DONE
    000420  100376          BPL     DEFNWT          ; WAIT FOR TRANSFER REQUEST
    000422  005012          CLR     (XDB)           ; STATUS UPWARDS FROM LOAD ADDR
    000424  000000          HALT                    ; DEFINITIVE STATUS AT LOC 0
                        ;       BR      CHKDEV          ; ACCEPT UNIT AGAIN ON PROCEDE


                        ; FILL EMPTY TEST - DONE AT MULTIPLE BUFFER ADDRESSES IN ORDER
                        ; TO TOGGLE ALL ADDRESS BITS IN SYSTEM MEMORY
    000426  004567  RXFLEM: JSR     R5, FILEMP      ; DO FILL-EMPTY BUFFER TEST
            000016
    000432  000034          10+<5*4>        ; START FILL AT BEGINNING OF
    000434  017700          10+<5*1624.>    ; PATTERN REPETITION LEFT BY RAM TEST
    000436  037676          10+<5*3262.>    ; DO DMA TEST ACROSS ALL ADDRESS BITS
    000440  077704          10+<5*6540.>    ; THAT CAN BE SET IN AVAILABLE MEMORY
    000442  000000          0       ; ADDRESS TERMINATOR
    000444     007          .BYTE   7, 27   ; COMMAND SET BITS FOR UNIT 0,1,2,3
    000445     027
    000446     047          .BYTE   47,67   ; PROTECT AGAINST HIGH UNITS
    000447     067

                        ; NOTE - FILEMP DOES NOT RETURN BUT FLOWS THROUGH INTO BOOTRX

                        ; FILL - EMPTY BUFFER TEST

    000450  012504  FILEMP: MOV     (R5)+, R4       ; GET BUFFER ADDRESS
    000452  001464          BEQ     BOOTRX          ; GO BOOT UNIT IN R0
    000454  005764          TST     404(R4)         ; DOES MEMORY EXIST?
            000404
    000460  103773          BCS     FILEMP          ; NO - STEP TO END OF LIST
    000462  010102          MOV     XCS, XDB        ; INIT FOR RXDB
    000464  004767          CALL    WTFLAG          ; WAIT FOR DONE FLAG UP
            000300
    000470  103777          BCS     .               ; LOOP IF NO BUS RESPONSE
    000472  005711          TST     (R1)            ; RX02 ERROR SET?
    000474  100001          BPL     .+4             ; HALT IF ERROR
    000476  000000          HALT                    ; INTERFACE SETUP ERROR

                        ; DSD880 - RX02 INTERFACE LATCHED BIT TEST

    000500  012722          MOV     #1420, (XDB)+   ; LOAD INTO RXCS
            001420
    000504  022711          CMP     #5460, (XCS)    ; DID THEY LATCH OK?
            005460
    000510  001005          BNE     RXHALT          ; LATCHED BIT COMPARE ERROR
                        ;       CMP     #1420, (XDB)    ; LATCHED OK IN RXDB?
                        ;       BNE     RXHALT          ; NO - BAD INTERFACE.

    000512  012712  RXDBTS: MOV     #173767, (XDB)  ; CHECK RXDB LATCH
            173767
    000516  022712          CMP     #173767, (XDB)  ; DID THEY LATCH
            173767
```

Table 4-11. DSD 880x/8 Bootstrap Program Listing (Cont)
(PDP-11/23+ with 22 bit addressing)

```
DSD 880X8,20,30  Q22-BOOTSTRAP  MACRO M1113  22-APR-82 15:41  PAGE 5-1
RL COMPATIBLE BOOT

    000522  001401              BEQ     RXFIEM           ; DO FILL-EMPTY DATA TEST
    000524  000000  RXHALT:     HALT                     ; HALT IF INCORRECT BIT LATCHUP

    000526  010102  RXFIEM: MOV     XCS, XDB         ; SET UP RXDB POINTER
    000530  012722          MOV     #401, (XDB)+     ; DO FILL COMMAND
            000401
    000534  105711          TSTB    (XCS)            ; WAIT FOR TRREQ
    000536  100376           BPL    .-2
    000540  012712            MOV   #200, (XDB)      ; WORDCOUNT (=200)
            000200
    000544  105711          TSTB    (XCS)            ; WAIT FOR TRREQ
    000546  100376           BPL    .-2
    000550  010412            MOV   R4, (XDB)        ; BUFFER ADDR
    000552  004767          CALL    WTFLAG           ; WAIT FOR DONE OR TRREQ
            000212


                    ; NOW EMPTY SECTOR BUFFER AND CHECK DATA VALIDITY

    000556  022424  EMPBFT: CMP     (R4)+, (R4)+     ; BUMP EMPTY BUFFER ADDR
                                                     ; SO ERROR IF NO DATA TRANSFER.
    000560  012711          MOV     #403, (XCS)      ; DO EMPTY BUFFER COMMAND
            000403
    000564  010403          MOV     R4, R3           ; SAVE BUFFER START ADDRESS
    000566  105711          TSTB    (XCS)            ; WAIT FOR TRREQ
    000570  100376           BPL    .-2
    000572  012712            MOV   #200, (XDB)      ; LOAD WORD COUNT
            000200
    000576  105711          TSTB    (XCS)            ; WAIT FOR TRREQ
    000600  100376           BPL    .-2
    000602  010412            MOV   R4, (XDB)        ; AND FILL BUFFER ADDR+2
    000604  004767          CALL    WTFLAG           ; WAIT FOR ERROR, DONE OR TRREQ
            000160

    000610  010402  CHKEMP: MOV     R4, R2           ; SET UP UPPER CHECK LIMIT
    000612  062702          ADD     #400, R2         ; SET R2 = END ADDR TO CHECK
            000400
    000616  004567          JSR     R5, CHKPTL       ; DO DATA CHECK
            177542
    000622  000712          BR      FILEMP           ; DO NEXT FILL-EMPTY
```

Table 4-11. DSD 880x/8 Bootstrap Program Listing (Cont) (PDP-11/23+ with 22 bit addressing)

```
DSD 880X8,20,30  Q22-BOOTSTRAP  MACRO M1113  22-APR-82 15:41  PAGE 6
RL COMPATIBLE BOOT

                              ; BOOT THE DEVICE IN R1, REGISTERS USED AS INDICATED BELOW
                                        ; R0      LOGICAL UNIT #
             000001  XCS=     %1        ; R1    POINTER TO RXCS
             000002  XDB=     %2        ; R2    POINTER TO RXDB
                                        ; R3    READ COMMAND VALUE WITH DENSITY BIT
             000004  LDP=     %4        ; R4    LOAD POINTER
             000005  SCT=     %5        ; R5    CURRENT SECTOR # (1, 3, 5, 7)
                                        ; (SP)  WORD COUNT FOR CURRENT DENSITY
    000624   060005  BOOTRX: ADD     R0, R5          ; PTR TO READ UNIT N COMMAND
    000626   111503          MOVB    (R5), R3        ; GET COMMAND FOR UNIT
    000630   005004          CLR     LDP             ; INIT LOAD ADDRESS POINTER
    000632   012746          MOV     #100, -(SP)     ; SET LOW DENSITY WORDCOUNT
             000100
    000636   012705          MOV     #1, SCT         ; INIT SECTOR TO READ
             000001
    000642   004767  RDLP:   CALL    WTFLAG          ; WAIT FOR DONE FLAG SET?
             000122
    000646   010102          MOV     XCS, XDB        ; COPY RXCS POINTER
    000650   010322          MOV     R3, (XDB)+      ; LOAD READ COMMAND AND BUMP XDB TO RXDB
    000652   105711          TSTB    (XCS)           ; WAIT FOR TRREQ
    000654   100376           BPL    .-2
    000656   010512          MOV     SCT, (XDB)      ; LOAD SECTOR
    000660   105711          TSTB    (XCS)
    000662   100376           BPL    .-2
    000664   012712          MOV     #1, (XDB)       ; LOAD TRACK
             000001
    000670   004767          CALL    WTFLAG          ; WAIT FOR DONE
             000074
    000674   005711          TST     (XCS)           ; CLUDGE SINCE DEC RX02 SETS ERROR
                                                     ; BEFORE IT SETS DONE
    000676   100010           BPL    EMPBUF          ; EMPTY IF NO ERROR
    000700   032712             BIT  #20, (XDB)      ; IS ERROR A DENSITY ERROR?
             000020
    000704   001642               BEQ DEFNST         ; NO- DO DEFINITIVE STATUS
    000706   052703                BIS #400, R3      ; SET COMMAND TO DOUBLE DENSITY
             000400
    000712   012716                MOV #200, (SP)    ; SET TO D.D. WORD COUNT
             000200
    000716   000751                BR  RDLP          ; AND TRY READING AGAIN
    000720   010346  EMPBUF: MOV     R3, -(SP)       ; GET COMMAND COPY
    000722   042716          BIC     #4, (SP)        ; MAKE INTO AN EMPTY BUFFER COMMAND
             000004
    000726   012611          MOV     (SP)+, (XCS)    ; AND EXECUTE
    000730   105711          TSTB    (XCS)           ; WAIT FOR FIRST TRREQ
    000732   100376           BPL    .-2
    000734   011612            MOV   (SP), (XDB)     ; LOAD THE WORD COUNT
    000736   105711          TSTB    (XCS)
    000740   100376           BPL    .-2
    000742   010412            MOV   LDP, (XDB)      ; AND XFER ADDRESS
    000744   004767          CALL    WTFLAG          ; WAIT FOR DONE OR TRREQ
             000020
    000750   061604          ADD     (SP), LDP       ; BUMP LOAD ADDRESS FOR NEXT SECT
    000752   061604          ADD     (SP), LDP       ; ADD ACTUAL BYTE COUNT
    000754   122525          CMPB    (SCT)+, (SCT)+  ; BUMP SECTOR # BY 2
```

Table 4-11. DSD 880x/8 Bootstrap Program Listing (Cont)
(PDP-11/23+ with 22 bit addressing)

```
DSD 880X8,20,30  Q22-BOOTSTRAP  MACRO M1113  22-APR-82 15:41  PAGE 6-1
RL COMPATIBLE BOOT

    000756  020427           CMP     LDP, #1000        ; FINISHED LOADING?
            001000
    000762  002727            BLT    RDLP              ; READ NEXT SECTOR
    000764  000167           JMP     CHKNOP            ; CHECK LOC 0 = NOP AND DISPATCH
            177260

                     ; WAIT FOR FLOPPY FLAGS, DONE, ERROR, TRREQ

    000770  032711   WTFLAG: BIT     #240, (XCS)       ; WAIT FOR DONE OR TRREQ
            000240
    000774  001775            BEQ    WTFLAG            ; CAN'T TEST RX02 ERROR HERE
    000776  000207           RETURN

    001000           BOTLST::

            000020'          .END    BOT170
```

Table 4-11. DSD 880x/8 Bootstrap Program Listing (Cont)
(PDP-11/23+ with 22 bit addressing)

```
DSD 880X8,20,30  Q22-BOOTSTRAP  MACRO M1113  22-APR-82 15:41  PAGE 6-2
SYMBOL TABLE

BOOTRL  000154R     002 DEFNWT  000416R     002 RXFIEM  000526R     002
BOOTRX  000624R     002 EMPBFT  000556R     002 RXFLEM  000426R     002
BOTENT  000036R     002 EMPBUF  000720R     002 RXHALT  000524R     002
BOTLST  001000RG    002 FILEMP  000450R     002 SCT    =%000005
BOTW00  000000R     002 LDP    =%000004         SETCOM  000314R     002
BOTW10  000010R     002 MEMCHK  000270R     002 SETPAT  000342R     002
BOT150  000030R     002 MEMEXT  000410R     002 TKB    = 177562
BOT170  000020R     002 MEMHGH  000064R     002 TKS    = 177560
CHKADP  000302R     002 NCKADP  000310R     002 TRAP4   000054R     002
CHKCOM  000324R     002 NCKCOM  000334R     002 WTFLAG  000770R     002
CHKDEV  000116R     002 RDLP    000642R     002 XCS    =%000001
CHKEMP  000610R     002 RLCS   = 174400         XDB    =%000002
CHKNOP  000250R     002 RXCS   = 177170         .RLBA = 000002
CHKPAT  000362R     002 RXDB   = 177172         .RLDA = 000004
CHKPTL  000364R     002 RXDBTS  000512R     002 .RLMP = 000006
DEFNST  000412R     002

. ABS.  000000     000
        000000     001
BOOT    001000     002
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  321 WORDS  ( 2 PAGES)
DYNAMIC MEMORY:  2822 WORDS .( 10 PAGES)
ELAPSED TIME:  00:00:11
BOT88T,BOT88T/LI:TTM=BOT88T
```

4.6 Off-Line Operation

In addition to normal computer controlled operations, the DSD 880x/8 is capable of various supplemental operations under internal control. These operations include format, reload, backup, and HyperDiagnostics. Table 4-2 gives the mode and class switch settings for selection of the options available under each type of operation.

CAUTION

> To ensure operating system integrity, no attempt to access the DSD 880x/8 from the host computer should be made while using the DSD 880x/8 off-line capabilities.

Performance of a DSD 880x/8 off-line function is achieved by first ensuring no DSD 880x/8 computer controlled operation is taking place, selection of the desired function on the mode and class switches on the DSD 880x/8 control panel, and pushing the Execute button once. At the completion of the selected operation, return the mode and class switches to the desired normal operating mode and press the EXECUTE pushbutton once to return to normal computer controlled operation.

4.6.1 Format Mode

The format mode (mode 1, class 0) is used to format the entire floppy disk in DEC double-density format, or (mode 1, class 1) to format the entire floppy disk in DEC/IBM single-density format.

4.6.2 Backup and Reload Modes

The DSD 880x/8 Data Storage System provides the user with the facility to transfer data between the nonremovable winchester disk and floppy disks without the intervention of a host processor. The resulting backup floppy disks are physical images of the winchester and may be used to regenerate the winchester disk data on the original or any other DSD 880x/8 winchester disk.

Data integrity may be verified by selecting a backup or a reload routine which includes a verify pass. The verify routine will be executed following the reload or backup routine and compares the data on the backup floppy to the data on the winchester. If data does not compare, a 30 error will be reported and the verify routine will terminate.

Backup

Since the backup routine cannot determine the extent of valid data on the winchester disk, it is designed to copy the entire winchester disk onto the backup floppy disks. Each time a backup is initiated a unique version number is recorded on the backup floppy disks along with the disk number.

The entire winchester disk should be backed up, regardless of the actual amount of disk space used. Therefore, continue the backup process until the code 00 is displayed by the seven segment displays.

A complete winchester backup requires the following numbers of floppy disks:

Single-Density, Single-Sided	- 32
Single-Density, Double-Sided	- 16
Double-Density, Single-Sided	- 16
Double-Density, Double-Sided	- 8

If an unrecoverable floppy disk errors occurs during the backup, try another disk. The backup routine will restart at the beginning of the floppy disk on which the failure occurred.

The error recovery abilities of the backup routine are limited. Therefore, it is highly recommended that the backup process be done regularly, prior to any winchester disk failures. It is not possible to backup a winchester disk with hard read errors. However, if the winchester disk has soft header or data CRC errors, the backup routine will retry 16 times before declaring the sector's data invalid.

If the backup routine retries 16 times and is unsuccessful in reading a winchester sector with CRC errors, it will flag the floppy data with a deleted data mark and continue to the next sector. In this manner, it is possible to successfully backup a winchester disk with hard CRC errors; however, the data for that sector stored on the backup floppy disk may be invalid.

The backup routine takes bad tracks into account. Therefore, it is possible to transfer winchester disk images between winchester disk drives with different bad track maps.

Reload

The reload routine does not keep track of how many backup disks have been reloaded onto the winchester. For this reason, it is necessary that the operator conscientiously reload the entire complement of backup floppy disks. Record keeping will be aided by the display of the backup disk number on the seven segment indicators.

Since each backup disk is uniquely identified as to backup version number, it is not possible to intermix the disks of backups which were done at different times.

The reload routine is limited in its error recovery abilities. If a hard read or write error is encountered, the routine will terminate.

CRC error on the floppy or winchester disks will be retried 16 times before the reload routine aborts.

If a deleted data mark is detected on the backup floppy disk in the course of reloading, a 45 error will be displayed by the seven segment indicators. The user should be aware that one or more winchester sectors were unrecoverable at the time of the backup.

4.6.3 Backing Up the Winchester Disk onto Floppy Disks

There are six possible backup classes which may be selected on the DSD 880.

Mode	Class	Description
7	0	Backup without format or verify
7	1	Backup without format, with verify
7	2	Backup with double-density format, without verify
7	3	Backup with double-density format and verify
7	4	Backup with single-density format, without verify
7	5	Backup with single-density format and verify

Select the appropriate backup class and set the MODE and CLASS switches accordingly, insert a floppy disk into the floppy drive, close the door and momentarily depress the EXECUTE pushbutton.

The seven segment displays will echo the switch setting for as long as the EXECUTE pushbutton is depressed.

When the EXECUTE pushbutton is released, the controller will display the current floppy disk volume number (starting from one), lock the door of the floppy drive, and write a unique disk identifier on track 00 of the floppy disk. The disk identifier contains the disk volume number, backup version number starting winchester disk address of the data, and number of sectors of winchester data contained on the floppy.

The controller will then copy the appropriate winchester data onto the floppy from the winchester.

When the operation is complete, the controller will unlock the door of the floppy drive. When the door of the floppy drive is opened, the controller will increment the disk volume number being displayed.

Repeat the preceding steps until the seven segment display again displays 00 indicating that the winchester drive has been successfully backed up.

Select the desired operating mode of the DSD 880, set the switches accordingly, and momentarily depress the EXECUTE pushbutton. The seven segment display will indicate the selected MODE and CLASS until the pushbutton is released and execution begins.

Error Reporting During Backup

1. If a hard error occurs on the floppy drive while the controller is writing to the floppy disk, the operation will terminate. To continue backup, remove the bad disk from the floppy drive and replace it with a new one, then close the door and momentarily depress the EXECUTE pushbutton again. The controller will attempt to recopy the data onto the new disk and continue where it left off.

2. If header and errors occur while copying the floppy, the operator may either insert a new disk into the drive and continue as above, or may select one of the backup classes which will format the floppy before attempting to copy from the winchester and use the same disk again.

3. If unrecoverable CRC errors occur on the winchester drive during the backup procedure, the controller will write deleted data marks on the floppy for the

length of the unrecoverable error code on the seven segment displays. The controller will continue writing deleted data on the floppy until recoverable winchester data is found or the floppy is full.

4.6.4 Reloading the Winchester Disk from Floppy Disks

There are two possible classes which may be selected on the DSD 880.

Mode	Class	Description
6	0	Reload without verify
6	1	Reload with verify

1. Insert the first disk to be reloaded into the floppy disk drive and close the drive door.
2. Start the reload program by selecting the desired MODE and CLASS and momentarily depressing the EXECUTE pushbutton.
3. The seven segment displays will echo the switch setting for as long as the EXECUTE pushbutton is depressed.
4. When the EXECUTE pushbutton is released, the controller will lock the door of the floppy drive and read the disk identifier. If the identifier is valid, the controller will display the disk volume number in the seven segment displays and proceed to copy the contents of the floppy disk onto the winchester disk.
5. When the controller has successfully copied the contents of the floppy onto the winchester, it will unlock the door of the floppy drive and display 00 on the seven segment displays.
6. Repeat steps one and two until all the floppy disks have been reloaded.

Select the desired operating mode of the DSD 880, set the switches accordingly, and momentarily depress the EXECUTE pushbutton. The seven segment displays will indicate the selected MODE and CLASS until the pushbutton is released and execution begins.

Error Reporting During Reload

1. If a hard error occurs during reading the floppy, the same disk may be retried by depressing the EXECUTE pushbutton again. If the error occurs again, the disk may be skipped entirely by removing it and inserting the next disk to be reloaded before depressing the EXECUTE pushbutton.
2. If a disk with an invalid disk identifier is detected, the controller will report an error. The invalid disk must be removed and a valid disk inserted before depressing the EXECUTE pushbutton.
3. If a hard error occurs while the controller is writing to the winchester, the controller will report an error and terminate the reload procedure.
4. An error is indicated by flashing the appropriate error code in the seven segment displays and illuminating the fault and appropriate drive error indicators.

5. If a deleted data mark is detected on the floppy disk during the reload operation, the reload routine will report a deleted data error and continue to copy the questionable data onto the winchester disk.

4.6.5 HyperDiagnostics Mode

The DSD 880x/8 HyperDiagnostics may be used to verify system integrity, troubleshootings, and fault isolation. An expanded description of the HyperDiagnostics and their use is provided in Section 7 of this manual.

5.0 BASIC PROGRAMMING INFORMATION

5.1 General Information

This chapter provides basic programming and register usage information for the DSD 880 System.

5.2 Operating Modes

The DSD 880 has three operating modes: normal, extended, and direct access. The floppy disk drive of the 880 emulates a DEC RX02 with double-sided capability in standard or extended mode. The 880 winchester disk drive emulates a DEC RL01 in standard mode, and provides RL01 operation with increased capacity in extended mode. The RX02 and RL01 emulations in standard mode are fully hardware and software compatible with DEC operating systems.

The direct access mode is intended for use as a diagnostic aid only. The direct access mode provides additional features not available on the DEC RX02 or RL01. The HyperDiagnostics are microcode routines for stand alone self-testing and detailed disk system status reporting.

5.2.1 Single-Sided Operation

The floppy disk drive in the DSD 880 operates as a single-sided disk drive, with single-sided diskettes, and provides a true emulation of the DEC RX02.

5.2.2 Double-Sided Operation

The DSD 880 floppy disk drive is configured for double-sided operation either through standard (single-sided) RSX-11 system options or by using the DSD monitor patch program.

5.2.3 Programming Interface

The system interface for the DSD 880 varies according to both the host computer type and the operational mode for which the system is configured. The DSD 880 operating characteristics are embedded in the DSD 880 controller.

5.3 DSD 880 Floppy Disk Operation and Programming

Data are transferred to and from the diskette in fixed-length blocks called sectors. A sector contains 64, 16-bit words when the system is in single-density mode, and 128, 16-bit words in double-density mode.

The programmer can direct the DSD 880 controller to perform several tasks. Each of these tasks facilitates the storage and retrieval of information on a diskette.

For example, two operations are needed to move a sector of data from main memory to a particular sector on a diskette. The first operation, a fill buffer, moves the data from computer main memory to a RAM buffer internal to the disk controller. The second operation, write sector, positions the read/write head of the flexible disk drive over the specified portion of the diskette and writes the data from the controller sector buffer onto the diskette.

The handler communicates the task requirements to the DSD 880 controller through two physical peripheral device registers which are addressable as though they are in computer memory. The control and status register is normally located at address 777170 octal. The data buffer register is normally located at address 777172 octal.

There are a total of seven logical registers described in this section. These registers represent such information as data, controller status, track addresses, and sector addresses. The handler always reads and writes logical registers through the data buffer register, which is a physical register.

Writing a specific bit pattern to the control and status register initiates a task. Each task is associated with a specific protocol, a set of rules which determines the parameters, or data the computer should pass through the data buffer register during the execution of a task.

For example, operations which move the read/write head in the disk drive require a track address and a sector address. The protocol for these functions is as follows:

1. The command is written to the control and status register.

2. The sector address is written to the data buffer register when the controller requests it.

3. The track address is written to the data buffer register when the controller requests it.

Programmed input/output is used to transfer parameters, but direct memory access (DMA) is used to transfer data between the controller and main memory.

5.3.1 Addressable Registers in RX02-Compatible Operation

Programs communicate with the DSD 880 through two physical registers, the comand and status register (RX2CS), and the data buffer register (RX2DB).

The peripheral device registers reside in the top 4K-words memory address space in DEC-11 computers. The registers are addressed as memory, and any instruction that operates on a memory location can operate on a peripheral device register in the same way; except that certain bits may indicate read only or write only.

Note that the data buffer register, a physical register, acts as a multiple-use logical register as explained under data buffer register (RX2DB).

5.3.2 Command and Status Register

This register is normally at location 777170 (octal) in the memory address space. The bits of this physical register control the DSD 880 floppy disk. The format for this register is shown in Figure 5-1. The RX2CS register also provides the user program with status information and error indications.

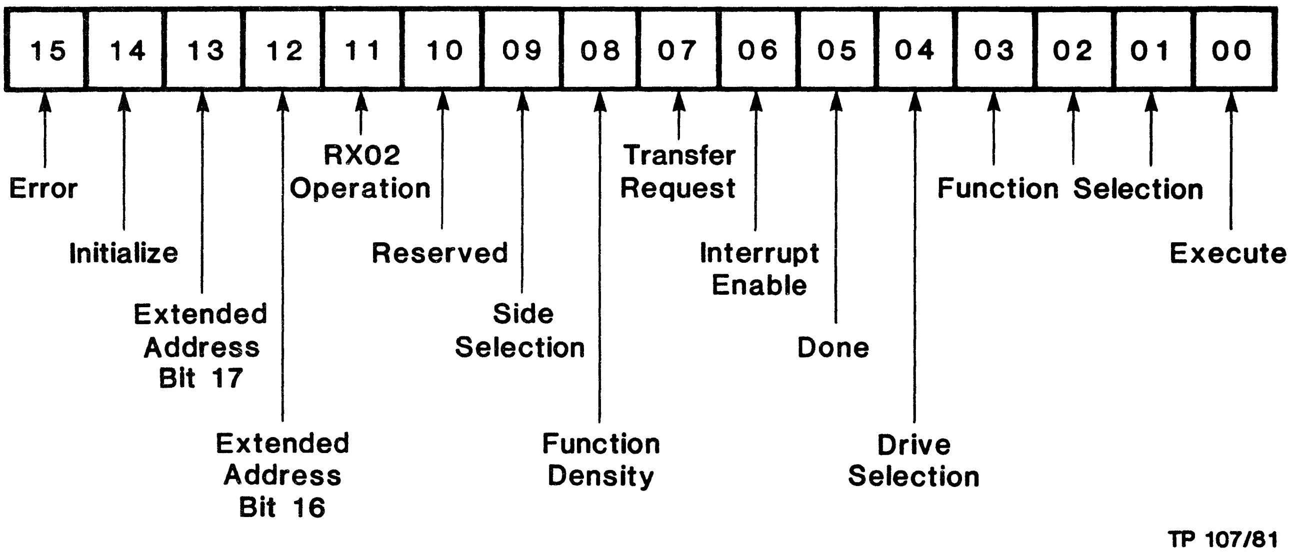


Figure 5-1. Command and Status Register

BIT 15 - ER - Error

This read-only bit is set by the RX02 to indicate that an error has occurred during an attempt to execute a command. It is cleared by the initialize bit (bit 14) hardware bus initialize or by issuing a new command.

BIT 14 - IN - Initilize the DSD 880 floppy disk system

The done flag is reset. The controller resets some internal variables and executes the self-test microcode. The disk floppy drive goes to the home position (track 0).

If the controller is operating in the normal mode and the drive is ready, it reads track 1 sector 1 of the diskette in drive 0. Attempting the read sector operation sets the initialize done bit in the command and status register. Bit 14 is a write-only bit.

BIT 13 - A17 - Extended address bit 17

This write-only bit is asserted on Unibus or Q-Bus address line 17 (A17) when the DSD 880 transfers data by DMA. An initialize bit clears this bit. A17 toggles if A01 through A16 are all ones and the bus address register increments.

BIT 12 - A16 - Extended address bit 16

This write-only bit is asserted on Unibus or Q-Bus address line 16 (A16) when the DSD 880 transfers data by DMA. An initialize bit clears this bit. A16 toggles if A01 through A15 are all ones and the bus address register increments.

BIT 11 - RX02 system identification bit

The software normally uses this read-only bit to differentiate RX01 systems from RX02 systems. The DSD 880 always sets this bit.

BIT 10 - XX - Reserved for possible future use

BIT 9 - HS - Head select bit

This read/write bit selects side 0 or 1 (lower head or upper head). It is set to select side 1, and cleared to select side 0.

BIT 8 - DEN - Density of function

This read/write bit specifies the density for the function encoded in bits 1, 2, and 3. This bit specifies high density when it is set.

NOTE

Even though the fill buffer and empty buffer functions do not use magnetic media, a valid density bit is required for the controller to evaluate the validity of the word count parameter.

BIT 7 - TR - Transfer request flag

This read-only bit indicates to the program that the data buffer register is empty and needs loading, or is loaded and needs emptying.

BIT 6 - IE - Interrupt enable bit

This read/write bit, when set, allows an interrupt to be generated whenever the done flag is set.

BIT 5 - DN - Done flag

This read-only bit indicates the completion of an operation. The bit works in conjunction with the interrupt enable (IE) bit to generate interrupts.

BIT 4 - UNI - Unit select bit

This read/write bit selects floppy drive 0 or 1. In the DSD 880, the floppy drive selected is always drive 0. Drive selection occurs only if a drive-related function is executed.

BITS 3 through 1 - F2, F1, F0 - Function select

The binary encoding of these write-only bits selects the function to be performed by the DSD 880 system as indicated below:

F2	F1	F0	Command Specified	Octal Function Code
0	0	0	Fill Buffer	0
0	0	1	Empty Buffer	1
0	1	0	Write Sector	2
0	1	1	Read Sector	3
1	0	0	Set Media Density	4
1	0	1	Read Status	5
1	1	0	Write Deleted Data Sector	6
1	1	1	Read Error Code	7

BIT 0 - EX - Function execute

This bit controls the execution of the function encoded in bits 1 through 3 of this register. This is a write-only bit.

5.3.3 Data Buffer Register (RX2DB)

The RX2DB data buffer register provides the communication link between the host processor and the DSD 880 system. The register transfers data to and from the controller data buffer. The logical register information passing through the register depends on a predetermined protocol.

If the DSD 880 is not executing a command, the RX2DB can be modified without risk of adverse effects. However, during the execution of an instruction, the RX2DB register provides or accepts information (according to the RX2DB protocol) whenever the transfer request flag is set.

CAUTION

Data may be lost if an incorrect protocol is followed.

The following descriptions explain the various logical register formats of the physical data register (RX2DB).

Disk Track Address Register (RX2TA at 777172) - During commands such as write sector and read sector, which require a track number (or a cylinder number) during double-sided operation, the number is written into the physical RX2DB register. Track or cylinder numbers from 0 to 76 (decimal) are valid.

Disk Sector Address Register (RX2SA at 777172) - During commands such as write sector and read sector, which require a sector address, the address is written into the physical RX2DB register. Sectors addresses from 1 to 26 (decimal) are valid. Bits 6 and 7 of RX2SA are masked to zero.

Word Count Register (RX2WC at 777172) - The word count register specifies the number of words for DMA transfer between the controller sector buffer and main memory. For a double-density sector, the maximum word count is 128 (decimal), or 256 bytes. For single-density sector, the maximum word count is 64 (decimal), or 128 bytes. In each case, the programmer loads the actual word count, not the two's complement of the word count, into the word count register.

Bus Address Register (RX2BA at 777172) - This register specifies the bus address for the data transfer during a DMA operation. It increments by two following each data transfer.

The bus address register is write-only. It should always be loaded with the starting memory address of a data buffer at the appropriate time during the fill buffer, empty buffer, or read extended status operations.

System Error and Status Register (RX2ES) - The RX2ES register is another logical register implemented using the physical RX2DB register. It provides error and status information about the drive specified by bit 4 of the (physical) RX2CS register. At the completion of a command, the controller places the contents of the RX2ES register into a data buffer register (RX2DB = 777172) so the host processor can check the status and error results of the most recent operation. When the controller completes an operation that does not select a drive (fill buffer, empty buffer), the RX2ES unit select and drive density bits remain unmodified. All the other RX2ES bits are cleared at the initiation of each new function. See Figure 5-2 for the format of this register.

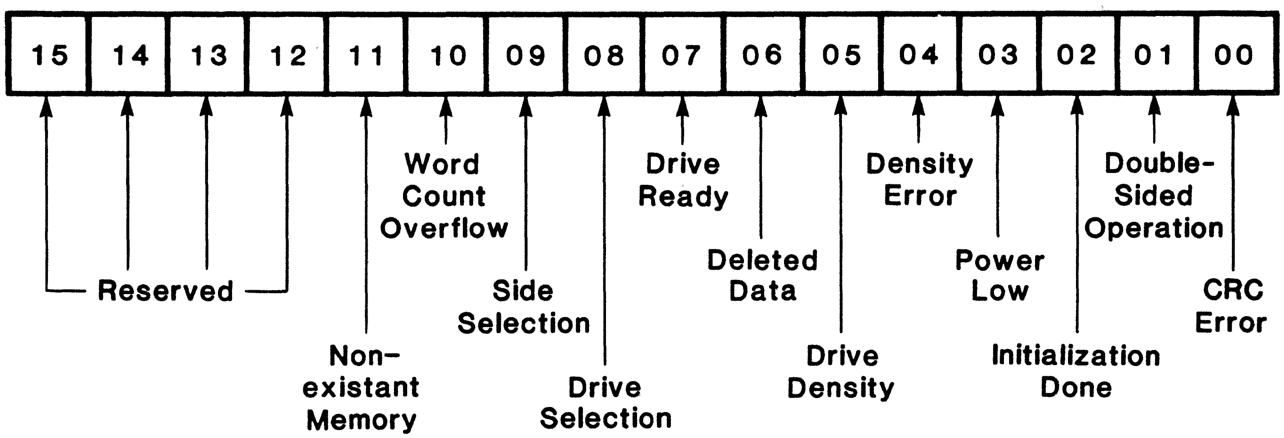


Figure 5-2. System Error and Status Register Bit Format

BITS 15 through 12 - Not used

BIT 11 - NXM - Nonexistent memory error

This bit sets if, during a DMA cycle, the interface does not receive a bus reply when it tries to write or read a word to or from memory. Usually no bus reply means that the address in the RX2BA or the extended address bits in the RX2CS are invalid. The operation terminates and error and done bits are set. To recover from this error condition, generate either a bus or a programmed initialize.

BIT 10 - WC OVFL - Word count overflow

This bit sets if the word count specified during a fill or empty buffer command is too large for the sector size indicated by the density bit. At a word count overflow, the operation terminates, and the error and done bits are set.

BIT 9 - HD SEL - Head selected

This bit indicates the read/write head selected during the most recent read or write operation. It sets to indicate the upper head, and clears to indicate the lower head.

BIT 8 - UNIT SEL - Unit select

This bit indicates the disk drive head selected during the most recent read or write operation. It sets to indicate drive 1 and clears to indicate drive 0.

BIT 7 - DRV READY - Drive ready

This bit, when set, indicates that the selected disk drive has a diskette correctly installed and up to speed. The drive ready bit is valid immediately following the read status function. This bit is also valid for drive 0 immediately following an initialization. (See BIT 1 SDI RDY)

BIT 6 - DD - Deleted data

This bit indicates that a deleted data address mark was found during the most recent read sector operation, or that the most recently executed command was write deleted data sector.

BIT 5 - DRV DEN - Drive density

This bit indicates the density of the diskette in the drive indicated by bit 8. Bit 5 is updated during a read or write sector operation.

BIT 4 - DEN ERR - Density error

This bit indicates that during a read sector, write sector, write deleted data sector, or read status operation the diskette density and the density indicated by the density bit of the RX2CS do not match. Any operation terminates, and the error and done bits are set.

BIT 3 - PWR LO - Power low

This bit indicates a power failure in the controller/drive subsystem. It also sets if the interface cable disconnects. Any operation terminates, and error and done bits are set.

BIT 2 - ID - Initialize done

This bit indicates that the controller/drive has completed an initialization sequence. This sequence may be initiated by a power failure, a bus or a programmed initialize.

BIT 1 - SDI RDY - Side 1 ready

Bit 1 and bit 7 are both set when a double-sided diskette is correctly installed and up to speed. When bit 7 is set but bit 1 is clear, a single-sided diskette is installed and up to speed. A single-sided diskette is restricted to side 0 functions only.

BIT 0 - CRC - Cyclic redundancy check error

This bit indicates that a cyclic redundancy check error was detected during the most recent read sector operation. The operation terminates, and the error and done bits are set.

5.3.4 Floppy Disk Controller Command Protocols

The following sections describe the protocol for each command that can be sent to the controller. Failure to adhere to the correct protocol results in lost or incorrect data.

Function Code 0 - Fill Sector Buffer Command

The fill sector buffer command fills a storage buffer in the DSD 880 with 128 or 256, eight-bit bytes of data from computer memory. To write the data to the diskette or transfer it back to memory, use other functions.

When the fill sector buffer command is given, the DSD 880 responds by clearing the done flag (RX2CS bit 5). The controller then requests a word count by setting the transfer request flag. The program should respond by writing a valid RX2WC (word count) into the RX2DB. When the controller again asserts transfer request, the program should respond by writing a valid starting memory address (RX2BA) into the RX2DB.

Loading RX2BA clears transfer request, and it remains clear for the duration of the fill sector buffer. The data bytes transfer directly from memory to the controller sector buffer. The done flag sets when the word count is decremented to zero and the controller has zero-filled the remainder of the sector buffer (if necessary). Also, if interrupts are enabled (RX2CS bit 6 is set) when the done flag sets, an interrupt request occurs. The contents of the RX2ES register are left in the RX2DB at the completion of the operation.

NOTE

> Bit 4 of the RX2CS does not affect this function because no disk drives are selected. The density bit, RX2CS bit 8, must be set correctly because the controller uses this bit in evaluating the validity of the word count.

Function Code 1 - Empty Sector Buffer Command

The empty sector buffer command transfers the contents of the floppy sector buffer to main memory. The sector buffer is loaded from a previous fill sector buffer or read sector command.

The controller responds to an empty sector buffer command by clearing the done flag (RX2CS bit 5). The controller then sets the transfer request flag (RX2CS bit 7) to request the contents of the word count register. The program should respond by loading a valid word count into the data buffer register.

When transfer request is asserted again, the program responds by loading the starting memory address into the data buffer register. The controller than clears the transfer request flag which remains clear for the rest of the operation.

The data in the sector buffer is transferred to memory one word at a time, decrementing the contents of the word count register at each transfer, until the word count becomes zero. When the data transfer is completed, the controller places the contents of RX2ES into the data buffer register and sets the done flag. If the interrupt enable bit is set, setting the done flag initiates an interrupt request.

The information above, which applies to the fill buffer command, applies equally to the empty sector buffer command. Note that the empty buffer operation does not modify the contents of the sector buffer.

Function Code 2 - Write Sector Command - (Bit 9 selects side 0/side 1)

The write sector command transfers the contents of the sector buffer to a specified track and sector of the diskette.

When the write sector command is given, the controller clears the logical RX2ES register and the done flag.

Next, the transfer request flag (RX2CS register bit 7) is set to request the sector address (RX2SA) from the CPU. When the sector address is received, the transfer request flag is removed. The transfer request flag is then set to request the desired track address (RX2TA) from the CPU. When the track address is written to the RX2TA, the transfer request flag is cleared.

After the track address is received, the controller makes the selected drive seek the desired track. Transfer request is left reset for the remainder of the operation. The heads are loaded against the media and positioned over the specified track. If the controller does not know the density and format of the media, it reads a random sector on the target track to determine the density.

If the media density does not agree with the command density (RX2CS bit 8), the operation terminates and bit 4 of the RX2ES register indicates a density error. If the densities agree, the controller checks the track address and looks for the specified sector address. If the correct track and sector are found, the controller writes either 128 bytes of single-density data or 256 bytes of double-density data from the sector buffer to the diskette. Two CRC bytes are written immediately after the data.

If the controller finds an invalid track address, the extended status error code is set to 40. If the contents of RX2TA does not match the track address from the header, the extended status error code is set to 150. If the specified sector cannot be found within the two diskette revolutions, the extended status error code is set to 70. Either of these error conditions, or a density error terminates the operation. The error flag (RX2CS bit 15) and the done flag (RX2CS bit 5) are asserted when the function terminates due to an error condition. As with the error free termination, an interrupt request is generated if the interrupt enable bit is set when the done flag becomes true. The extended error status can only be read by the read extended status command (17_8).

NOTE

> The contents of the sector buffer are not modified by the write sector function. If the contents of the sector buffer are modified as a result of a power failure or the initialize command, users must be sure that valid data are written back into the sector buffer. This is especially true before executing the write sector command. If a sector number of 154 or 155 is written to the RX2SA, the write sector function turns into a write format track function.

Function Code 3 - Read Sector Command (Bit 9 selects side 0 or side 1)

The read sector command locates a specified track and sector of a diskette and transfers the contents of the data filed into the sector buffer in the controller.

The controller clears the logical RX2ES register and the done flag when the read sector command is given. Next, the transfer request flag sets (RX2CS bit 7) to request a sector address. The program responds by writing the desired sector address (RX2SA) into the data buffer register, RX2DB (at 177172 typically), which clears the transfer request. After receiving the sector address, the transfer request flag is again set to request the track address. The program responds by writing the desired track address into the RX2TA (at 177172, typically). When the RX2TA is received, the transfer request flag is again cleared.

After receiving the track address, the controller causes the selected drive to seek the desired track. Transfer request is left reset for the remainder of the operation.

The controller loads the heads against the media and determines the density of the media if the density is unknown. If the diskette density does not agree with the command density (RX2CS bit 8), an error is reported and the operation terminates. If the densities agree, the controller looks for the specified sector. When the correct sector is located, the controller looks for the appropriate data or deleted data address mark.

If a data address mark is found, the controller transfers the next 128 bytes (single-density) or 256 bytes (double-density) into the sector buffer followed by the two CRC bytes. An error free read is indicated if the address mark, data bytes, and two CRC bytes produce a zero residue when passed sequentially throughout the CRC checker hardware circuits. As soon as the data are available in the buffer, the controller terminates the operation by writing the contents of RX2ES to the data buffer register and setting the done flag. An interrupt request is generated if the interrupt enable bit is set when done becomes true.

If a deleted data address mark is detected, the controller sets the deleted data flag. This flag appears in the error/status register (as RX2ES bit 6). If a CRC error is detected, the controller sets RX2ES bit 0 and the error flag (RX2CS bit 15). Seek errors and missing-sector errors are reported as in the write sector command.

Function Code 4 - Set Media Density Command

This command initializes an entire DEC-formatted diskette to a specified density. When the set media density command is executed, the controller attempts to write zeroes in every field on the diskette. Bit 8 of the RX2CS determines the recording density and the type of data address mark to be written in each data field. No sector headers are written when the set media density command is executed.

When the set media density command is received, the controller clears the done flag. Next, the controller sets the transfer request flag. The program responds by writing a key byte into the physical register RX2DB. If the key byte is an ASCII I (111 in octal), the set media density function is executed. If the key byte written into the RX2DB is not an I, the done and error flags are set and the operation terminates. The extended error status register is then loaded with 250 to indicate an invalid key byte. The purpose of the key byte is to make accidental erasure of the data on a diskette difficult.

As soon as the safety character I is received, the controller moves the heads to track 0. When sector 1 is found, the controller starts writing. If bit 8 of the RX2CS is a 0, a single-density data address mark and 128 FM-format zeroes are written. If bit 8 of the RX2CS is a 1, a double-density data address mark and 256 DEC-MFM-format zeroes are written. After writing all 26 sectors on track 0, the controller seeks track 1, track 2, etc., writing all 26 sectors on each track. If the disk is two-sided, the second is done automatically. The write continues until either every sector has been written through track 76: sector 26, or a bad header is found. The error and done flags are set if the operation terminates due to a bad header.

The set media density command requires approximately 27 seconds for a single-sided disk, and 54 seconds for a double-sided disk, depending on the sector interleave. Never interrupt the set media density command before it is completed. If the function does not terminate normally, an illegal diskette with data address marks of both densities may be created. In this case, completely rewrite the diskette. If the set media density command is incomplete due to an unreadable header, use the track format procedure to rewrite the incorrect header information.

Function Code 5 - Read Status Command

The read status command determines the current status of the drive selected by RX2CS bit 4. The information returned consists of the drive readiness status and the density of the diskette currently in the drive.

Issuing the read status command clears the done flag. The controller checks that the door of the selected drive is closed, a diskette is inserted, and the diskette is up to speed. Diskette speed is determined by measuring the amount of time between successive index pulses. Because this measurement takes an average of 250 milliseconds, excessive use of the read status function causes reduced throughput. If the drive is ready, the controller sets bit 7 (drive ready) of the RX2ES, then loads the heads and reads the first sector it finds. If the disk is double-sided, bit 1 of the RX2ES is set to 1.

If a double-density address mark is detected, bit 5 (drive density) of the RX2ES is set. If a single-density mark is found, bit 5 is cleared. The controller terminates the function by shifting the contents of the RX2ES to the RX2DB and setting the done flag. An interrupt request is generated if the interrupt enable bit, RX2CS bit 6, is set when done becomes true.

Function Code 6 - Write Deleted Data Sector Command

This command performs the same task as the write sector command, except that it writes a deleted data address mark just before the data field. The standard write sector command writes a regular data address mark. Reading a sector written with a deleted data address mark sets bit 6 of the logical RX2ES register.

The density bit associated with this function (RX2CS bit 8) determines whether a single- or double-density deleted data address mark is written.

Function Code 7 - Read Extended Status Command

The read extended status command retrieves information from several internal controller registers, including the error register, as shown below. These registers are transferred to memory using DMA. As soon as the command is loaded into the RX2CS, the done flag clears. The controller then asserts the transfer request flag.

The program then loads a starting memory address into the RX2DB. The controller transfers four words directly to memory. When the words are in memory, the controller asserts done, generating an interrupt request if interrupts are enabled.

The words transferred to memory are as follows:

Word 1	:	BITS	0 - 7	Error Code (See Table 5-1)
	:	BITS	8 - 15	Word Count Register
Word 2	:	BITS	0 - 7	Current Track Address of Drive 0
	:	BITS	8 - 15	All 0s
Word 3	:	BITS	0 - 7	Target Track of Current Disk Access
	:	BITS	8 - 15	Target Sector of Current Disk Access
Word 4	:	BIT	0	Density of Read Error Register Command
	:	BITS	1, 2, 3	Unused
	:	BIT	4	Drive Density of Drive 0
	:	BIT	5	Head Load Bit
	:	BIT	6	0
	:	BIT	7	0
	:	BITS	8 - 15	Track Address of Selected Drive

Table 5-1. Error Register Codes for RX2ES (Function Code 7)

Octal Code	Description
000	No errors
010	Drive failed to home on initialize
020	Nonexistent drive

Table 5-1. Error Register Codes for RX2ES (Cont)
(Function Code 7)

Octal Code	Description
030	Track 00 found while stepping in on initialize
040	Invalid RX02 track address
050	Track 00 found before desired track while stepping in
070	Requested sector not found in two revolutions
100	Write protect violation
120	No preamble found
130	Preamble found, but no address mark within window
140	CRC error on what appeared to be a header
150	Address in header did not match desired track
160	Too many tries for an ID address mark
170	Data address mark not found in allotted time
200	CRC error on data field
240	Media density did not match desired density (RX02 only)
250	Wrong key in set media density command
260	Indeterminate media density (RX02 only)
270	Write format failure
350	Nonexistent memory error during DMA
360	Drive not ready (door open, speed error, or absent media)
370	Low ac power caused abort of write activity

5.3.5 Disk Formatting

This procedure allows repair of magnetically damaged disks. When configured for RX02 operation, the DSD 880 can format disks in the two formats shown in Table 5-2. The entire disk is formatted.

NOTE

> The DEC RX02 does not support the command protocol described below. It is a special feature which is unique to the DSD 880.

1. The program issues the write sector function code (010) to the controller using the command and status register. The density bit (bit 9) is ignored. The side bit is also ignored.

2. The controller then clears the done flag and sets the transfer request flag (bit 7 RX2CS).

3. The user must then write an octal value corresponding to the desired format into the data buffer (RX2DB). The controller sets transfer request flag again. The user then writes 0 into RX2DB. Table 5-2 lists the available formats. When the operation is completed, the controller sets the done flag. An interrupt occurs if bit 6 (interrupt enable) is set prior to the format command.

Table 5-2. Diskette Format Codes

ID Code	Description	Density	Sectors/ Track	Track #
154_8	Format the entire disk with FM-coded single-density. Both sides of a double-sided diskette are formatted	Single	26	0 to 76
155_8	Format the entire disk with DEC-modified MFM, double-density. Both sides of a double-sided disk are formatted.	Double	26	0 to 76

5.3.6 Power Fail

When a power failure occurs, or dc power to the DSD 880 is interrupted, the controller gradually drains the filter capacitors and stops executing microcode. The done and error bits set in the RX2CS, and the PWR LOW bit sets in the RX2DB signal the program that the controller/drive subsystem has lost power. When power is restored, the DSD 880 controller initiates the following sequence. At the end of this sequence, the controller sets RX2CS bit 5 (done flag).

1. Clears done.
2. Executes the hardware self-tests.
3. Positions drive to track 00.
4. Clears RX2ES of all active error bits.
5. Reads sector 1, track 1 of the floppy disk into the floppy buffer, if the drive is ready, and leaves floppy head at track 1.
6. Sets bit 2 of RX2ES (initialize done).
7. Updates bits 7 (drive ready) and 5 (drive density) of RX2ES according to the status of drive 0.

5.3.7 Common Programming Mistakes

Use the following descriptions of common programming mistakes and hints to avoid data loss and/or error conditions.

1. Sending an illegal track or sector address to the controller. The valid sectors are 1 through 26 (decimal), and the valid tracks are 0 through 76 (decimal).
2. Providing an incorrect word count for the length of a variable length sector/density set in the fill or empty command.
3. Underestimating the duration of the read status command. The read status command requires up to two revolutions of the disk to complete. To avoid excessive delays, use this command only when necessary.
4. Not checking the initialize done bit following a read or write operation. A short power outage sets the done flag without error indication. Check the initialize done (RX2ES bit 2) for an indication of power failure.

5. Decoding the drive select bit during fill buffer and empty buffer operations. The drive select bit, RX2CS bit 4, may not be decoded by the controller during fill and empty buffer functions.

6. Using a one-sector interleave. Use a two-sector interleave (sectors 1, 3, 5, etc.) for optimal data transfer rate.

7. Using the incorrect type of diskette. For both single- and double-density recording, use only a 26 sector per track diskette. Do not use a hard sectored disk (multiple sector/index holes).

8. Typically, a fill buffer command precedes a write sector command. Similarly, a read sector command precedes an empty buffer command.

5.3.8 Interrupts

The interface module requests an interrupt whenever the interrupt enable and done flag bits of the RX2CS both become set. The standard interrupt vector address is location 264 octal.

5.4 DSD 880 Winchester Disk Operation and Programming

The DSD 880 winchester disk drive has two operating modes. In the normal mode, the drive emulates a single DEC RL01 with a formatted capacity of 5.2 megabytes. In the extended mode, the drive operates as a diminished RL02 with the formatted capacity increased to 7.8 megabytes. The two operating modes are selected by means of the DSD HyperDiagnostics panel mode switch as described in Section 4 of this manual.

5.4.1 Bad Track Mapping

The winchester drive, used in the DSD 880, provides 256 cylinders with four tracks per cylinder and 32 sectors per track. Each sector contains 256 bytes. The total capacity of the winchester drive is 32,768 sectors, or 8,388,608 bytes.

The current state of the art in the production of winchester recording media is such that it is not possible to guarantee a flawless recording surface; it is expected that there will be a certain number of defects on the disk. The locations of these defects are recorded at the factory in a bad-track map, located on physical cylinder 0 of the winchester drive. The DSD 880 controller automatically reads this bad track map when power is first applied, and subsequent accesses of the winchester disk are adjusted automatically by the controller to avoid the flawed areas. Fifteen tracks per head, or 60 tracks in all, are reserved as spares.

It is possible to add entries to the bad track map, by use of a special diagnostic program (WINEXR) supplied by DSD. Its use is described in an appendix to this manual. The winchester disk should be backed up onto floppy disks prior to use of the WINEXR program. A hard-copy record is made at the factory of the data entered into the bad track map. This record is stored in an envelope on the front of the winchester drive, just behind the HyperDiagnostic panel. Changes to the bad track map should be noted on the record.

The bad track map and spare tracks are not available for user data storage. The maximum usable capacity of the winchester disk is 240 cylinders, with four tracks per cylinder and 32 sectors per track, or 30,720 sectors (7,864,320 bytes).

5.4.1.1 Locating Additional Bad Tracks

It is assumed that the DSD 880 has been unpacked, set up to run HyperDiagnostic 23 and encountered an error, or that the unit has been running in the field and an error has been encountered. The equipment needed for this procedure are an LSI-11 System and a DSD 060050, Revision C diagnostic disk.

Error Determination. If any error occurs while running HyperDiagnostic 23, the definitive error code (See Table 7-4) is displayed on the seven segment display and the appropriate drive fault indicator is lit. Do not restart the test upon encountering the first error; some mappable errors don't occur until after four or five passes.

Read Errors. Bad tracks generally cause various read errors. When a read error is found on the Winchester, perform the following scan to locate the bad track. The bad track can be mapped out by the bad track mapping function of WINEXR.

1. Connect the 880 to an LSI-11 System using an 8836 interface card. Set the mode and class switches to 02.
2. Boot the systems on the diagnostic floppy disk supplied with the 880.
3. Run WINEXR (SATEST on older systems); answer the configuration questions.
4. To the prompt COMMAND: enter SCAN. The Winchester then scans all tracks for CRC errors in the header and data fields (the data is ignored).

If any error occurs during the scan, the track and head information (in octal) will be displayed in the error message.

If no errors were found during the scan run the WINEXR acceptance test overnight. If no errors occur at this time, it means that any previous errors encountered in HyperDiagnostic 23 were soft and should be restarted on 23. This should be allowed twice during the 168-hour run time.

If errors occur all on one track, map out the bad track and restart HyperDiagnostic 23. If the errors are on random tracks, reformat the drive and restart HyperDiagnostic 23. If this happens a second time, replace the drive.

Mapping Out Tracks. A bad track founo during the scan can be mapped out using the bad track mapping function of WINEXR, as shown in the following.

```
COMMAND: BAD TRACK MAPPING
ENTRY OF NEW BAD TRACK MAP? N
MODIFY BAD TRACK MAP
```

ADD ENTRIES TO BAD TRACK MAP? Y
ENTER LATEST UPDATE: 5 NOV 81
DECIMAL? OCTAL INPUT? 0 (for octal)
TRACK: 55 HEAD: 2 (enter the bad track and head)
TRACK: (TYPE SPACE AFTER THE LAST ENTRY)

ANY MORE INPUT? N
EDIT ADDITIONS? N
FLOPPY DRIVE IN UNIT? Y
SA800 FLOPPY DRIVE? (enter Y for single-sided and N for double-sided)
WRITE BAD TRACK MAP ON DISK? Y
ANY MORE BAD TRACKS? N
COMMAND:

After the track is added to the bad track map, record it on the document in the pocket on the front of the drive, behind the diagnostic panel. Disconnect the 880 from the LSI-11 and restart in HyperDiagnostic 23.

5.4.2 Normal Mode (RL01 Emulation)

The DEC RL01 provides 256 cylinders, with two heads (tracks) per cylinder and 40 sectors per track, for a total of 20,480 sectors. Each sector contains 256 data bytes. The total capacity of the RL01 is 5,242,880 bytes. In normal mode, the DSD 880 controller converts RL01 cylinder, head, and sector addresses into a form compatible with the winchester drive. Corrections for bad tracks are totally transparent.

5.4.3 Extended Mode

In normal (RL01 emulation) mode, 10,240 sectors (2,621,440 bytes) of available winchester storage is inaccessible to the user. Extended mode makes this capacity available by emulating a diminished RL02 (an RL02 provides 10.4 megabytes of storage, greater than the capacity of the winchester drive). In this mode, the DSD 880 controller converts RL02 cylinder, head, and sector addresses into a form compatible with the winchester drive, and corrections are automatically made for bad tracks. The last available sector is RL02 cylinder 577 (octal), head 1, sector 47 (octal). An error will be reported if an attempt is made to access a higher sector, except that the last track of the RL02 (bad block map) is mapped onto the winchester disk.

Extended mode also provides a spiral read/write/write check capability. The DEC RL02 requires that a seek command be issued to position the heads, followed by a read, write, or write check command to do the data transfer. The read, write, or write check command must specify the same cylinder and head set up by the seek command. If the word count exceeds the capacity of a single track, an error will result. In extended mode, the DSD 880 will seek to the specified cylinder and head on receipt of a read, write, or write check command, and will seek again if the word count exceeds the capacity of a single track; it is actually not necessary to use seek commands at all. DEC software does not support this feature, but it may be useful when special handlers are being planned.

5.4.4 DEC Bad Block Map

DEC provides a method of flagging bad blocks (one block is two sectors) in the RL01 and RL02 by providing a list of bad blocks on the last track of the disk pack (cylinder 377 octal, head 1 for the RL01, and cylinder 777 octal, head 1 for the RL02). This technique is fully supported by the DSD 880 since the bad block maps are present on the winchester disk and the correction for bad blocks is handled by DEC software. The DSD SATEST diagnostic, which updates the bad track map, also writes valid (empty) bad block data into the appropriate sectors.

DEC provides utility programs to add entries into the bad block area. These may be used with the DSD 880. The bad block data will be saved on floppy disks during a backup operation, and will be restored during the reload operation. This should be taken into consideration if the backup and reload functions are used to transfer a disk image between different DSD 880s.

5.4.5 Addressable Registers

The DSD 880 winchester disk drive (RL01 emulation) provides the following four types of physical, addressable registers:

- Control Status Register
- Bus Address Register
- Disk Address Register
- Multipurpose Register

These registers are described in the following.

5.4.6 Control Status Register

The 16-bit control status (CS) register has a base address of 774400. As shown in Figure 5-3, bits 1 through 9 and read/write bits (bit 0 and 10 through 15) are read-only.

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
ERR	DE	NXM	DLT	DCRC	DP1	DS1	DS0	CRDY	IE	BA17	BA16	F2	F1	F0	DRDY
			HNF	HCRC	DP1										

Read Only (bits 15–10) — Read/Write (bits 09–01) — Read Only (bit 00)

TP 234/81

Figure 5-3. Control Status (CS) Register Format

A bus initialize (BINIT L) sets bits 7 and 0 (continuously), and clears bits 1 through 6 and 8 through 13.

The start of each controller command clears the error indicating bits (10 through 13). The completion of each controller command sets bit 7. Note that the detection of an error during command execution also sets bit 7. The function of the control status register bits is detailed in the following.

BIT 15 - ER - Composite error bit

When set, this bit indicates that at least one of the error detection bits (bits 10 through 14) is set. Note that if an error occurs when the interrupt enable bit (bit 6) is set, the current operation terminates and interrupt occurs.

BIT 14 - DE - Drive error bit

This bit is set if a winchester drive related error occurs. The execution of a get status command identifies the source of the drive error. Clear this bit by correcting the drive error, or by executing the get status command with bits 3, 0, and 1 of the data address register set.

BIT 13 - NXM - Nonexistent memory bit

During a DMA data transfer, bit 13 set specifies that no memory response was received with 10 to 20 μs.

BIT 12 - DLT/HNF - Data late or header not found

The function of this bit is explained as follows:

OPI (Operation Incomplete) (bit 10)	DLT/HNF (bit 12)	Indication
Set	Set	Header not found; controller search for the correct read or write sector exceeded the 200 milliseconds timeout limit.

BIT 11 - DCRC/HCRC - Data or header cyclic redundancy check

This bit indicates data and header cyclic redundancy check errors as follows:

OPI (Operation Incomplete) (bit 10)	DCRC/HCRC (bit 11)	Indication
Cleared	Set	Data CRC error
Set	Set	Header CRC error

Note that on a write check command, DCRC/HCRC set and OPI clear indicates that the CRC error is a write check error.

BIT 10 - OPI - Operation incomplete

OPI sets when an error occurs which prevents transfer of data.

BITS 8, 9 - DS0, DS1 - Drive select

These bits specify which drive communicates with the controller. Note that the DSD 880 currently supplies a single rigid-disk drive (DS0). Selecting DS1 causes an error. (Both DS0 and DS1 should be 0.)

BIT 7 - CRDY - Controller ready

The software clears this bit to initiate the execution of the command in bits 1 through 3. When this bit is set, the controller is ready to accept another command.

BIT 6 - IE - Interrupt enable

When this bit is set (by software), the controller will interrupt the processor at the normal or error caused termination of a command.

BITS 4, 5 - BA16, BA17 - Bus address extension

These bits function as the two high-order address bits of the bus address register, but are read and written as bits in the control status register.

BITS 1, 2, 3 - F2, F1, F0 - Function

These bits specify the command to be executed according to the following:

F2	F1	F0	Command Specified	Octal Code
0	0	0	NOP (clear errors)	0
0	0	1	Write Check	1
0	1	0	Get Status	2
0	1	1	Seek	3
1	0	0	Read Header	4
1	0	1	Write Data	5
1	1	0	Read Data	6
1	1	1	Read Data Without Header Check	7

BIT 0 - DRDY - Drive ready

When bit 0 is set, the drive is ready to receive a command.

5.4.7 Bus Address Register

The 16-bit bus address (BA) register has a base address of 774402. The BA register (Figure 5-4) specifies the memory location for the data transfer of a normal read or write operation. At the transfer of each word between the disk drive and the processor bus, the BA register contents increment by two. The BA register may be read only when bit 7 (CRDY) of the CS register is set.

Bit 0 in the BA register is always zero. All 16 bits are read/write bits. To clear the register, execute a bus initialize or load the register with zeroes. Note that the BA register expands to an 18-bit register with bits 4 and 5 of the control status register becoming BA16 and BA17.

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
BA15	BA14	BA13	BA12	BA11	BA10	BA9	BA8	BA7	BA6	BA5	BA4	BA3	BA2	BA1	0

Read/Write

TP 110/81

Figure 5-4. Bus Address Register Format

5.4.8 Disk Address Register

The 16-bit disk address (DA) register, at address 774404, is a three function register. The function depends upon the current command as explained below. The DA register may be read only when bit 7 (CRDY) of the CS register is set.

1. Disk Address Register for a Seek Command

During a seek operation, the DA register provides the drive with the head direction, head select, and cylinder address difference as shown in Figure 5-5 and described below.

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
DF8	DF7	DF6	DF5	DF4	DF3	DF2	DF1	DF0	0	0	HS	0	DIR	0	1

Extended Mode Only

TP 111/81

Figure 5-5. Disk Address Register Format During a Seek Command

BITS 7 through 15

These bits provide the cylinder address difference, which is the number of cyclinders the heads must move for the seek.

BITS 5, 6 - Reserved

BIT 4 - HS - Head Select

This bit specifies upper (HS clear) or lower (HS set) head (and disk surface) for the seek operation.

BIT 3 - Must be 0

BIT 2 - DIR - Direction for the seek operation

Bit 2 set specifies head movement toward the spindle. The head movement is away from the spindle if bit 2 is clear.

BIT 1 - Must be 0

BIT 0 - Must be 1

2. Disk Address Register for a Read or Write Command

For a read or write operation, the DA register initially contains the address of the first sector for the read or write. The contents of the register increment by one with each sector transfer. Figure 5-6 shows the DA register format for standard mode operation. The contents are described below.

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
CA8	CA7	CA6	CA5	CA4	CA3	CA2	CA1	CA0	HS	SA5	SA4	SA3	SA2	SA1	SA0

Set For Extended Mode Only

TP 112/81

Figure 5-6. Disk Address Register Format for a Read or Write Command

SA0 through SA5 - Sector address for one of the 40 sectors on the track. (Valid sectors are 0 through 39).

HS - Head select specifies the head (disk surface) for the read or write: upper (clear) or lower (set).

CA0 through CA8 - Cylinder address of one of the 256 cylinders. CA8 is used for extended mode only.

3. Disk Address Register for a Get Status Command

The contents of the DA register for a get status command are shown in Figure 5-7 and explained below.

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
X	X	X	X	X	X	X	X	0	0	0	0	RST	0	GS	1

TP 113/81

Figure 5-7. Disk Address Register Format for a Get Status Command

BITS 8 through 15 - Not used

BITS 4 through 7 - Must be 0

BIT 3 - RST - Reset bit

When the bit is set, the drive first clears the error bits, then sends the status word to the controller.

BIT 2 - Must be 0

BIT 1 - GS - Get status

This bit must be a 1 to request the status word from the drive and to direct the drive to ignore bits 8 through 15. As soon as the get status command is completed, the controller multipurpose register (described below) is loaded with the drive status word.

BIT 0 - Must be 1

5.4.9 Multipurpose Register

The 16-bit multipurpose (MP) register, like the disk address register, is a triple-function register. The function depends on the command used.

1. Multipurpose Register for a Get Status Command

When a status word is returned to the controller following execution of a get status command, the MP register contents are as pictured in Figure 5-8 and explained below. The MP register may be read only when bit 7 (CRDY) of the CS register is set.

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
WDE	HCE	WL	SKTO	SPE	WGE	VC	DSE	0	HS	CO	HO	SH	STC	STB	STA

TP 114/81

Figure 5-8. Multipurpose Register Format for a Get Status Command

BIT 15 Always 0
BIT 14 Head current error - write current was detected in the heads when the write gate was not asserted
BIT 13 Write lock - winchester drive is write protected
BIT 12 Seek timeout - winchester drive did not complete a seek in the allotted time
BIT 11 Speed error - winchester drive not ready
BIT 10 Write gate error - set when write fault is set in winchester drive
BIT 9 Always 0

BIT 8	Drive select error - attempt was made to select a non-existent drive
BIT 7	Set if DSD 880 is in extended mode
BIT 6	Head Select - this bit specifies the head currently selected (0 or 1)
BIT 5	Always 0
BIT 4	Heads out - always 1
BIT 3	Always 1
BITS 0 - 2	STA, STB, and STC - states A, B, and C

These bits define the current state of the winchester drive as follows:

C	B	A	State Specified
0	0	0	Load
1	0	0	Seek
1	0	1	Lock On

2. Multipurpose Register During a Read Header Command

Execution of a read header command loads three words into the MP register. The first word contains the sector address, head select, and cylinder address information. The second word is all zeroes. The third word contains the header CRC data. Figure 5-9 shows the format for each word. The MP register may be read only when bit 7 (CRDY) of the CS register is set.

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
1ST Word	CA8	CA7	CA6	CA5	CA4	CA3	CA2	CA1	CA0	HS	SA5	SA4	SA3	SA2	SA1	SA0

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
2ND Word	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
3RD Word	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

TP 115/81

Figure 5-9. Multipurpose Register Format for a Read Header Command

3. Multipurpose Register for a Read/Write Data Command

The multipurpose register acts as a word counter when the drive is reading or writing data. Initially, the MP register is loaded with the two's complement of the number of words to be transferred. Word counter overflow normally terminates the read or write operation. Figure 5-10 shows the MP register during a read/write data command in both standard and extended operating modes. The largest valid word count for the normal mode is 5120 words. The longest valid word count for the extended mode (where a spiral read/write is allowed) is 65536 words.

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
WC15	WC14	WC13	WC12	WC11	WC10	WC9	WC8	WC7	WC6	WC5	WC4	WC3	WC2	WC1	WC0

TP 116/81

Figure 5-10. Multipurpose Register Format for a Read/Write Data Command

5.4.10 Winchester Controller Commands

The winchester disk drive commands to the controller are specified by bits 1, 2, and 3 of the control status (CS) register.

Function Code 0 - NOP

The drive clears errors (except for a drive error, DE in the CS register), sets the controller ready (CRDY) bit in the CS register, and causes an interrupt if interrupts are enabled (IE is set).

Function Code 1 - Write Check Command

Write check verifies that data were accurately written on the disk in the following manner. The write command writes a block of data from the data buffer in main memory onto the disk. Then the write check reads that block of data from the disk and serially compares it with the original data in the data buffer. Note that this comparison occurs in the controller which requires a source data transfer from memory into the controller data buffer.

Before executing the write check command, initialize the bus address (BA), multipurpose (word count), and disk address registers as follows:

Register	Contents
Bus Address	Address of first data block in main memory
Multipurpose (Word Count)	Length of the data block
Disk Address	Starting disk address location

Immediately, the DMA transfer of data from the main memory data buffer to the controller begins. The logical RL01 disk address is mapped onto a physical winchester disk address and header address words, read from the disk, are compared to the starting physical address.

As soon as the starting address is found, the controller is monitored until it contains a complete sector. If there are no header cyclic redundancy check (HCRC) errors, the data (128 words) are then read from the disk and compared to the data in the controller's data buffer. An error in this comparison, or in the data cyclic redundancy check, sets the DCRC bit in the control status register.

Function Code 2 - Get Status Command

Upon execution of the get status command, the drive sends the drive status word to the controller if the get status bit (bit 1) in the disk address register is set. The get status command loads the drive status word into the multipurpose register. The controller sets CRDY (controller ready) and causes an interrupt, if interrupts are enabled (IE set). Note that if bit 3 (RST, the reset bit) of the DA register is set, the drive first clears the error bits then sends the status word.

If the get status bit in the DA register is clear, the get status command is undefined and an error is repeated.

Function Code 3 - Seek Command

On executing the seek command, if DAO in the DA register is set and DA1 is clear, then on receiving the seek information the controller sets CRDY and, if interrups are enabled (IE set), causes an interrupt. The seek information includes the head direction, head select, and cylinder address difference. When the drive receives the seek information from the controller, it seeks and/or selects a new read/write head. DA0 must be set and DA1 clear for a seek command; any other combinations are undefined, and an error is repeated.

If the size of the cylinder address difference would move the heads beyond permissable limits (inside the innermost track or beyond track 0), the head stops at the limit track. A maximum length seek out may therefore be used as restore command.

Function Code 4 - Read Header Command

This command finds the current location on the disk as follows. If CRDY (controller ready) is clear, a read header command causes the controller to read the current disk location into the multipurpose register. The controller then sets CRDY and, if interrupts are enabled (IE is set), causes an interrupt. To obtain the two header words, the software reads the MP register contents for the current cylinder, head, or sector location of the drive, then calculates the cylinder address difference for a seek operation.

The header cyclic redundancy check (HCRC) word enters the silo behind the two header words, to be available from the MP register for diagnostic use.

Function Code 5 - Write Data Command

This command moves the head to the correct location and writes the required data as follows. If CRDY is clear, a write data command causes the controller to map the logical RL01 disk address onto a physical winchester disk address. It then reads and compares successive header words with the physical disk address (DA) register until an address match is found. Then the header cyclic redundancy check (HCRC) occurs and, if there is no HCRC error, the data specified by the bus address (BA) register are written into the sector. If the data does not fill the sector, zeroes are written in the remaining locations.

If the amount of data requires any additional sectors, the sector address in the DA increments when the current sector is full, then the write continues in the next sector. Completion of the data transfer sets CRDY and, if interrupts are enabled (IE is set), causes an interrupt.

Function Code 6 - Read Data Command

This command moves the head to the correct location and reads the required data as follows. If CRDY is clear, the read data command causes the controller to map the logical RL01 disk address onto a physical winchester disk address and read and compare successive header words with the required disk address (DA) word in the DA register until a match occurs. If there are no header cyclic redundancy check (HCRC) errors, the data in the sector are read into the location specified by the contents of the bus address (BA) register. A data cyclic redundancy check (DCRC) occurs. If there are no errors, the contents of the DS increment by one. If the word count (contents of the multipurpose register) overflows, CRDY sets. If interrupts are enabled (IE is set), an interrupt occurs. If the MP register does not overflow, the read continues with the next sector.

Function Code 7 - Read Data Without Header Check Command

If CRDY is clear, a read data without header check command reads the data from the next sector to the location specified by the contents of the bus address (BA) register. The DCRC occurs at the end of the sector. Then, if the word count (in the multipurpose register) has not overflowed, the read continues at the next sector. The word count overflow sets CRDY and, if interrupts are enabled, an interrupt occurs.

Note that the header is not compared or checked for cyclic redundancy errors with this command. The read data without header check command is normally used by issuing read header commands until the sector prior to the desired sector is found, then issuing the read data without header check command.

6.0 BASIC CIRCUIT DESCRIPTION

6.1 General Information

This section provides a basic, block diagram level description of the DSD 880x/8 circuitry.

6.2 DSD 8832 Interface Board

The DSD 8832 is the interface between the DSD 880x/8 System and the DEC LSI-11 processor. The DSD 8832 interface board performs several functions, the primary ones being:

- Emulation of RL01 and RL02 control and status registers.
- Control of the data transfer between the DSD 880x/8 interface bus and the LSI-11 Q-bus.
- Contains the user selectable RL01 and RX02 bootstrap program.
- Arbitrates RL01 and RX02 command transfers between the DSD 880x/8 controller and the LSI-11 processor.

The unique capability of the DSD 880x/8 to emulate both a RL01 and a RX02 in a single cost effective package is due in part to the ability of the interface to arbitrate between RL01 and RX02 commands.

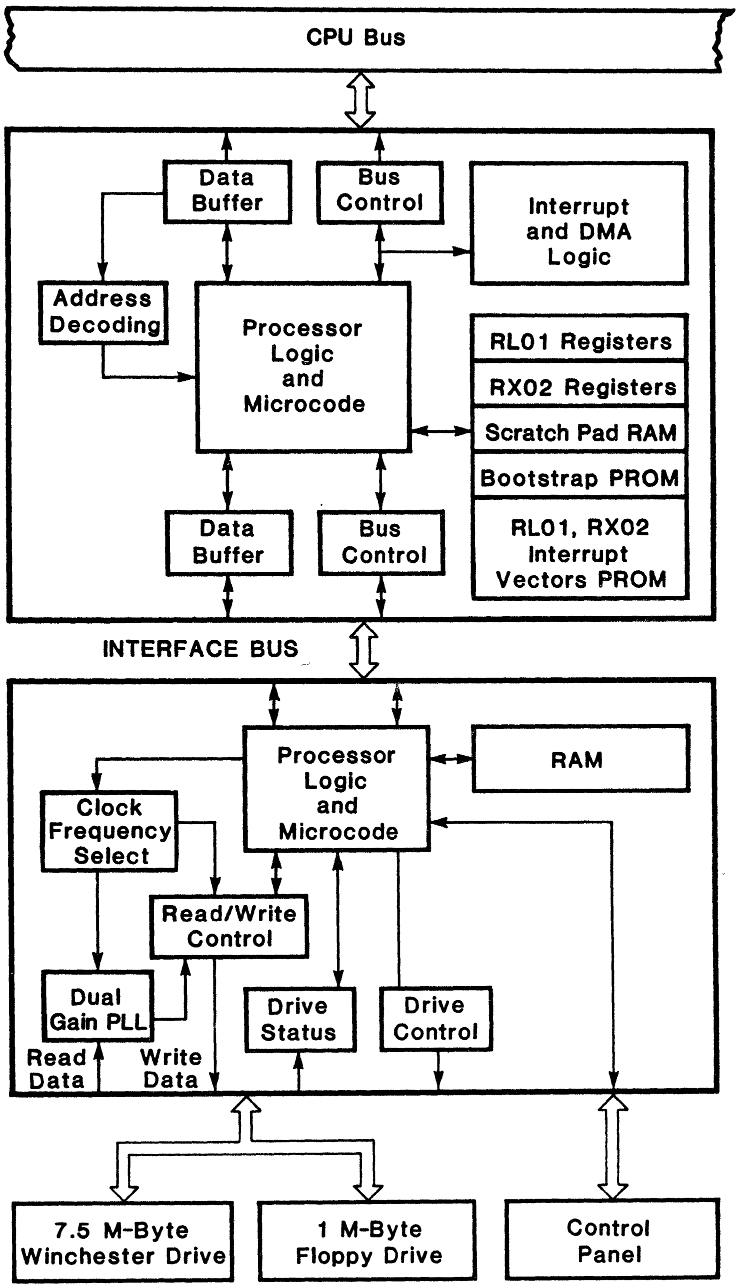
Although the DSD 880x/8 system controller emulates both a single-drive RL01 and a single-drive RX02 disk system, it cannot do so simultaneously. In order to maintain system compatibility and resolve device conflicts, the DSD 8832 interface arbitrates command transfers in the following manner.

Assume that, initially, neither the RL01 or RX02 is executing a command and a command is received by the interface for the RX02 device. The command will immediately be sent to the DSD 880x/8 controller for execution, and the done bit in the RX2CS will be cleared. If a command is received for the RL01 device before the RX02 command has completed execution, the interface will accept the command, place it in a one level queue for transfer to the controller, and clear the controller ready bit in the CSR. At this point, both devices will appear busy.

When the RX02 device completes execution, the interface will set the done bit in the RX2CS register and immediately send the queued RL01 command to the controller for execution. If a new command is received for the RX02 device before the RL01 command completes execution, it will be placed in the one level queue and the done bit will be cleared.

When the controller completes execution of the RL01 command, the interface will set the controller ready bit in the CSR. If a command is in the queue for the RX02 device, it will be executed. Otherwise, both devices will be ready to accept new commands.

The DSD 8832 interface has been implemented using bipolar technology in order to provide the desired fast LSI-11 response time and DMA throughput. Refer to the block diagram and the DSD 880x/8 shown in Figure 6-1. Note the logic of the interface can be divided into three major subsections; processor and associated logic, LSI-11 Q-Bus interface, and DSD 880x/8 I-Bus interface.



CPU INTERFACE

- Emulates RL01 and RX02 command and status registers
- Arbitrates RL01 and RX02 command transfers between DSD880 controller and CPU
- Controls data transfer between DSD 880 controller and CPU
- Contains user selectable RL01 and RX02 bootstrap programs
- Contains DMA and interrupt logic

DSD 880 CONTROLLER/FORMATTER

- Directy emulates RL01 and RX02 hardware and software operations
- Controls data transfer to and from disk drives
  - Encoding decoding
  - Formatting
  - Implied seeks
  - Multiple sector transfers
  - Bad track remapping
- Executes self diagnostics

CONTROL PANEL

- Selection display
  - Diskette formatting
  - Backup loading
  - Fault indication
  - Write protection
  - System diagnostics

TP 117/81

Figure 6-1. DSD 880x/8 Block Diagram

The processor subsection forms the intelligent heart of the interface. It consists of the processor logic (ALU, sequencer, etc.), the microcode PROM, and the RAM data buffer. The processor subsection controls data and command transfer between the LSI-11 Q-Bus and the DSD 880x/8 controller I-Bus, implements the device registers, and performs RL01 and RX02 command queing. Note that the command and status reqisters for the RL01 and RX02 devices are implemented in software using the RAM data buffer rather than as discrete hardware registers.

The LSI-11 Q-Bus interface subsection consists of the device address decorder, the interrupt logic, Q-Bus register, and Q-Bus buffers. This subsection controls the transfer of data between the processor subsection and LSI-11 Q-Bus. The address decoder recognizes jumper selectable RX02 and RL01 device and bootstrap addresses. The Q-Bus register stores data, and address and status information while it is being transferred to the LSI-11 processor via the Q-Bus. The interrupt request logic and interrupt vector PROM control the interrupt of the LSI-11 processor by the processor subsection. The desired interrupt vector and level are jumper selectable.

The DSD 880x/8 I-Bus interface subsection consists of the I-Bus register, I-Bus controller, and I-Bus buffers. This subsection controls the transfer of data between the processor subsection and the DSD 880x/8 controller I-Bus. The I-Bus register allows the transfer of data between the controller and interface to be as rapid as possible without exceeding the capability of either. The I-Bus controller coordinates the transfer of data into and out of the I-Bus register while the I-Bus buffers match the I-Bus cable to the logic requirements of the I-Bus interface.

6.3 DSD 8830 Interface Board

The DSD 8830 interface board is available for those customers utilizing the DSD 880x/8 Data Storage System with the DEC PDP-11 processor. The DSD 8830 controls data transfer between the PDP-11 Unibus and the DSD 880x/8 interface bus.

The 8830 can emulate both RX02 and RL01 device registers according to DEC standards. Since the 880 controller can only operate on one device at a time, the 8830 arbitrates between sending the latest RL command the bootstrap eliminates the need for a DEC bootstrap board. Finally, five switch packs allow the user to select any of the possible boot addresses, device register addresses, or vector addresses.

Basically, the 8830 is a simple bit slice or nibble machine. A straight forward micro-instruction set can be derived since the ALU A input is designated for straight 128X4 RAM nibbles. The ALU B input is selected through the ALU MUX, the ALU F output is latched into the RAM (AO register) and/or buffer register A. The 2911 based micro-instruction sequencer allows JMP, JSR, and RTS type branches. A high 880 to Unibus throughput rate during DMA is enhanced by the two 16-bit data buffer registers A and B which can be parallel loaded, or nibble shifted, in a way that allows the 880 to read or write data through register B while the rest of the 8830 operates through register A.

6.4 DSD 880x/8 Controller/Formatter Board

The processor logic, which is the heart of the DSD 880x/8 controller, is made up of 2901 bit slice logic circuitry. It performs the following basic functions:

- Handles the I-Bus protocol between the interface and the controller.
- Executes DEC-compatible RL01 and RX02 command sets.
- Executes seek, head load, read, write, and other disk drive related functions.
- Handles data flow to and from the interface and the read/write circuitry.
- Provides format control.
- Controls the diagnostic front panel.
- Executes HyperDiagnostics.

The phase-lock-loop circuitry consists of dual front-end phase comparators with their associated low pass filters and a common voltage controlled oscillator. The use of a dual gain approach provides extended margins of acquisition and tracking range. It is used to:

- Discriminate preamble for winchester data.
- Reconstruct clock and data margins from raw data.

A sophisticated clock system is used to synchronize the processor logic with the read/write format control circuitry. The system uses three clock sources:

- A 6 Mhz crystal for floppy write and system housekeeping functions.
- A 17.36 Mhz crystal for floppy read, winchester write and other critical timing functions.
- A VCO for floppy and winchester read.

The heart of the read/write format control circuitry is a 82S100 FPLA. The circuitry is used to:

- Encode and decode FM and DEC-modified MFM formats for the floppy disk.
- Encode and decode MFM format for the winchester disk.
- Check the CRC of header and data fields.
- Provide proper precompensation for both the floppy and winchester drives.

The DSD 8840 is shown in Figure 6-2, and the 8841 in Figure 6-3.

6.5 DSD 8833 HyperDiagnostic Panel

The DSD 8832 HyperDiagnostic panel provides user access to controller functions and status indicators. These functions include:

- System mode selection.
- Backup and load operations.
- Diskette formatting.
- Write protection for both the floppy and winchester drives.
- HyperDiagnostic test selection.
- Fault and status indication.

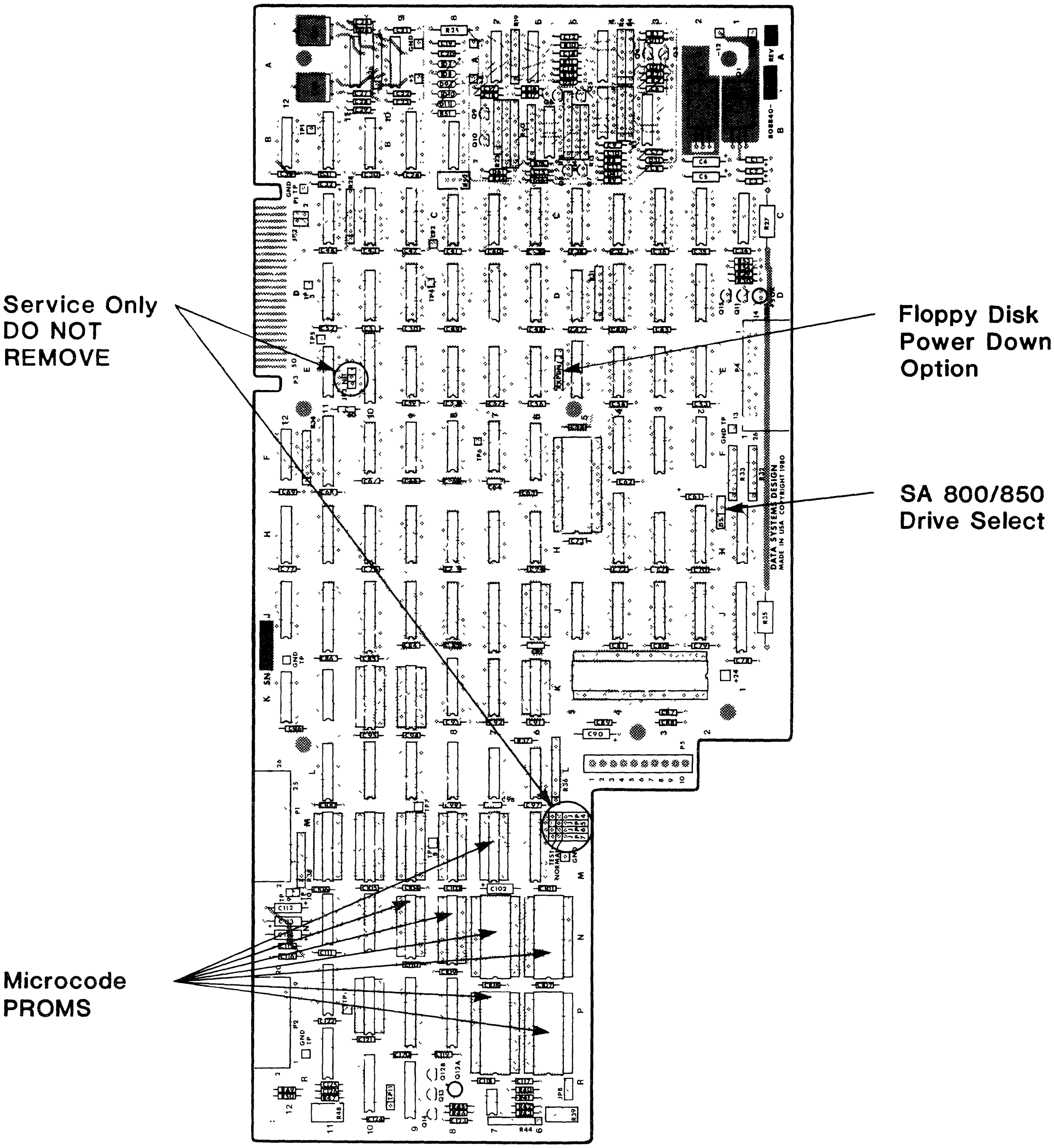


Figure 6-2. DSD 8840 Controller/Formatter Board

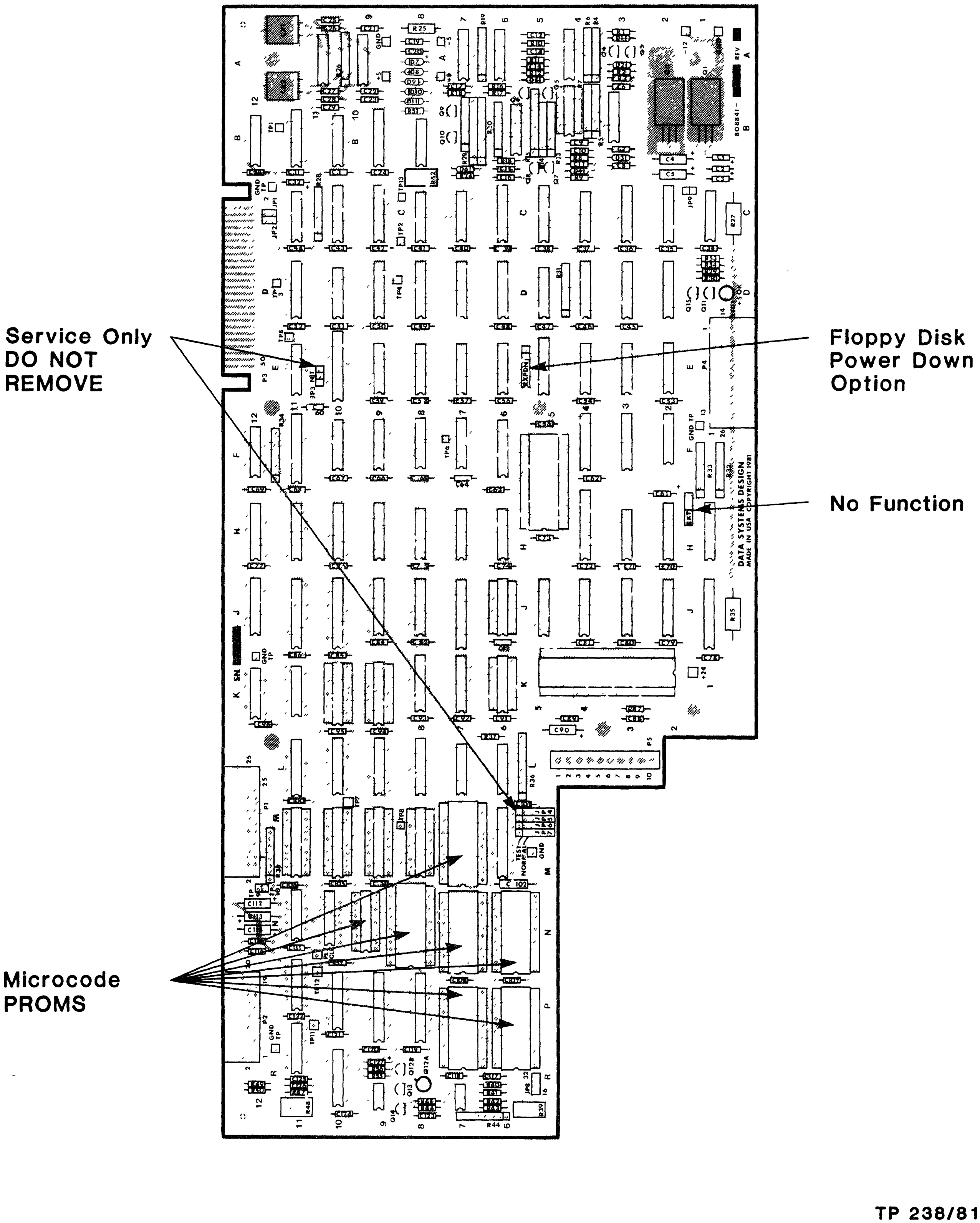


Figure 6-3. DSD 8841 Controller/Formatter Board

7.0 USER LEVEL MAINTENANCE

7.1 General Information

This section provides information on the maintenance of the DSD 880 Data Storage System. The first part discusses the routine procedures required to maintain the equipment at its peak efficiency. The second part provides basic troubleshooting and fault isolation techniques to be utilized in quickly locating the portion of the system causing a problem.

7.2 Preventive Maintenance

The DSD 880 is designed to minimize the amount of periodic maintenance required. The prime factor in maintaining electronic equipment is ensuring that it is operated within its design parameters and specified environmental limits. (See Section 2.) Cleanliness should be considered as part of the environmental requirement.

During any routine or scheduled maintenance, the first step should always be a visual inspection. Check for corrosion, dirt, and undue wear on moving parts. Check all connector assemblies for proper and firm installation.

7.2.1 Floppy Disk Drive Preventive Maintenance

Preventive maintenance schedules for the floppy disk drives furnished with the DSD 880 system are provided in Tables 7-1 and 7-2. The maintenance intervals specified are considered minimum for normal usage and may be changed to more frequent intervals as determined by the user. Any maintenance or adjustments beyond those specified should be attempted only by qualified technicians using procedures outlined in the service manual for the drive.

1. SA800 Single-Sided Drive:

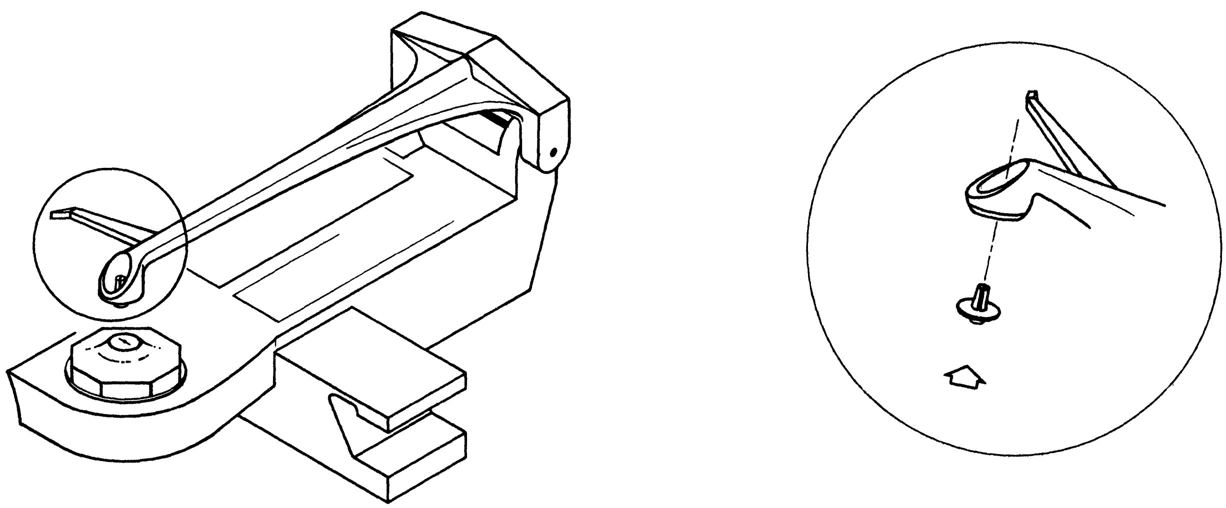
Table 7-1. Single-Sided Floppy Drive Preventive Maintenance

Unit	Frequency (Months)	Action	Observe
Read/Write Heads	6	Clean read/write head ONLY IF NECESSARY	Oxide build up
Read/Write Load Head Button	6 *	Replace as necessary	Color; bright red =OK, pink = replace
Stepper Motor and Lead Screw	6	Clean off all oil, dust, and dirt	Inspect for nicks and burrs
Belt	6		Inspect for frayed or weakened areas
Base	6	Clean base	Inspect for loose screws, connectors, and switches

*Assumes normal usage

A. Read/write head load button removal and replacement procedure:

- To remove the old button, hold the arm out away from the head, squeeze the locking tabs together with a pair of needle nose pliers and press forward.
- To install the new load button, press the button into the arm from the head side; it will snap into place. See Figure 7-1.



TP 242/81

Figure 7-1. Removal and Replacement of Head Load Button

CAUTION

To prevent damage to the torsion spring, the load arm should never be opened over 90^{o} from the carriage assembly, or while at track 00.

B. Single-sided drive cleaning procedure:

Single-sided heads can be cleaned using a clean cotton swab and a solution of at least 90% isopropol alcohol. Take care that none of the solution gets on the head load pad.

2. SA850 Double-Sided Drive:

Table 7-2. Double-Sided Drive Preventive Maintenance

Unit	Frequency (Months)	Action	Observe
Read/Write Heads	6	Clean read/write head ONLY IF NECESSARY	Oxide build up
Actuator Band	6	Clean off all oil, dust, and dirt	
Belt	6	Replace if damaged	Inspect for frayed or weakened areas
Base	6	Clean base	Inspect for loose screws, connectors, and switches

A. Double-sided drive cleaning procedure:

Use the approved head cleaning diskette, Innovative Computer Products P/N 2024, or DSD P/N 530010 for the SA850 double-sided drive. The cleaning kit comes with diskettes, fluid, and full instructions for use.

CAUTION

A perforated tab is removed from the diskette for use in cleaning double-sided drives. Use of this same diskette for cleaning heads in single-sided drives will cause damage to the heads.

7.2.2 Winchester Drive Preventive Maintenance

The winchester drives used with the DSD 880 systems require no preventive maintenance.

7.2.3 Power Supply Preventive Maintenance

Preventive maintenance of the DSD 880 power supply consists of checking the dc voltages at test jacks provided on the HyperDiagnostic panel. A digital voltmeter is required to check these voltages. This routine should be performed at six month intervals. Proceed as follows:

1. Turn off the power to the DSD 880 chassis and remove the front bezel to gain access to the HyperDiagnostic panel.

2. Set range and function controls on the voltmeter to read +5 Vdc.

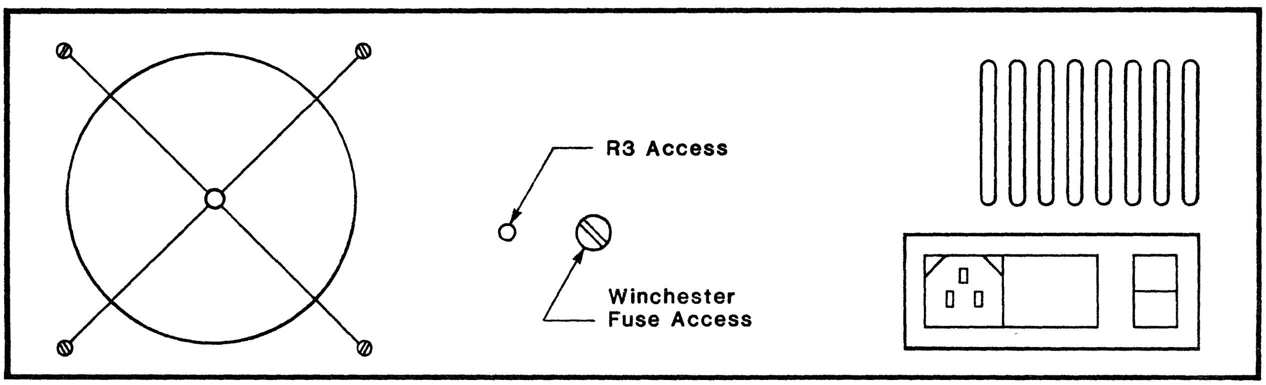
3. Connect meter to the +5 and ground test jacks on the HyperDiagnostic panel. Observe meter polarity.

4. Turn on chassis power and verify meter reading of +5 Vdc $\pm$ 0.1 Vdc.

   A. If reading is not within tolerance, adjust R3 trim pot on rear panel of main chassis to bring the voltage within specification. See Figure 7-2.

CAUTION

The ac fuse for the winchester drive is located on the rear panel adjacent to trim pot R3. Use extreme caution during adjustment of R3 to avoid the inadvertent release of the fuse with the power on. Inadvertent release of the winchester drive ac fuse will cause damage to the drive and void warranty.



TP 220/81

Figure 7-2. Location of R3, +5 Vdc Trim Pot

5. Disconnect the meter from the +5 test jack, and connect it in turn to measure the +24 Vdc and -12 Vdc voltages. These checks are made after the +5 Vdc is verified within tolerance.

   A. Verify +24 Vdc $\pm$ 3 Vdc (+21 to +27 Vdc) meter reading.

   B. Verify -12 Vdc. A reading of from -8 to -18 Vdc is acceptable.

6. Disconnect meter and reinstall the front bezel covering the HyperDiagnostic panel.

NOTE

If difficulties arise during the performance of any of the preventive maintenance routines contact your Customer Service Representative for assistance.

7.3 Troubleshooting and Fault Isolation

The following list of diagnostic tools should be used to assist in the isolation of faults in the system.

1. Built in self-tests.
2. Error lights and indicators on the HyperDiagnostic panel.
3. Power supply test points on the HyperDiagnostic panel.
4. Internal controller registers indicating the status of the RX and RL devices. Refer to Section 5 for instructions on recovering register data.
5. Halts and loops in the bootstrap program.
6. HyperDiagnostic routines.
7. FLPEXR, RLEXR, and WINEXR diagnostic programs.
8. DEC diagnostics.
9. DSD Customer Service.

Table 7-3 is furnished for initial, user level, fault isolation on the DSD 880. This guide should be used as a preliminary check list prior to any extensive maintenance procedures.

Table 7-3. Preliminary Troubleshooting Guide

Trouble Indication	Possible Cause
DSD 880 floppy disk and/or winchester disk will not operate	• Power switch not turned on • Power cord is disconnected • Interface cable improperly installed • Fuse blown • Overheated condition
Floppy disk drive activity lights do not light. Disk drives do not initialize	• Power supply failure • Floppy disk drive door open • Diskette improperly loaded into floppy disk drive
Floppy disk drive activity light remains lit at all times	• Defective or empty drive • Defective controller • Attempted boot on blank diskette
Disk drive will not initialize	• Defective interface, power supply, controller, or drive. Halt switch on computer is set to on

Table 7-3. Preliminary Troubleshooting Guide (Cont)

Trouble Indication	Possible Cause
Bootstrapping cannot be performed	• Interface cable improperly installed • Interface cable improperly installed at computer backplane • Defective interface • Halt switch on computer front panel is set to on • Possible drive malfunction • Bus grant continuity broken • DMA grant jumper not removed • DMA grant chain broken • Diskette not bootable

7.4 Use of DSD 880 HyperDiagnostics

The DSD 880 provides diagnostic aid in the form of the built-in, microcoded, HyperDiagnostic mode of operation. Added diagnostic assistance is available through use the DSD RLEXR, FLPEXR, and WINEXR programs.

If the preliminary troubleshooting guide, Table 7-3, fails to locate the cause of the system malfunction, the built-in diagnostic capabilities of the DSD 880 should be used to isolate the fault to a replaceable subsystem (interface card, controller board, floppy disk drive, winchester disk drive, interface cable, or power supply).

The DSD 880 Data Storage System provides the user with extensive built-in self-test features, HyperDiagnostics, which permit testing of the system without requiring the use of a computer. The HyperDiagnostics are a series of routines in microcode which self-test the 8841 controller and exercise both the floppy and winchester disk drives. The tests are initiated and monitored from the HyperDiagnostic panel, located behind the front bezel.

The following MODES may be selected:

0 Normal and direct access modes, and selection of write protected RL logical units.

1 Floppy disk format routines, used to format the floppy disk in single or double density, with or without rewriting headers, or scan verification.

2 General exerciser tests of the floppy disk, the winchester disk, or both; used to verify proper system operation.

3 Controller hardware tests, which do not exercise the drives.

4 Floppy disk alignment routines.

5 Individual tests of the floppy and winchester drives; used mostly for troubleshooting.

6 Reload winchester disk from backup floppy disks.

7 Backup winchester disk data onto floppy disks. Selects floppy option flag.

CAUTION

> Any test that causes data to be written on the winchester disk can cause loss of data that are on the disk prior to testing.

7.4.1 HyperDiagnostic Operation

DSD 880 HyperDiagnostics are initiated by selecting the appropriate MODE and CLASS switch settings and momentarily depressing the EXECUTE pushbutton. The selected MODE and CLASS is echoed by the seven segment displays while the EXECUTE pushbutton is depressed.

If a floppy disk is required for the HyperDiagnostic, it must be inserted prior to inititating the test. Otherwise, a drive error (36) will be reported. Likewise, if the HyperDiagnostic includes a write operation, the appropriate drive(s) must be write enabled. Otherwise, a write protect error (10) will be reported.

Most HyperDiagnostics display the selected CLASS and MODE while the test is running. If the test fails, the appropriate error code and fault indicators will be flashing. If the selected HyperDiagnostics is a single-pass test, the code 00 will be displayed upon successful completion. If the HyperDiagnostic selected is repetitive, the code 00 will be displayed for one second between each pass.

Most HyperDiagnostics can be terminated at any time by selecting the new HyperDiagnostic test code and depressing the EXECUTE pushbutton. The floppy disk format HyperDiagnostics cannot be terminated via the EXECUTE pushbutton and must be allowed to complete before selecting a new test.

Since the HyperDiagnostics are controlled by microcode, the microprocessor in the DSD 880 must be at least partially functioning before any tests can be run. HyperDiagnostics do not perform any tests on the interface board or on the I-Bus cable. It is not necessary to have the I-Bus cable connected while running HyperDiagnostics. In most cases it is better to disconnect the I-Bus cable to prevent computer system activity from affecting test results. In particular, bus initialize from the computer will always abort HyperDiagnostics.

7.4.2 Error Reporting During HyperDiagnostics

Errors are indicated by displaying the appropriate error code in the seven segment displays and illuminating the composite and appropriate drive fault indicators located on the HyperDiagnostic panel. Table 7-4 lists the DSD 880 definitive error codes. Paragraph 7.5 provides an expanded definition of the error codes.

Errors other than header or data CRC (14 or 20) errors will cause the HyperDiagnostics routine to terminate immediately upon their occurrence. Each occurrence of the CRC error is logged and a running total kept. The HyperDiagnostic will terminate when a total of 16 (decimal) CRC errors have occurred since the HyperDiagnostic was initiated.

Table 7-4. Definitive Error Codes

These errors are flashed on the HyperDiagnostic panel when the indicated error occurs:

CODE Displayed	Description
00	No errors - operation complete (HyperDiagnostics only)
01	Drive failed to home on initialize
02	Nonexistent drive
03	Track 00 found while stepping in on initialize
04	Invalid RX02 track address
05	Track 00 found before desired track while stepping
06	Seek timeout while stepping (RL01 only)
07	Requested sector not found in two revolutions
10	Write protect violation
11	Not defined
12	No preamble found
13	Preamble found, but no address mark within window
14	CRC error on what appeared to be a header
15	Address in header did not match desired track
16	Too many tries for an ID address mark
17	Data address mark not found in allotted time
20	CRC error on data field
21	Write gate error (RL01 only)
22	VCO failure during read operation (RL01 only)
23	Invalid word count specified
24	Media density did not match desired density (RX02 only)
25	Invalid key for set media density or format command (RX02 only)
26	Indeterminate media density (RX02 only)
27	Write format failure
30	Data compare error (RL01 and read/write HyperDiagnostics)
31	Invalid bad track map detected during intialize (RL01 only)
32	Bad track map checksum did not match stored value
33	Not defined
34	Not defined
35	Nonexistent memory (NXM) error during DMA transfer
36	Drive not ready (door open, speed error, or absent media)
37	Low ac power caused abort of write activity
40	Invalid disk used for reload (RL01 reload only)
41	Multiple reload disk versions used (RL01 reload only)
42	Invalid class selected (HyperDiagnostics only)
43	Invalid winchester disk address
44	Winchester disk word count overflow
45	Deleted data mark encountered on reload floppy (RL01 reload only)
46	Invalid backup mode
47	Confirmation of intent to reconfigure floppy
51	Memory test failure
52	CRC test failure
53	PLL test failure

7.4.3 Winchester Write Enable

HyperDiagnostics which include a winchester disk sequential write operation must be write enabled prior to initiating the test. Write enable is accomplished by selecting CLASS 7 of the appropriate MODE (2 or 5), then depressing the EXECUTE pushbutton. The selected MODE will then be write enabled and will remain so until a new MODE is selected. Note that winchester read/write HyperDiagnostics destroy data on the winchester disk.

7.4.4 Floppy Disk Format Routines (MODE 1)

The floppy disk format routines are entered by setting the MODE switch to position 1 (format), selecting the desired CLASS, and depressing the EXECUTE pushbutton. These routines affect only the floppy disk drive; it is not possible to format the winchester drive from the HyperDiagnostic panel. The floppy write protect switch must be off, and a write enabled floppy disk must be placed in the drive. All data on the floppy disk will be lost. Either single- or double-sided disks may be used. Unlike most HyperDiagnostics, it is not possible to interrupt the operation by pressing the EXECUTE pushbutton during the test. This prevents mixed-density diskettes from being created.

The following CLASSES may be selected:

0 FORMAT DOUBLE-DENSITY - formats the entire floppy disk in DEC double-density format. Headers are rewritten.

1 FORMAT SINGLE-DENSITY - formats the entire floppy disk in DEC/IBM single-density format. Headers are rewritten.

2 SET MEDIA DOUBLE-DENSITY - writes all data fields in DEC double-density format, with all data bytes = 0. Headers are not rewritten.

3 SET MEDIA SINGLE-DENSITY - writes all data fields in DEC single-density format, with all data bytes = 0. Headers are not rewritten.

4 SET MEDIA DOUBLE-DENSITY AND SCAN - writes all data fields in DEC double-density format and scans the disk looking for errors.

5 SET MEDIA SINGLE-DENSITY AND SCAN - writes all data fields in DEC single-density format and scans the disk looking for errors.

7.4.5 System Tests (MODE 2)

The system tests are entered by setting the MODE switch to position 2 (system), selecting the desired CLASS, and depressing the EXECUTE pushbutton. The tests are normally used to verify that the 880 system is working correctly, rather than for troubleshooting. The tests exercise the 8840 controller and one or both disk drives, but do not test the interface card or the I-Bus cable. These tests are useful for verifying system operation during incoming inspection and after site installation of the system.

The following CLASSES may be selected:

0 FLOPPY DISK EXERCISER WITH WRITE FORMAT - runs the following sequence of HyperDiagnostic tests on the floppy drive only:

a. Single-Density Write Format
b. Sequential Scan All Sectors
c. Butterfly Read Headers
d. Sequential Write/Read All Sectors
e. Set Media Double-Density
f. Sequential Scan All Sectors
g. Butterfly Read Headers
h. Sequential Write/Read All Sectors
i. Set Media Double-Density

1 FLOPPY DISK EXERCISER WITHOUT WRITE FORMAT - runs the same sequence of tests as the floppy disk exerciser described previously, with the exception of the single-density write format.

2 FIXED DISK EXERCISER - runs the following sequence of HyperDiagnostic tests on the fixed disk drive only:

a. Sequential Scan All Sectors
b. Butterfly Seek Test
c. Sequential Write/Read All Sectors

3 GENERAL EXERCISER WITH FLOPPY DISK WRITE FORMAT - runs the floppy disk general exerciser, then runs the fixed disk exerciser tests.

4 SINGLE-PASS GENERAL EXERCISER WITH FLOPPY WRITE FORMAT - runs a single pass of the floppy and fixed disk exercisers.

5 SINGLE-PASS GENERAL EXERCISER WITHOUT FLOPPY WRITE FORMAT - runs a single pass of the floppy and fixed disk exercisers without formatting the floppy disk.

6 GENERAL EXERCISER WITHOUT FLOPPY WRITE FORMAT AND FIXED READ/WRITE TESTS - runs the floppy disk general exerciser without formatting the floppy disk, then runs the fixed disk exerciser without executing the sequential write/read tests.

7 FIXED DISK EXERCISER WRITE ENABLE - permits sequential write operations on the winchester disk. (For tests 2, 3, 4, and 5.)

7.4.6 Controller Tests (MODE 3)

The controller tests are entered by setting the MODE switch to position 3 (controller), selecting the desired CLASS, and depressing the EXECUTE pushbutton. The tests are intended for troubleshooting the controller logic to determine if a problem is drive related.

The following CLASSES may be selected:

0 SWITCH AND INDICATOR TEST - tests the various controller switches and indicators on the diagnostic panel for proper operation.

Setting the floppy write protect switch to the on position will illuminate the floppy write protect and floppy fault indicators, and cause the digits 88 to flash in the seven segment displays.

Setting the winchester write protect switch to the on position will illuminate the winchester write protect and winchester fault indicators, and cause the digits 99 to flash in the seven segment displays.

If neither the floppy or winchester write protect switches are in the on position, the winchester fault, floppy fault, floppy write protect, composite fault, and winchester ready indicators will be sequentially illuminated one at a time. In addition, the position of the CLASS and MODE switches will be echoed in the seven segment displays.

1 GENERAL CONTROLLER HARDWARE TEST - runs the following controller hardware diagnostics:

- a. ALU logic test
- b. RAM memory test
- c. CRC logic test
- d. PLL logic test

This test verifies the controller hardware and is useful in localizing failure to a specific functional block.

2 ALU LOGIC TEST - tests the operation of the arithmetic logic unit.

3 RAM MEMORY TEST - tests the operation of the RAM buffer memory.

4 CRC LOGIC TEST - tests the operation of the CRC logic.

5 PLL LOGIC TEST - tests the operation of the phase-locked-loop circuit.

6 MICROCODE VERSION - displays microcode version number.

7.4.7 Floppy Disk Alignment Routines (MODE 4)

The floppy disk alignment routines are entered by setting the MODE switch to position 4 (align floppy), selecting the desired CLASS, and depressing the EXECUTE pushbutton. These routines affect only the floppy disk drive and are intended for use by qualified service personnel when an alignment disk (DYSAN part number 360-2A or DSD part number 530003) is used to adjust the drive. The following CLASSES may be selected:

0 FLOPPY DISK TRACK 00 DETECTOR ADJUSTMENT - loads floppy head and repeatedly seeks between tract 00 and 01 every 100 ms.

1 FLOPPY DISK SEEK TRACK 01 AND LOAD HEAD - seeks floppy head to track 01 and loads it.

2 FLOPPY DISK SEEK TRACK 02 AND LOAD HEAD - seeks floppy head to track 02 and loads it.

3 FLOPPY DISK SEEK TRACK 38 AND LOAD HEAD - seeks floppy head to track 38 and loads it.

4 FLOPPY DISK SEEK TRACK 76 AND LOAD HEAD - seeks floppy head to track 76 and loads it.

5 FLOPPY DISK HEAD LOAD TIMING ADJUSTMENT - seeks floppy head to track 00 then alternately loads head for 100 ms and unloads head for 200 ms.

7.4.8 Read/Write Tests (MODE 5)

The read/write tests are entered by setting the MODE switch to position 5 (read/write), selecting the desired CLASS, and depressing the EXECUTE pushbutton. These routines are intended for troubleshooting of problems encountered during computer system operation, or during the system mode HyperDiagnostics. They consist of individual read, write, scan, and seek tests on both the floppy and winchester drives. Write protect switches should be off. A disk must be inserted in the floppy disk drive if tests are being performed on that drive. Single- or double-sided floppy disks of either density may be used. Data on the effected disk will be lost if the sequential write/read test is run. The following CLASSES may be selected:

0 SINGLE-PASS SEQUENTIAL SCAN FLOPPY DISK - scans the entire disk for CRC errors and valid disk headers. Data on the floppy disk is not affected. This test is extremely useful, if a system disk cannot be booted, to check for errors on the disk. The test stops after one pass is made.

1 BUTTERFLY SEEK TEST FLOPPY DISK DRIVE - steps head of floppy disk drive using a butterfly pattern, then seeks track 00. This test is used to detect head positioning problems in the floppy disk drive. The test runs until halted. Note that this test can be run without media in the floppy drive.

2 BUTTERFLY READ HEADERS ON FLOPPY DISK - steps head of floppy disk drive using a butterfly pattern, checking for correct disk headers. This test is similar to the butterfly seek test except that head positioning is verified by comparing the track number, in the disk header, to a expected track number. The test runs until halted.

3 SEQUENTIAL WRITE/READ FLOPPY DISK - sequentially writes then reads the entire floppy disk checking for data or header errors. This test exercises the read/write circuitry of the controller and floppy disk drive, and is useful in diagnosing problems in this area. The test runs until halted.

4 SEQUENTIAL SCAN FIXED DISK - scans entire fixed disk for CRC errors and valid disk headers. Data on the disk are not changed by this test. This test is useful in verifying the winchester disk media when intermittant CRC errors occur during operation. The test runs until halted.

5 BUTTERFLY SEEK TEST FIXED DISK - steps head of fixed disk drive using butterfly pattern, then seeks to cylinder 00 and verifies that it is there. This test is useful in detecting head positioning problems in the winchester disk drive. The test runs until halted.

6 SEQUENTIAL WRITE/READ FIXED DISK - sequentially writes then reads the entire winchester disk checking for data or header errors. This test exercises the read/write circuitry of the controller and winchester disk drive and is useful in diagnosing problems in this area. The test runs until halted.

7 FIXED DISK WRITE ENABLE - permits sequential write operations on the winchester disk. (For test 6.)

7.4.9 Reconfiguration of Floppy Drive/Backup Mode

If the type of floppy drive must be altered, the drive-type flags must be updated in the bad track map. To modify drive-type flags, proceed as follows:

1. Set MODE and CLASS to 77. Press EXECUTE pushbutton.
2. System will ask for confirmation by displaying 47 in seven segment displays.
3. Set MODE and CLASS to 22 for confirmation and press EXECUTE pushbutton.
4. Enter:

MODE	CLASS	DRIVE TYPE
0	0	No floppy
0	1	SA800
0	2	SA850
0	3	Normal backup/restore
0	4	8841-02 restore

and press EXECUTE pushbutton.

NOTE

8841-02 Restore (Mode 0, Class 4) is provided for a ONE TIME disk restore of backups made with the 8841-02 controller. When this restore is completed return system to normal backup/restore (Mode 0, Class 3) and make new set of disks.

7.5 DSD 880 Error Code Interpretation

This section details the error codes reported by the DSD 880 controller, their possible causes, and troubleshooting tips. Note that the error code displayed by the seven segment LED displays is the same as the octal error code reported by the RX02 read error code command with the trailing zero deleted. There is no provision for reporting winchester numeric prior codes to the host processor.

Errors are indicated by displaying the error code in the seven segment displays, and illuminating the composite and appropriate drive fault indicators. Note that some errors are applicable to the winchester drive, some to the floppy drive, some to either drive, and some non-drive related.

When operating in normal mode, the occurrence of any error will cause the current operation to terminate and the error to be reported. When an error occurs during a HyperDiagnostic routine, it is checked to determine if it is a data or header CRC error (14 or 20). If it isn't, the current operation will terminate and the error will be reported. If the error was a CRC error, it is logged in a totalizing counter and the operation is retried. When the total number of CRC errors encountered since the start of the HyperDiagnostic reaches 16 (decimal), the HyperDiagnostic will terminate.

Unless otherwise indicated all errors apply to either drive.

ERROR CODE = XX (X = blank seven segment display)

NON DRIVE RELATED

Fault: Controller failed to complete hardware initialize

Possible cause: Defective +5 volt power supply
Defective front panel display
Interface is forcing controller to initialize continuously
Interface cable may be plugged in backwards

Troubleshooting:	Observe +5 volts OK indicator Measure +5 volt power supply at front panel test point Run switch and light HyperDiagnostic Remove interface cable, check orientation

ERROR CODE = 00 (000 octal)

Fault:	None, this is the normal operating condition

ERROR CODE = 01 (010 octal)

Fault:	Drive failed to home on initialize
Possible cause:	WINCHESTER: Winchester head retainer not removed during installation FLOPPY: Incorrect installation of SA800/SA850 jumper on controller EITHER: No drive in system Incorrect drive select jumpering Defective +24 volt power supply Defective drive
Troubleshooting:	WINCHESTER: Remove winchester drive head retainer FLOPPY: Check installation of SA800/SA850 jumper on controller board EITHER: Check head movement during initialize. If head does not move, the drive select may be incorrectly jumpered. Measure +24 volt power supply at front panel test point

ERROR CODE = 02 (020 octal)

Fault:	Nonexistent drive selected.
Possible cause:	Software attempted to access nonexistent drive
Troubleshooting:	Verify software operation

ERROR CODE = 03 (030 octal)

Fault:	Track 00 found while stepping inwards (toward hub) during initialize.
Possible cause:	Drive head may have been out beyond track zero before initialize Incorrect drive select jumpering Incorrect installation of drive cable Defective drive

Troubleshooting: Retry initialize operation
Check drive select jumpering
Check installation of drive cable

ERROR CODE = 04 (040 octal)

Fault: Invalid cylinder address

Possible cause: Software attempting to access nonexistent cylinder

Troubleshooting: Verify software

ERROR CODE = 05 (050 octal)

Fault: Track 00 found while stepping

Possible cause: Defective drive

Troubleshooting: Service drive

ERROR CODE = 06 (not reported to host processor)

Fault: WINCHESTER: SA1004 seek did not complete when expected

Possible cause: Defective SA1004

Troubleshooting: Service drive

ERROR CODE = 07 (070 octal)

Fault: Requested sector not found in two revolutions

Possible cause: Desired sector header has a hard CRC error
Disk headers incorrectly formatted
Software requested nonexistent sector address

Troubleshooting: Check disk headers for validity and reformat if necessary
Verify applications software operation

ERROR CODE = 10 (100 octal)

Fault: Write protect violation (attempted to write on write protected disk)

Possible cause: WINCHESTER: Winchester disk write protected via front panel switch
Winchester disk not stabilized (two minutes from power up)
Winchester disk write/read HyperDiagnostics not write enabled

FLOPPY: Floppy disk write enable tab missing or not opaque
Floppy disk write protected via front panel switch
Defective drive

Troubleshooting: WINCHESTER: Write enable winchester disk from front panel
Wait two minutes until winchester disk stabilizes (drive ready - stops flashing)
Write enable winchester disk write/read HyperDiagnostics

FLOPPY: Install or replace floppy disk write enable tab
Write enable floppy disk from front panel
Service drive

EITHER: Check operation of front panel write protect switches via switch and light HyperDiagnostic

ERROR CODE = 12 (120 octal)

Fault: Unable to find preamble of disk header (could not identify preamble independently of PLL).

Possible cause: WINCHESTER: SA1004 data cable reversed

FLOPPY: Floppy disk head not loaded
Incorrect installation of head load jumper

EITHER: Incorrect installation of -5 volt jumper on affected drive
Defective -12 volt power supply
Defective media

Troubleshooting: WINCHESTER: Check SA1004 data cable

FLOPPY: Check floppy disk head load
Check floppy disk load jumper

EITHER: Check installation of -5 volt jumper on affected drive
Measure -12 volt power supply at front panel test point
Reformat disk media

ERROR CODE = 13 (130 octal)

Fault: Preamble found, but no disk ID address mark within window (preamble continues forever)

Possible cause: Defective media
Media not fully formatted

Troubleshooting: Perform SET TRACK to include track 0 and reformat disk media

ERROR CODE = 14 (140 octal)

Fault:	CRC error on what appeard to be a header (found preamble)
Possible cause:	Floppy disk head load defective Incorrect headed CRC Defective media
Troubleshooting:	Check floppy disk head load Reformat disk headers Run sequential write/read HyperDiagnostics to verify disk media

ERROR CODE = 15 (150 octal)

Fault:	Address in header did not match expected track (CRC code of ID sector field was correct; track or head specified in ID field did not match expected value)
Possible cause:	FLOPPY: Incorrect installation of SA850/SA800 jumper on controller board EITHER: Defective drive Incorrect disk headers
Troubleshooting:	FLOPPY: Check installation of SA850/SA800 jumper on controller board EITHER: Check disk headers and reformat if necessary Check head positioning by running butterfly HyperDiagnostics

ERROR CODE = 16 (160 octal)

Fault:	Too many tries to find good ID address mark (found preamble)
Possible cause:	Phase-locked-loop defective Defective drive
Troubleshooting:	Check read channel signal on good track or diskette Check operation of PLL by running PLL HyperDiagnostic Service drive

ERROR CODE = 17 (170 octal)

Fault:	Data address mark not found in allotted time (correct sector ID and valid data premble found, but no data address mark followed)
Possible cause:	Incorrectly formatted media Defective media

Troubleshooting: Check read operation on good track or diskette
Reformat disk media if necessary

ERROR CODE = 20 (200 octal)

Fault: CRC error on data field

Possible cause: Defective media
Encountering excessive radiated or conducted electrical interference

Troubleshooting: Examine media for excessive wear
Attempt to reread affected data
Replace drive

ERROR CODE = 21 (210 octal)

Fault: WINCHESTER: Write gate error

Possible cause: SA1004 sensed write current in head without write gate active

Troubleshooting: Replace SA1004 disk drive

ERROR CODE = 22 (not reported to host processor)

Fault: WINCHESTER: VCO failed during read operation

Possible cause: Defective PLL circuit on controller (8840)

Troubleshooting: Check operation of PLL by running PLL HyperDiagnostic
Replace controller

ERROR CODE = 23 (230 octal)

Fault: Invalid word count specified

Possible cause: Software specified a word count inconsistant with sector size (64 words for single-density, 128 words for double-density)

Troubleshooting: Verify software

ERROR CODE = 24 (240 octal)

Fault:	FLOPPY: Media density did not match density of read or read status command.
Possible cause:	Incorrect disk density specified Disk incorrectly formatted with mixed densities
Troubleshooting:	Correct specified density Reformat disk to desired density

ERROR CODE = 25 (250 octal)

Fault:	WINCHESTER: Invalid key word specified during seek, get status, or format command FLOPPY: Invalid key word specified for set media density or format command
Possible cause:	Software specified invalid key word for command (111 octal for set media density, 154 or 155 octal for format)
Troubleshooting:	Verify software

ERROR CODE = 26 (260 octal)

Fault:	FLOPPY: Indeterminate floppy media density (controller was unable to determine the density of the media)
Possible cause:	Incorrectly formatted diskette (may be IBM 2D) Defective drive
Troubleshooting:	Check disk density in a known good drive. Reformat if necessary Service drive

ERROR CODE = 27 (270 octal)

Fault:	Write format failure
Possible cause:	Index did not appear in allotted time during write format
Troubleshooting:	Check drive spindle pulley for correct size Replace drive

ERROR CODE = 30 (300 octal)

Fault:	Data compare error (data CRC was valid but disk data did not match sector buffer data) Backup floppy data does not match winchester data read or written
Possible cause:	Defective controller
Troubleshooting:	Check sector buffer by running RAM test HyperDiagnostic Check read/write channels and media by running write/read HyperDiagnostic

ERROR CODE = 31 (310 octal)

Fault:	WINCHESTER: Invalid bad track map detected during initialize (able to read data, but data was not a valid bad track map)
Possible cause:	Bad track map overwritten Invalid backup/restore mode
Troubleshooting:	Use DSD supplied support software to rewrite bad track map

ERROR CODE = 32 (320 octal)

Fault:	WINCHESTER: Checksum of bad track map did not match stored value.
Possible cause:	Defective controller
Troubleshooting:	Reinitialize SA1000 drive Replace controller

ERROR CODE = 35 (350 octal)

NON DRIVE RELATED

Fault:	Nonexistent memory error occurred during DMA
Possible cause:	Programming error (starting address and word count was inconsistant with available memory) Defective DSD 880 interface board Defective host processor memory
Troubleshooting:	Verify software Use DSD supplied support software to test host processor memory and DSD 880 interface board

ERROR CODE = 36 (360 octal)

Fault:	Drive not ready
Possible cause:	WINCHESTER: Winchester spindle lock not removed Unable to initialize SA1004 FLOPPY: No floppy disk in drive Floppy door open Floppy drive not up to speed following automatic power down Side one of single-sided floppy disk selected by software EITHER: Drive not within speed tolerance (incorrect drive spindle pulley) Incorrect drive select jumpering Defective drive ready or index signals
Troubleshooting:	WINCHESTER: Remove winchester spindle lock Restore SA1004 bad track map FLOPPY: Check installation of media, close floppy drive door Verify software selection of floppy side Check operation of automatic power down solid state relay EITHER: Check drive spindle pulley size Check drive cables Replace drive

ERROR CODE = 37 (370 octal)

Fault:	Low ac (primary) power caused abort of write operation
Possible cause:	Temporary loss of primary power caused controller to abort the specified write operation
Troubleshooting:	Retry write operation Check if primary power is within specifications

ERROR CODE = 40 (not reported to host processor)

NON DRIVE RELATED

Fault:	Invalid disk was used for reload
Possible cause:	Invalid disk identifier was detected on a disk used for reload
Troubleshooting:	Use correct reload disk

ERROR CODE = 41 (not reported to host processor)

NON DRIVE RELATED

Fault: Multiple backup disk versions detected during reload

Possible cause: Version number of disk used for reload did not match the version number of the first valid disk
Track 0 of winchester disk is corrupted

Troubleshooting: Use correct reload disk
Perform SET TRACK to include track 0 and reformat disk

ERROR CODE = 42 (not reported to host processor)

NON DRIVE RELATED

Fault: Invalid class selected

Possible cause: Nonexistent HyperDiagnostic test selected

Troubleshooting: Reposition class switch to correct postion
Check operation of class and mode switches by running the switch and indicator HyperDiagnostic

ERROR CODE = 43 (not reported to host processor)

Fault: WINCHESTER: Invalid winchester disk address (header not found)

Possible cause: Invalid winchester sector address specified
Requested cylinder address was different from the current cylinder at which the head was positioned (implied seek)

Troubleshooting: Verify software operation
If implied seeks are desired, extended mode must be selected

ERROR CODE = 44 (not reported to host processor)

Fault: WINCHESTER: Winchester disk word count overflow

Possible cause: Multiple sector read or write operation caused SA1004 cylinder address to overflow (greater than 256 cylinders)

Troubleshooting: Verify software operation
Limit maximum RX02 cylinder to 383 decimal

ERROR CODE = 45 (not reported to host processor)

NON DRIVE RELATED

Fault:	Deleted data mark was encountered on reload floppy
Possible cause:	Reload routine encountered a deleted data sector on backup floppy
Troubleshooting:	None required Note that one or more sectors on the winchester disk following the backup may have invalid data

ERROR CODE = 46 (460 Octal)

Fault:	Invalid backup mode
Possible cause:	System incorrectly configured
Troubleshooting:	Reconfigure backup/restore mode

ERROR CODE = 47 (not reported to host processor)

Fault:	Confirmation of intent to reconfigure

ERROR CODE = 51 (not reported to host processor)

NON DRIVE RELATED

Fault:	RAM failed hardware test HyperDiagnostic
Possible cause:	Defective controller
Troubleshooting:	Replace controller

ERROR CODE = 52 (not reported to host processor)

NON DRIVE RELATED

Fault:	CRC logic failed hardware test HyperDiagnostic
Possible cause:	Malfunctioning controller
Troubleshooting:	Replace controller

ERROR CODE = 53 (not reported to host processor)

NON DRIVE RELATED

Fault:	PLL failed hardware test HyperDiagnostic
Possible cause:	Defective controller
Troubleshooting:	Replace controller

ERROR CODE = XX (XXX = undefined error code)

NON DRIVE RELATED

Fault:	Defective front panel interface
Possible cause:	Defective front panel interface logic Defective front panel logic Defective front panel cable
Troubleshooting:	Check operation of front panel by running switch and indicator HyperDiagnostic Check operation of SERDES by running ALU test Hyper-Diagnostic Replace controller PCB assembly

7.6 Subsystem Replacement

After it has been determined that a hardware malfunction exists and the problem has been isolated to a subsystem, repair can be accomplished by replacement of the faulty subsystem. All subsystems can be replaced without the use of special tools.

Repairs to the individual subsystems should only be attempted by qualified maintenance technicians on a bench setup, or at the factory.

7.7 Maintenance Assistance

Data Systems Design maintains a fully staffed Customer Service Department. If at any time during inspection, installation, or operation you encounter a problem, contact one of the offices listed below. Our trained staff can help you diagnose the cause of a failure, and if necessary, speed replacement parts to you. Any time you need to return a product to the factory, please contact Customer Service to obtain a Material Return Authorization Number.

NOTE

> If a floppy disk drive is to be shipped, a cardboard shipping disk should be inserted into the drive prior to shipment. This prevents head damage during shipment. If the winchester drive is being shipped, install the head and spindle locks to prevent damage.

Data Systems Design
Customer Service

WESTERN REGION
718 Sycamore
San Jose, CA 95035
(408) 946-5815

CENTRAL REGION
5050 Quorum Drive
Suite 339
Dallas, TX 75240
(214) 980-4884

EASTERN REGION
51 Morgan Drive
Norwood, MA 02062
(617) 769-7620
TWX: 710-336-0120

CORPORATE HEADQUARTERS
2241 Lundy Avenue
San Jose, CA 95131

For products sold outside the United States, contact your local DSD distributor for parts and customer service assistance.

APPENDIX A

DISKETTE DIRECTORIES

DSD 880A DIAGNOSTIC DISKETTE DIRECTORY

DSD#060050 REV D RELEASE DATE : 21-SEP-82

filename	DSD part number	blocks	date
DSDMON.SYS	DSD#650101 REV C (3A)	4.	18-SEP-81
DYDSD .MAC	DSD#651213 REV B	31.	22-APR-81
DYDSD .SYS	DSD#651214 REV B	3.	22-APR-81
PATCH .SAV	DSD#650105 REV A	10.	1-FEB-80
PATSET.COM	DSD#651202 REV A	1.	30-DEC-80
PATSJ .COM	DSD#651203 REV A	8.	30-DEC-80
PATFB .COM	DSD#651204 REV A	9.	30-DEC-80
PAT1 .TXT	DSD#651207 REV A	3.	30-DEC-80
PAT2 .TXT	DSD#651208 REV A	1.	30-DEC-80
PAT3 .TXT	DSD#651209 REV A	1.	30-DEC-80
PATSTR.COM	DSD#651205 REV A	1.	30-DEC-80
PATERR.TXT	DSD#651206 REV A	1.	30-DEC-80
DYV4DS.DIF	DSD#651211 REV B	3.	15-MAR-82
DYV4DS.DOC	DSD#651210 REV A	5.	30-DEC-80
RSX11M.DOC	DSD#651212 REV A	13.	21-SEP-81
DLRSX .CMD	DSD#651264 REV A	4.	18-MAR-81
DLSYSV.CMD	DSD#651215 REV B	3.	18-MAR-81
DYRSX .CMD	DSD#651217 REV B	4.	18-MAR-81
DYSYSV.CMD	DSD#651216 REV B	3.	18-MAR-81
HELP .TXT	DSD#651293 REV B	2.	10-DEC-81
RSXBF .DOC	DSD#651285 REV A	10.	13-AUG-81
RSXBF .TSK	DSD#650117 REV A (1A)	100.	12-AUG-81
RTBF .DOC	DSD#651286 REV A	10.	13-AUG-81
RTBF .SAV	DSD#650118 REV A (1A)	27.	11-AUG-81
DLWAIT.MAC	DSD#651287 REV A	6.	7-APR-81
DLWAIT.SAV	DSD#650119 REV A (1A)	2.	7-APR-81
DSDFMT.MAC	DSD#651239 REV A	7.	30-DEC-80
DSDFMT.SAV	DSD#650106 REV A	3.	30-DEC-80
RSXFMT.DOC	DSD#651288 REV A	3.	21-APR-81
RSXFMT.MAC	DSD#651289 REV A	18.	21-APR-81
RSXFMT.TSK	DSD#650120 REV A (1A)	6.	22-APR-81
DYCOM .TSK	DSD#650121 REV A	3.	20-APR-81
DYCOM .STB	DSD#651290 REV A	1.	20-APR-81
DLV488.DIF	DSD#651221 REV A	1.	10-NOV-80
DLV488.DOC	DSD#651222 REV B	3.	30-APR-81
DLV388.DOC	DSD#651238 REV B	3.	30-APR-81
BOT88T.MAC	DSD#651294 REV C	31.	06-AUG-82
88XFLP.DOC	DSD#651265 REV B	4.	26-JAN-82
88XFLP.COM	DSD#651220 REV B	3.	26-JAN-82
88TFL1.TXT	DSD#651228 REV A	1.	10-NOV-80
88TFL2.TXT	DSD#651229 REV A	1.	10-NOV-80
88TFL3.TXT	DSD#651230 REV A	1.	10-NOV-80
88TFL4.TXT	DSD#651231 REV A	1.	10-NOV-80
88TFL5.TXT	DSD#651232 REV A	1.	10-NOV-80
88TFL6.TXT	DSD#651233 REV A	1.	10-NOV-80
88TFL7.TXT	DSD#651234 REV A	1.	10-NOV-80
88TFL8.TXT	DSD#651235 REV A	1.	10-NOV-80
88TFL9.TXT	DSD#651236 REV A	1.	10-NOV-80
88TFLA.TXT	DSD#651226 REV A	1.	10-NOV-80
88TFLB.TXT	DSD#651227 REV A	1.	10-NOV-80
FLPX88.COM	DSD#651219 REV B	2.	26-JAN-82
FLT881.TXT	DSD#651225 REV A	1.	10-NOV-80
FLPX88.TXT	DSD#651223 REV A	1.	10-NOV-80
FLPY88.TXT	DSD#651224 REV A	1.	10-NOV-80
FLBT88.COM	DSD#651218 REV A	1.	10-NOV-80
DYV4DS.RSX	DSD#651390 REV A	7.	06-AUG-82
DYV40 .DOC	DSD#651391 REV A	5.	06-AUG-82
DYDRVQ.MAC	DSD#651394 REV A	84.	15-SEP-82
TRNSFM.DOC	DSD#651395 REV A	2.	15-SEP-82
TRNSFM.SAV	DSD#651393 REV A (V1A)	50.	15-SEP-82
SCNBAD.SAV	DSD#651392 REV A (V1A)	68.	15-SEP-82
FLPEXR.SAV	DSD#650102 REV C (4A)	49.	15-SEP-82
RLEXR .SAV	DSD#650124 REV C (5)	60.	15-SEP-82
WINEXR.SAV	DSD#650116 REV C (7A)	66.	15-SEP-82
RELEAS.DOC	DSD#651291 REV E	8.	16-SEP-82
INDEX .DOC	DSD#651292 REV E	8.	16-SEP-82

199. FREE BLOCKS

TOTAL OF 775. BLOCKS IN 66. FILES

```
DIRECTORY DY0:[1,54]
30-APR-81 15:33

RSX11M.SYS;1          258.     C   07-JAN-81 21:23
RSX11M.TSK;1          130.     C   07-JAN-81 21:24
RSX11M.STB;4          11.          07-JAN-81 21:24
DYDRV.STB;6           1.           07-JAN-81 21:24
DYDRV.TSK;6           5.       C   07-JAN-81 21:24
DLDRV.STB;4           1.           07-JAN-81 21:24
DLDRV.TSK;4           4.       C   07-JAN-81 21:24
LDR.TSK;3             5.       C   07-JAN-81 21:24
TTDRV.STB;4           5.           07-JAN-81 21:24
TTDRV.TSK;4           18.      C   07-JAN-81 21:24
LPDRV.STB;5           1.           07-JAN-81 21:24
LPDRV.TSK;5           4.       C   07-JAN-81 21:24
DRDRV.STB;5           1.           07-JAN-81 21:24
DRDRV.TSK;5           5.       C   07-JAN-81 21:24
FCPMD1.TSK;3          62.      C   07-JAN-81 21:24
COT.TSK;4             24.      C   07-JAN-81 21:25
LOA.TSK;4             29.      C   07-JAN-81 21:25
MCRMU.TSK;3           28.      C   07-JAN-81 21:25
SAV.TSK;4             65.      C   07-JAN-81 21:25
SHF.TSK;3             12.      C   07-JAN-81 21:25
ACS.TSK;4             15.      C   07-JAN-81 21:25
BOO.TSK;4             22.      C   07-JAN-81 21:25
IND.TSK;4             101.     C   07-JAN-81 21:25
DMO.TSK;4             13.      C   07-JAN-81 21:26
ERF.TSK;3             4.       C   07-JAN-81 21:26
ERL.TSK;3             30.      C   07-JAN-81 21:26
INI.TSK;4             34.      C   07-JAN-81 21:26
INS.TSK;5             27.      C   07-JAN-81 21:26
MOU.TSK;4             24.      C   07-JAN-81 21:26
SYS.TSK;3             78.      C   07-JAN-81 21:26
TKN.TSK;4             16.      C   07-JAN-81 21:26
UFD.TSK;4             7.       C   07-JAN-81 21:26
UNL.TSK;4             23.      C   07-JAN-81 21:26
HEL.TSK;3             33.      C   07-JAN-81 21:27
BYE.TSK;3             6.       C   07-JAN-81 21:27
ACNT.TSK;4            57.      C   07-JAN-81 21:27
PIP.TSK;2             69.      C   07-JAN-81 21:27
TEC.TSK;1             63.      C   07-JAN-81 21:27
BAD.TSK;2             50.      C   07-JAN-81 21:27
VMR.TSK;2             142.     C   07-JAN-81 21:27
MAC.TSK;1             81.      C   07-JAN-81 21:28
DMP.TSK;2             57.      C   07-JAN-81 21:28
BRO.TSK;3             25.      C   07-JAN-81 21:28

TOTAL OF 1646./1646. BLOCKS IN 43. FILES


DIRECTORY DY0:[1,2]
30-APR-81 15:34

DLSYSVMR.CMD;2        3.           02-FEB-81 02:19
DYRSXDSK.CMD;4        4.           02-FEB-81 02:26
DLRSXDSK.CMD;2        4.           02-FEB-81 02:27
DYSYSVMR.CMD;7        3.           09-JAN-81 00:08
STARTUP.CMD;11        1.           02-FEB-81 00:59

TOTAL OF 15./23. BLOCKS IN 5. FILES


DIRECTORY DY0:[5,1]
30-APR-81 15:34


TOTAL OF 0./0. BLOCKS IN 0. FILES

GRAND TOTAL OF 1661./1669. BLOCKS IN 48. FILES IN 3. DIRECTORIES
```

APPENDIX B

COMMAND FILE LISTINGS

```
! COMMAND AND DOCUMENTATION FILE TO UPDATE THE DISTRIBUTION RT11-V4 HANDLER
!   DYV4DS.DOC  30-DEC-80        880 VERSION

! THIS FILE BOTH DOCUMENTS THE PROCEDURE AND CONTAINS THE COMMANDS
! REQUIRED TO MODIFY THE DEC RT11-V4 RX02 HANDLER TO SUPPORT DOUBLE
! SIDED OPERATION.
!
! SETUP FOR DUAL FLOPPY SYSTEM
! ----- --- ---- ------ ------
!
! FIRST MAKE A COPY OF THE RX02 BOOTABLE DISTRIBUTION DISKETTE.
! THEN BOOT THIS DISK IN DY0: (LEFT HAND DRIVE)
! THEN COPY THE FILES (DYV4DS.DOC AND DYV4DS.DIF) FROM THE DSD DIAGNOSTIC DISK
! TO THE BOOTED RT-11 V4 DISKETTE IN DY0:.
!
! NOTE: THERE SHOULD BE AT LEAST 40. CONTIGUOUS FREE BLOCKS ON THIS DISK.
!       AND IT MUST CONTAIN DY.MAC, MACRO.SAV, LINK.SAV, SYSMAC.SML AND DUP.SAV
!
! SETUP FOR SINGLE FLOPPY SYSTEM (DSD880)
! ----- --- ------ ------ ------ --------
!
! 1)    COPY THE BOOTABLE RT-11 DISTRIBUTION DISKETTE ONTO THE WINCHESTER DRIVE
!           INSERT THE BOOTABLE RT-11 DISTRIBUTION DISK INTO DY0: AND BOOT IT.
!               INIT DL0:
!               COPY /SYS DY0:*.* DL0:
!               COPY /BOOT DL0:RT11SJ DL0:
!               BOOT DL0:
! 2)    COPY DY.MAC FROM THE DRIVER SOURCE DEC DISTRIBUTION DISKETTE TO DL0:
!               COPY DY0:DY.MAC DL0:
!
! 3)    COPY THE DYV4 FILES FROM THE DSD DIAGNOSTIC DISKETTE TO DL0:
!               COPY DY0:DYV4*.* DL0:
! COMMON UPDATE PROCEDURE FOR ALL HARDWARE CONFIGURATIONS.
! ------ ------ --------- --- --- -------- ---------------

!
!
! THE USER SHOULD THEN TYPE THE QUOTED COMMAND TO THE MONITOR PROMPT.
! ."@DYV4DS.DOC<CR>"
!
! UPDATE THE DY.MAC SOURCE FILE USING SLP (SOURCE LANGUAGE PATCHER)
R SLP
DYV4DS.MAC,=DY.MAC,DYV4DS.DIF

!THIS PRODUCES A REVISED HANDLER SOURCE THAT WILL NOW BE ASSEMBLED
!
R MACRO
DYV4DS,=DYV4DS

!
! SAVE THE DEC STANDARD HANDLER BY RENAMING IT.
!
RENAME /SYS/NOPROTECT DY.SYS DY.SYS
RENAME /SYS DY.SYS DY.DEC
!
! GENERATE THE NEW DY.SYS HANDLER FILE
!
R LINK
DY.SYS=DYV4DS

!
! THE NEW HANDLER SHOULD BE BOUND TO A MONITOR ON THE FLOPPY USING COPY/BOOT
!       INSERT A BOOTABLE RT-11 V4 FLOPPY INTO DY0: FOR HANDLER UPDATE

COPY /SYS DY.SYS DY0:DY.SYS
COPY/BOOT DY:RT11SJ.SYS DY:
! OR FOR THE FOREGROUND/BACKGROUND MONITOR
! COPY/BOOT DY0:RT11FB.SYS DY:
BOOT DY:
```

```
! DOCUMENTATION FILE TO UPDATE THE DISTRIBUTION RT11-V3B RL01/RL02 HANDLER
!   DLV388.DOC  30-APR-81

! THIS FILE BOTH DOCUMENTS THE PROCEDURE AND CONTAINS THE COMMANDS
! REQUIRED TO MODIFY THE DEC RT11-V3 RL01 HANDLER TO SUPPORT THE DSD880
! OPERATING IN EXTENDED MODE.
! FIRST MAKE A COPY OF THE RL01 BOOTABLE DISTRIBUTION DISK
! THEN BOOT THIS DISK IN DL0:
!
!       THE RTV3 DISTRIBUTION DL.SYS CAN ALTERNATIVELY BE MODIFIED BY
!       PATCHING DL.SYS AS FOLLOWS
!       R PATCH
!       *DL.SYS/A
!       1124/ 35600<CR>  WAS 47742
!       1466/ 35600<CR>  WAS 47742
!       2136/ 35600<CR>  WAS 47742

!       THE RT-11V3B DISTRIBUTION CAN BE PATCHED SIMILARLY
!       R PATCH
!       *DL.SYS/A
!       0050/ 35600<CR>  WAS 47742
!
!       DLMNSJ.SYS/A
!       44160/ 35600<CR>  WAS 47742
!
!       *DLMNFB.SYS/A
!       54630/ 35600<CR>  WAS 47742
!
!       NOTE: THESE LOCATIONS HOLD FOR ALL RT-11 V3B DISTRIBUTIONS

! SUPPORT FOR DSD-880 IN SYSGENED MONITORS.
! ------- --- ------- -- --------- --------
! NOTE - THE DEFINITION OF DLDSI2 SHOULD BE CHANGED IN DL.MAC BEFORE SYSGEN
! TO    DLDSI2  = <382.*20*2>-20-DLNBAD
! WAS   DLDSI2  = <512.*20*2>-20-DLNBAD  IN STANDARD DISTRIBUTION

!       THIS REFLECTS THE DIFFERENT NUMBER OF AVAILABLE RL CYLINDERS
```

```
! COMMAND AND DOCUMENTATION FILE TO UPDATE THE DISTRIBUTION RT11-V4 HANDLER
!   DLV488.DOC  30-APR-81

! THIS FILE BOTH DOCUMENTS THE PROCEDURE AND CONTAINS THE COMMANDS
! REQUIRED TO MODIFY THE DEC RT11-V4 RL01 HANDLER TO SUPPORT THE DSD880
! OPERATING IN EXTENDED MODE.
! FIRST MAKE A COPY OF THE RL01 BOOTABLE DISTRIBUTION DISK
! THEN BOOT THIS DISK IN DL0:
! THEN COPY THE FILES (DLV488.DOC AND DLV488.DIF) FROM THE DSD DIAGNOSTIC DISK
! TO THE BOOTED RT-11 V4 IN DL0: AND FOLLOW THE PROCEDURE BELOW.
!
!       THE RT-11 V4 DISTRIBUTION DL.SYS CAN ALTERNATIVELY BE MODIFIED BY
!       PATCHING DL.SYS AS FOLLOWS
!       R PATCH
!       *DL.SYS/A
!       1124/ 35600<CR>  WAS 47742
!       1466/ 35600<CR>  WAS 47742
!       2136/ 35600<CR>  WAS 47742


! NOTE: THERE SHOULD BE AT LEAST 40. CONTIGUOUS FREE BLOCKS ON THIS DISK.
!       AND IT MUST CONTAIN DL.MAC, MACRO.SAV, LINK.SAV, SYSMAC.SML AND DUP.SAV
!
!
! THE USER SHOULD THEN TYPE THE QUOTED COMMAND TO THE MONITOR PROMPT.
! ."@DLV488.DOC<CR>"
!
! UPDATE THE DL.MAC SOURCE FILE USING SLP (SOURCE LANGUAGE PATCHER)
R SLP
DLV488.MAC,=DL.MAC,DLV488.DIF

!THIS PRODUCES A REVISED HANDLER SOURCE THAT WILL NOW BE ASSEMBLED
!
R MACRO
DLV488,=DLV488

!
! SAVE THE DEC STANDARD HANDLER BY RENAMING IT.
!
RENAME /SYS/NOPROTECT DL.SYS DL.SYS
RENAME /SYS DL.SYS DL.DEC
!
! GENERATE THE NEW DL.SYS HANDLER FILE
!
R LINK
DL.SYS=DLV488

!
! THE NEW HANDLER SHOULD NOW BE BOUND TO A MONITOR USING COPY /BOOT
!
COPY/BOOT DL:RT11SJ.SYS DL:
BOOT DL:
```

; 88XFLP.DOC - DOCUMENTATION FOR RT-11 V4 BACKUP COMMAND FILES
88XFLP IS A COMMAND FILE THAT ALLOWS BACKING UP A DSD-880 WINCHESTER
ONTO MULTIPLE DOUBLE DENSITY DOUBLE SIDED DISKETTES.

BACKUP IS DONE BY COPYING SUCCESSIVE CHUNKS OF THE WINCHESTER (1951 BLKS) ONTO NAMED FILES (88BAK1.IMG ... 88BAK9.IMG) ON SUCCESSIVE DOUBLE DENSITY DOUBLE SIDED DISKETTES. NO ASSUMPTIONS ARE MADE ABOUT FILE ORGANIZATION SO AN RSX11 OR DLDP+ TYPE DISK MAY BE BACKED UP.

THE BACKUP PROCESS IS STARTED BY EXECUTING THE COMMAND FILE 88XFLP.COM. TYPE @88XFLP<CR> AFTER COPYING THE BACKUP FILE SET ONTO AN RT-11V4 DISKETTE WITH DOUBLE SIDED FLOPPY SUPPORT. (SEE DYV4DS.DOC)

A RESTORE IS DONE BY EXECUTING FLPX88.COM WHICH ASKS FOR THE SECOND DISKETTE FIRST UP THROUGH THE LAST DISKETTE. THE FIRST DISKETTE IS LOADED LAST.

CAUTION: IF A NON RT DISK IS TO BE BACKED UP THEN RT-11 MUST BE RUN FROM A SYSTEM DEVICE OTHER THAN THE RL01 OR DY0 AND THE HANDLER MUST BE PATCHED IN ORDER TO NOT DO BAD BLOCK REMAPPING AS DIRECTED BY THE BLOCK 1 ERROR MAP (FIRST 12 LOCATIONS). OTHER OPERATING SYSTEMS MAY NOT DO THE SAME STYLE OF BAD BLOCK HANDELING.

```
        THUS PATCH LOCATION 2500 WAS 177777 TO 0
                            2502 WAS 177777 TO 0 IN DL.SYS DISTRIBUTION.
```

```
! FLPX88.COM FLOPPY TO 880 WINCHESTER MULTI DISK NON-FILE STRUCTURED RESTORE

TYPE FLPX88.TXT
INIT WIN:
!COPY/DEV/NOQ SY: DL0:
!SQ DL0:/NOQ
!COPY /BOOT DL0:RT11SJ.SYS DL0:
!COPY FLPY88.COM DL0:STARTS.COM
CREATE WIN:/START:1000./ALLOCATE:949.
!BOOT WIN:

TYPE 88TFL2.TXT
COPY/WAIT FLP:88BAK2.IMG WIN:

TYPE 88TFL3.TXT
COPY/WAIT FLP:88BAK3.IMG WIN:

TYPE 88TFL4.TXT
COPY/WAIT FLP:88BAK4.IMG WIN:

TYPE 88TFL5.TXT
COPY/WAIT FLP:88BAK5.IMG WIN:

TYPE 88TFL6.TXT
SET ERROR NONE
COPY/WAIT FLP:88BAK6.IMG WIN:

!TYPE 88TFL7.TXT
!COPY/WAIT FLP:88BAK7.IMG WIN:

!TYPE 88TFL8.TXT
!COPY/WAIT FLP:88BAK8.IMG WIN:

TYPE FLPY88.TXT
COPY/WAIT FLP:88BAK1.IMG WIN:
```

```
!88XFLP.COM  COMMAND FILE TO BACK UP 880 WINCHESTER WITHOUT REGARD TO FILES
! CAN BE USED FOR RSX-11 BACKUP IF NO BAD BLOCKS ARE TO BE MAPPED AND IF
! THE DL HANDLER IS PATCHED TO IGNORE BLOCK 1 BAD BLOCK MAPPING.

! ASSUMES USE OF DOUBLE SIDED DOUBLE DENSITY DISKETTES

ASS DY0: FLP:
ASS DL0: WIN:

TYPE DY1:88TFL1.TXT
INIT FLP:
COPY/DEV/FILES DL0:/START:0/END:1950. FLP:88BAK1.IMG

TYPE 88TFL2.TXT
INIT FLP:
COPY/DEV/FILES DL0:/START:1950./END:3900. FLP:88BAK2.IMG

TYPE 88TFL3.TXT
INIT FLP:
COPY/DEV/FILES DL0:/START:3900./END:5850. FLP:88BAK3.IMG

TYPE 88TFL4.TXT
INIT FLP:
COPY/DEV/FILES DL0:/START:5850./END:7800. FLP:88BAK4.IMG

TYPE 88TFL5.TXT
INIT FLP:
COPY/DEV/FILES DL0:/START:7800./END:9750. FLP:88BAK5.IMG

TYPE 88TFL6.TXT
INIT FLP:
COPY/DEV/FILES DL0:/START:9750./END:11700. FLP:88BAK6.IMG

TYPE 88TFL6.TXT
INIT FLP:
COPY/DEV/FILES DL0:/START:11700./END:13650. FLP:88BAK7.IMG

TYPE 88TFL6.TXT
INIT FLP:
COPY/DEV/FILES DL0:/START:13650./END:15600. FLP:88BAK8.IMG
```

```
! DLRSX.CMD - COMMAND FILE TO INITIALIZE AN 880-WINCHESTER WITH RSX-11 TASKS
!  GENERATES A BOOTABLE RSX11M SYSTEM ON WINCHESTER AFTER FINAL VMR PHASE
! 16-MAR-81   -     SETS UP READY FOR VMR SYSGEN PHASE

BAD DL0:
ALL DL0:
INI DL0:DYRSXSYS
MOU DL0:DYRSXSYS
UFD DL0:[1,54]
UFD DL0:[1,2]
SET /UIC=[1,54]
PIP DL0:RSX11M.SYS/CO/BL:494.=RSX11M.TSK
PIP DL0:RSX11M.TSK/CO=RSX11M.TSK
PIP DL0:=RSX11M.STB
PIP DL0:=DYDRV.*
PIP DL0:=DLDRV.*
PIP DL0:=LDR.*
PIP DL0:=TTDRV.*
PIP DL0:=LPDRV.*
PIP DL0:=DRDRV.*
PIP DL0:=FCPMDl.TSK
PIP DL0:=COT.TSK
PIP DL0:=LOA.TSK
PIP DL0:=MCRMU.TSK
PIP DL0:=SAV.TSK
PIP DL0:=SHF.TSK
PIP DL0:=ACS.TSK
PIP DL0:=BOO.TSK
PIP DL0:=IND.TSK
PIP DL0:=DMO.TSK
PIP DL0:=ERF.TSK
PIP DL0:=ERL.TSK
PIP DL0:=INI.TSK
PIP DL0:=INS.TSK
PIP DL0:=MOU.TSK
PIP DL0:=SYS.TSK
PIP DL0:=TKN.TSK
PIP DL0:=UFD.TSK
PIP DL0:=UNL.TSK
PIP DL0:=HEL.TSK
PIP DL0:=BYE.TSK
PIP DL0:=ACNT.TSK
!
PIP DL0:=PIP.TSK
PIP DL0:=TEC.TSK
PIP DL0:=BAD.TSK
PIP DL0:=VMR.TSK
!
SET /UIC=[1,2]
PIP DL0:=STARTUP.CMD
!
!
! SETUP TO TRANSFER COMMAND FILES
!
SET /UIC=[5,1]
UFD DL0:[5,1]
PIP DL0:=DLRSX.CMD
PIP DL0:=DYRSX.CMD
PIP DL0:=DLSYSVMR.CMD
PIP DL0:=DYSYSVMR.CMD
!
! SETUP TO TRANSFER UTILITIES
! NOTE:  ADDITIONAL UTILITIES AND LIBRARIES MAY BE DESIRED
SET /UIC=[1,54]
PIP DL0:=MAC.TSK
PIP DL0:=DMP.TSK
PIP DL0:=BRO.TSK
!PIP DL0:=TKB.TSK
!PIP DL0:=CRF.TSK
!
! TRANSFER SYSLIB.OLB
!
!UFD DL0:[1,1]
!SET /UIC=[1,1]
!PIP DL0:=SYSLIB.OLB
!
!
! SECTION TO SET UP FOR FINAL VMR PHASE
! TYPE "VMR @[5,1]DYSYSVMR.CMD<CR>"
!
SET /UIC=[1,54]
INS SY0:VMR
ASN DL0:=LB0:
ASN DL0:=SY0:
ALL LB0:
```

```
! DYRSX.CMD  -  COMMAND FILE TO INITIALIZE A DISKETTE WITH RSX-11 TASKS
! FOR TRANSFER OVER TO DSD-880 WINCHESTER.
! REQUIRES A DOUBLE SIDED DOUBLE DENSITY DISKETTE
!  GENERATES A BOOTABLE RSX11M DISKETTE AFTER FINAL VMR PHASE
! 16-MAR-81   -     SETS UP READY FOR VMR SYSGEN PHASE

ALL DY0:
INI DY0:DYRSXSYS
MOU DY0:DYRSXSYS
UFD DY0:[1,54]
UFD DY0:[1,2]
SET /UIC=[1,54]
PIP DY0:RSX11M.SYS/CO/BL:258.=RSX11M.TSK
PIP DY0:RSX11M.TSK/CO=RSX11M.TSK
PIP DY0:=RSX11M.STB
PIP DY0:=DYDRV.*
PIP DY0:=DLDRV.*
PIP DY0:=LDR.*
PIP DY0:=TTDRV.*
PIP DY0:=LPDRV.*
PIP DY0:=DRDRV.*
PIP DY0:=FCPMD1.TSK
PIP DY0:=COT.TSK
PIP DY0:=LOA.TSK
PIP DY0:=MCRMU.TSK
PIP DY0:=SAV.TSK
PIP DY0:=SHF.TSK
PIP DY0:=ACS.TSK
PIP DY0:=BOO.TSK
PIP DY0:=IND.TSK
PIP DY0:=DMO.TSK
PIP DY0:=ERF.TSK
PIP DY0:=ERL.TSK
PIP DY0:=INI.TSK
PIP DY0:=INS.TSK
PIP DY0:=MOU.TSK
PIP DY0:=SYS.TSK
PIP DY0:=TKN.TSK
PIP DY0:=UFD.TSK
PIP DY0:=UNL.TSK
PIP DY0:=HEL.TSK
PIP DY0:=BYE.TSK
PIP DY0:=ACNT.TSK
!
PIP DY0:=PIP.TSK
PIP DY0:=TEC.TSK
PIP DY0:=BAD.TSK
PIP DY0:=VMR.TSK
!
SET /UIC=[1,2]
PIP DY0:=STARTUP.CMD
!
!
SET /UIC=[5,1]
UFD DY0:[5,1]
PIP DY0:=DYRSX.CMD
PIP DY0:=DLRSX.CMD
PIP DY0:=DYSYSVMR.CMD
PIP DY0:=DLSYSVMR.CMD
!
SET /UIC=[1,54]
PIP DY0:=MAC.TSK
PIP DY0:=DMP.TSK
PIP DY0:=BRO.TSK
! ADDITIONAL UTILITIES MAY BE COPIED HERE
!PIP DY0:=MAC.TSK
!PIP DY0:=TKB.TSK
!PIP DY0:=CRF.TSK
!
!UFD [1,1]
!PIP DY0:[1,1]=[1,1]SYSLIB.OLB
!
! SECTION TO SET UP FOR FINAL VMR PHASE
! TYPE "VMR @[5,1]DYSYSVMR.CMD<CR>"
!
INS SY0:VMR
ASN DY0:=LB0:
ASN DY0:=SY0:
ALL LB0:
```

```
! DYSYSVMR.CMD - VMR A RSX11M SYS ON FLOPPY 8-JUN-80 - PART 2
! INDIRECT COMMAND STREAM TO VMR
SET /POOL=1000
SET /MAIN=LDRPAR:*:24:TASK
INS LDR
FIX ...LDR
SET /MAIN=TTPAR:*:200:TASK
LOA TT:
SET /MAIN=SYSPAR:*:100:TASK
SET /MAIN=FCPPAR:*:240:TASK
SET /MAIN=GEN:*:*:SYS
LOA DY:
LOA DL:
INS FCPMD1        ! INSTALL FILE SYSTEM
INS COT           ! INSTALL CO DRIVER TASK
INS ACS           ! INSTALL ALLOCATE CHECKPOINT FILE
INS BOO           ! INSTALL BOOT
INS DMO           ! INSTALL DISMOUNT
INS ERF           ! INSTALL ERROR OFF
INS ERL           ! INSTALL ERROR LOGGER
INS IND           ! INSTALL INDIRECT FILE PROCESSOR
INS INI           ! INSTALL INITVOL
INS INS           ! INSTALL INSTALL
! INS PMD/PAR=GEN         ! INSTALL POST-MORTEM DUMPER
INS LOA           ! INSTALL LOAD
INS MCRMU         ! INSTALL MULTI-USER MCR
INS HEL           ! INSTALL LOGIN PROCESSOR
INS BYE           ! INSTALL LOGOUT PROCESSOR
INS MOU           ! INSTALL MOUNT
INS SAV           ! INSTALL SAVE
INS SHF           ! INSTALL SHUFFLER
INS SYS           ! INSTALL SYSTEM DISPLAY PART OF MCR
INS TKN           ! INSTALL TASK TERMINATION TASK
INS UFD           ! INSTALL USER FILE DIRECTORY BUILDER
INS UNL           ! INSTALL UNLOAD
SET /UIC=[1,54]:TT0:
;
SET /POOL
;
PAR
;
TAS
;
DEV
```

```
! DLSYSVMR.CMD - VMR A RSX11M SYS ON RL01 13-FEB-81
! INDIRECT COMMAND STREAM TO VMR
SET /POOL=1000
SET /MAIN=LDRPAR:*:24:TASK
INS LDR
FIX ...LDR
SET /MAIN=TTPAR:*:200:TASK
LOA TT:
SET /MAIN=SYSPAR:*:100:TASK
SET /MAIN=FCPPAR:*:240:TASK
SET /MAIN=GEN:*:*:SYS
LOA DL:
LOA DY:
LOA DR:
INS FCPMD1       ! INSTALL FILE SYSTEM
INS COT          ! INSTALL CO DRIVER TASK
INS ACS          ! INSTALL ALLOCATE CHECKPOINT FILE
INS BOO          ! INSTALL BOOT
INS DMO          ! INSTALL DISMOUNT
INS ERF          ! INSTALL ERROR OFF
INS ERL          ! INSTALL ERROR LOGGER
INS IND          ! INSTALL INDIRECT FILE PROCESSOR
INS INI          ! INSTALL INITVOL
INS INS          ! INSTALL INSTALL
INS PMD/PAR=GEN  ! INSTALL POST-MORTEM DUMPER
INS LOA          ! INSTALL LOAD
INS MCRMU        ! INSTALL MULTI-USER MCR
INS HEL          ! INSTALL LOGIN PROCESSOR
INS BYE          ! INSTALL LOGOUT PROCESSOR
INS MOU          ! INSTALL MOUNT
INS SAV          ! INSTALL SAVE
INS SHF          ! INSTALL SHUFFLER
INS SYS          ! INSTALL SYSTEM DISPLAY PART OF MCR
INS TKN          ! INSTALL TASK TERMINATION TASK
INS UFD          ! INSTALL USER FILE DIRECTORY BUILDER
INS UNL          ! INSTALL UNLOAD
SET /UIC=[1,54]:TT0:
;
SET /POOL
;
PAR
;
TAS
;
DEV
```

APPENDIX C

FLPEXR USER'S MANUAL

Appendix C: FLPEXR Users Manual

INTRODUCTION

All DSD flexible disk systems with an LSI-11 or PDP-11 interface board are shipped with a diskette containing an interactive diagnostic program called FLPEXR. The manual explains the operation of this comprehensive set of tests and utility programs. This manual assumes the user is familiar with floppy diskette operations and terminology.

FLPEXR supports the full product line of floppy disk drive products and multiple drive systems with 1 through 4 drives per system. It is a standalone program, capable of being bootstrapped into the processor. It performs auto configuration of certain control parameters, determining both disk and CPU characteristics. It supports both hard copy and video display terminals with full x-on, x-off output control. In order to facilitate unattended testing, terminal output is also retained in a circular buffer autoconfigured to the full available memory; commands are also provided to display and reset the circular buffer. Commands are also provided for diskette formatting, examination, duplication, and comparison. Test commands fully exercise system capabilities with operational parameters being user selectable through commands. The acceptance test and verify commands are suitable for both incoming quality control checks and system exercise/burn-in.

PROGRAM LOADING

FLPEXR requires a standard console device, an LSI-11 or PDP-11 computer and at least 12K words of memory. Loading FLPEXR can be accomplished by two methods. One method is to bootstrap the diagnostic diskette. This loads FLPEXR into memory automatically. The other method requires an RT-11 operating system. The FLPEXR diagnostic diskette has an RT-11 compatible directory and file space. The files on the diagnostic diskette can be accessed using standard RT-11 procedures. For example, FLPEXR can be run from an RT-11 system by typing.

RU DEV: FLPEXR <CR>

where <DEV:> might be DX0:, DX1:, DY0:, DY1: as appropriate.

> On a system running other operating systems (e.g., RSX11M, IAS, RSTS, etc.), the distribution diskette must be bootstrapped into memory.

Since both bootstrap and diagnostic programs handle RX01 and RX02 protocols, FLPEXR diagnostic diskette may be used with any DEC compatible disk system.

Once the FLPEXR diagnostic program has been loaded into memory, the diagnostic diskette may be used with any DEC compatible disk system.

Once the FLPEXR diagnostic program has been loaded into memory, the diagnostic diskette should be removed from the drive so it is not erased.

Two high quality, write-enabled formatted diskettes of the same type (density and number of sides) should be installed in the FLPEXR drives before proceeding with any of the tests.

After FLPEXR is loaded into memory, a brief description is displayed on the terminal which includes a memory map, preliminary usage instructions, and a prompt for selection of device type.

The memory map indicates the ranges of the address space which responds with SSYNC (or BRPLY) when accessed by the host computer. The figure below shows the text initially output:

```
<Memory map>
Remove distribution diskette.
DSD floppy disk diagnostic with format capa-
   bility.
Type 'V' to do verify/acceptance test on two
   drives.
   This will do a set media and short verify.
   Then go into a regular acceptance test.
Type 'H' for a list of valid commands.
Type 'FO' to format a diskette.
CTRL-C returns to mode.
CTRL-R aborts function and returns to mode.
All numeric inputs/outputs are in octal.
Insert test diskettes (both must be of same
density).
Enter device type (0 to 8) or 'CR' for list of types.
```

The device type specification is used by FLPEXR to set up internal control values that tailor the program's operation to specific DSD product capabilities. An input of Ø will select a default value that is applicable for all products. The device flag (which is the major control value set by the device type specification) can be modified during program operation by the 'SET DEVICE' command. A 'CR' input in response to the device type prompt will output the list of types as shown below:

Type	*Device*
0	Default
1	110
2	210
3	430
4	440
5	470
6	480
7	880
8	4120

Which type of device? (0 to 8):

If device 8 is selected, the following list of drive types will be shown.

Type	*Drive*
0	SA460
1	T100-4
2	MP192

After the device type is selected, FLPEXR will output the device flag being used, as shown below.

```
Device flag being used is: XXXX
Use set device command to modify flag
```

FLPEXR then outputs the name and version number of the program.

```
DSD FLPEXR VXX
```

FLPEXR types "<CRLF> #" when starting, and the program then attempts an INIT (initialize) instruction. When the INIT cycle is successful, the program types the prompt word: "DD COMMAND:" or "COMMAND:". This prompt string allows the operator to input a command. The "DD" indicates that the program is accessing double density diskettes. A list of all the available commands may be obtained by typing a 'CR'.

PROGRAM EXIT

If FLPEXR was loaded via the bootstrap, the operating system must be rebooted.

If FLPEXR was loaded via the RT-11 operating system, direct return to the operating system may be possible. A control input of 'CRTL C' will cause FLPEXR to output "EXIT TO RT-11?'. A 'Y' response will cause the return to the RT-11 monitor. Exit to the monitor may not function if:

1. There is insufficient memory available.
2. The system device is not located at 177170.
3. The system device or diskette is not available.

If the direct monitor exit is not possible, the operating system must be rebooted.

PROGRAM COMMANDS

Legal responses to "COMMAND:" are listed in Table 1, grouped by class of command. Only the characters enclosed in parenthesis need to be typed. The parenthesis should NOT be typed. When the typed string is recognized, the terminal "BELL" will sound at which time <CR> should be typed. The program will fill in the remaining characters and then proceed to execute the function.

FLPEXR also recognizes various control inputs. Table 2 lists the control input and the associated action. This input can be performed at any time, even while a test is in progress.

Table 1. FLPEXR Commands

Command	*Description*
Comprehensive Tests	
• (V)ERIFY	General Exerciser
• (SH)ORT VERIFY	Short Exerciser
Individual Tests	
• (FI)LL EMPTY	Fill/Empty Buffer Test
• (SEQW)/R	Sequential Write/Read Test
• (SEQ)READ	Sequential Read
• (RA)NDOM R/W	Random Read/Write
• (REA)D RANDOM	Read Random
• (SC)AN	Scan
• (SEE)K RANGE	Seek Range
• (SA)125	Check Head Alignment
• (CL)EAN HEAD	Clean Head Utility
Media Modification	
• (SET M)EDIA DENSITY	Set Media Density
• (FO)RMAT	Format Diskette
Program Control Values	
• (SET U)NIT	Set Unit
• (SET T)RACK	Set Track Limits
• (SEC)TOR INCREMENT	Specify Sector Inteleave
• (I)NTERRUPT	Set Interrupt Status
• (DE)NSITY LOCKUP	Lock Density to Current Density
• (SET D)EVICE	Set Device
• (H)ELP	Output List of Commands
Program Status	
• (M)AP ADDRESS	Memory and Device Map
• (ST)ATUS	Display Status Information
• (RES)ET STATUS	Change Status
• (SA)VE STATUS	Save Status on Diskette
• (DUMP C)IR BUFFER	Display Circular Output Buffer
• (REC)OVER STATUS	Retrieve
Data Utilities	
• (DUP)LICATE	Duplicate
• (CO)MPARE	Compare by Sector
• (DUMP O)CTAL	Data Dump in Octal Format
• (DUMP B)YTE	Data Dump in Byte Format
• (DUMP A)SCII	Data Dump in ASCII Format

Table 2. Control Inputs

Input	*Meaning*	*Notes*
CTRL R	Aborts current test, restarts at command	
CTRL S	Freezes terminal output until another character is typed	
CTRL O	Throws away all output until another character is typed	
CTRL P	Throws away all output except errors until another character is typed	
CTRL Q	Causes output to resume	1
<LF>	Types current track and sector and status counts	4
CTRL C	Asks 'EXIT TO RT-11?' if RT-11 monitor is available. Type Y to exit. If RT-11 monitor not available, action is similar to CTRL R. If in ODT, may return control to program	
CTRL D	Causes control transfer to ODT	2,3
CTRL T	Causes control transfer to ODT with stack trace	2,3
CTRL L	Toggles extended error printout formats	
RUB or DEL	Deletes previous character in input string	

Notes:
1. Actually, any character being input will perform this function.
2. Exit to monitor and control transfer to debug may not function if there is not enough memory available or if booted from a device other than 177170.
3. Control transfer from ODT back into FLPEXR is accomplished by 'CTRL C'. If this does not work, the program may be restarted by XXXX'G, where XXXX is the appropriate restart address (see below).
4. This command always functions; however, for some tests, the track and sector information should be disregarded (e.g., fill-empty test).

FLPEXR has several restart addresses that can be used to restart the program if necessary. They are:

1104	—Normal start-restart address
1110	—Start address from monitor call
1114	—Start at command prompt, without performing INIT on device
1100	—Return address from ODT after CTRL D dispatch

PROGRAM INPUT/OUTPUT

All data input and output is in octal format unless otherwise specified.

The 'DEL' or 'RUB' key may be used during input to remove the previously input character. On some output devices, the cursor will be backspaced one position for each 'DEL'; on other devices, a '/' will be output followed by the characters being deleted. Normal input may be resumed at any time.

The program fully supports X-on, X-off protocol (i.e., CTRL S, CTRL 0 and CTRL O) to enable output to be suspended and restarted.

Diskette data is accessed via a combined address unit #, side #, track #, and sector #. Various commands are provided to specify the limits of the address components to be used for tests. These values are set to default values when the device type is selected following initial program load.

Input is typically terminated by either a <CR> or <SP>. Validation input (e.g., Y or N) typically does not require termination.

PROGRAM STATUS AND ERROR DISPLAYS

FLPEXR types out error and status information under a wide variety of circumstances. All printouts to the console terminal are sent to a circular buffer in memory as well. The buffer size is determined by available memory. The circular buffer is useful if a hard copy console terminal is not being used and error printouts no longer on the face of the CRT screen need to be examined. The display output buffer (DUMP C) function is used to examine messages in the circular buffer. The status variables that might appear on the console terminal are explained below:

DEV XXX	Is printed only when running multiple controllers. XXX are the last 3 octal digits of the RXCS address for the system whose error/status data is being displayed.
UN U	U represents the logical drive unit number for which the error/status data is being displayed.
TRACK = TK	Track address at time of status/error printout.
SECTOR = SC	Sector address at the time of status/error printout.
RXCS = XY	Shows the contents of the command and status register.
RXDB = XY	Shows the contents of the data buffer register. It should normally be 0 or 214 octal following an INIT.
INTERRUPT ERROR: X	If X is less than 0, this indicates that an expected interrupt failed to occur. If X is greater than 0, this indicates that more than one interrupt occurred.
#BAD = XX	This variable indicates the number of status errors detected.
#RD/WRT = XX	This variable indicates the number of sectors that were transferred error-free.
#XFERS = XX	This variable indicates the number of fill/empty command cycles that were completed successfully.
B-DATA = XX	Number of data errors where a byte or word of data did not compare with the value the program was expecting. This is different than a CRC error, which would be counted as bad status. There can be up to 128 data errors in 1 sector.
DEFSTT = DEFINITIVE ERROR STATUS	Error code associated with the error currently being displayed. The meaning of each error code can be found in the unit users manual.
SIDE 1	Indicates an error has occurred on side 1 (second side of a diskette). Error messages not specifying side 1 relate to side 0. Single sided products display only side 0.

EXPANDED ERROR DISPLAYS

If in RX02 compatible mode, and CTRL L has been typed to select expanded error printout mode, the following additional status variables appear in the error printout:

D0@TK = TK	Track address of drive 0
D1@TK = TK	Track address of drive 1
CURTK = TK	Track address of the current selected logical unit
CSCT = SC	Sector address of the current selected logical unit
DSTT = XX	Drive status byte—each of the bits in this status byte is used to encode some information about one or both of the flexible disk drives and/or the media presently installed. The bits get decoded into words which are displayed with the other status. These words are explained below.
US0	Drive 0 is currently selected
US1	Drive 1 is currently selected
DN0L	Drive 0 currently contains a single density diskette
DN0H	Drive 0 currently contains a double density diskette
DN1L	Drive 1 currently contains a single density diskette
DN1H	Drive 1 currently contains a double density diskette
HDUP	Head on currently selected unit is up (unloaded)
HDLD	Head on currently selected unit is loaded

TRKRD = TK	Track address read from a sector header. This number would only be useful following a DEFSTT = 150 error.
DEF – RXDB = XX	Contents of the RXDB following a definitive error status command.

ERROR ACTIVITY CODES

A number of 2-character activity codes are displayed in the context of error printouts. The codes listed below indicate what the diagnostic was doing when the error was detected.

Activity	*Code*	*Meaning*
FILL-EMPTY	FB	Problem loading sector buffer
FILL-EMPTY	E1, E2	Sector buffer data did not check during an empty buffer operation
FILL-EMPTY	FL, EL	DMA fill or empty error to low mem. buffer
FILL-EMPTY	FD, ED	DMA fill or empty error to cir. mem. buffer
FILL-EMPTY	FH, EH	DMA fill or empty error to high mem. buffer
SEQ. WRITE	SW, CW	Problem during sequential write
SEQRD	SR	Problem during sequential read
RANDOM	RW, RC, RR	Random (write, check, read) activity when error was detected
ANY READ RETRY	XE	Empty buffer check before retrying read
DUP UTILITY	IN	Error reading the source diskette
DUP UTILITY	CW	Error checking what was just written
DELETED DATA	DW, DR	Deleted data flag failure

EXAMPLES OF ERROR OUTPUT

The following printouts are examples of what the FLPEXR diagnostic program outputs to the console under varying circumstances.

EXAMPLE 1: Operator requests status of currently selected drive during a test by typing LF.

UN 0 TRACK = 0 SECTOR = 4
BAD = 0 RD/WRT = 0 XFERS = 0
B – DATA = 0

EXAMPLE 2: Operator requests status of both drives using the "STATUS"

UN 0 BAD = 0 RD/WRT = 0
XFERS = 0 B – DATA = 0
UN 1 BAD = 0 RD/WRT = 0
XFERS = 0 B – DATA = 0

EXAMPLE 3: Disk was write protected.

Error detected on drive #1 at track #1, sector #1
error code was 100
#BAD = 1 #RD/WRT = 2002
#XFERS = 0 B – DATA = 0

EXAMPLE 4: Read on drive with no disk installed.

Error detected on drive #0 at track #1, sector #11
error code was 110
#BAD = 3 #RD/WRT = 2049
XFERS = 0 B – DATA = 0

COMPREHENSIVE TEST COMMANDS

- VERIFY—(V)ERIFY

  The VERIFY test does one pass of a SHORT ACCEPTANCE TEST, on the first 7 tracks and then resets the limit variables back to the normal default values. It then induces an automatic "CTRL P" to inhibit all but error printout and initiates the long verify test. This test will run until terminated by a "CTRL R."

EXAMPLE

```
#DD COMMAND : VERIFY
SCRATCH DISKS INSTALLED? (Y, N) : Y
SET DENSITY TO (S, D) : S
ARE YOU SURE? (Y, N) : Y
VERIFY TEST NOW STARTING
SCAN CRC CHECKED WRITING READING
INTERRUPTS ENABLED
WRITING READING
```

- SHORT VERIFY—(SH)ORT VERIFY

  This interactive program changes the track range used by the VERIFY TEST so that only the first 9 tracks of each selected drive are tested. This test will run until terminated by a CTRL R.

INDIVIDUAL TESTS

- SCAN—(SC)AN

  The SCAN test reads all sectors on all selected drives sequentially and checks for CRC errors. It also determines media density. No direct data checking takes place in this test. Only status is checked. After all units are scanned once, the "COMMAND:" prompt is displayed on the console.

EXAMPLE

```
#COMMAND: SCAN
CRC CHECKED
#COMMAND:
```

- SEEK RANGE—(SE)EK RANGE

  The SEEK RANGE function is a versatile drive test that performs all possible seeks within the operator specified track and seek length boundaries. It specifies a read on the first sector that can be read on the destination track after compensating for step and head load times. Thus it is a worst case test of the drive stepper motor and head setting. Status information will be continuously displayed during execution of this test indicating the seek length currently being used (x) and direction of seek ([^] = outward). An '!' will be output at the conclusion of each pass. This test will run continuously until terminated by a CTRL R.

EXAMPLE

```
#DD COMMAND SEEK RANGE
NOTE: ALL TIMES ARE GIVEN IN 'OCTAL'
      TENTHS OF MSEC
SEEK LENGTH ( 1 ):  3 THROUGH ( 27 )
:  7
850 SEEK TIME ( 36 ):
850 SECTOR OFFSET:   ( 4 ):
COVERING TRACKS ( 1 ):   THROUGH
            ( 114 ):  [ 3 ][ ^ ][ 4 ][ ^ ][ 5 ][ 6 ]
            [ ^ ][ 7 ][ ^ ]  !  [ 3 ][ ^ ][ 4 ][ ^ ]...
```

- FILL-EMPTY—(FI)LL EMPTY

  The FILL-EMPTY test checks the FILL BUFFER and EMPTY BUFFER controller commands. If the controller under test is configured in the RX01 compatible mode, then the test involves only programmed I/O. If the controller is configured as an RX02, the controller does FILL/EMPTIES into three different buffers so as to verify proper operation of all possible address bits. FILL/EMPTIES are done in both densities covering all possible word counts. Since this test does not manipulate the drives, the system will operate in silence. This test continues until the operator types a 'CTRL R'.

- SEQUENTIAL WRITE/READ—(SEQW)/R

  The SEQUENTIAL WRITE / READ test writes pseudo-random data sequentially on all selected drives. The test then reads all the data and checks it. The message "WRITING" is typed on the console terminal when the test first starts writing. The message "READING" is typed when the test starts reading. This test continues until the operator types "CTRL R". It also performs a set media density operation if the diskette is not of the expected density.

- SA125—(SA) 125

  The SA125 test uses an SA125 alignment disk to check head alignment. This disk is recorded with correct address marks, but with data patterns offset radially in one mil steps. This test is intended for factory use only.

- CLEAN HEAD—(CL)EAN HEAD

  The CLEAN HEAD utility allows the user to clean the read/write head using the FD-08 Disk Drive Head Cleaning Kit. Turn the line time clock (LTC) ON. Do not allow cleaning disk to remain in the system for more than 30 seconds. This test is intended for use only as directed by the factory.

Note

The following three tests require a SEQUENTIAL WRITE pass be done first in order to initialize the pseudo-random data. Data compare errors are reported if this is not done. FLPEXR prompts 'IS DISKETTE SEQUENTIAL WRITTEN? (Y, N) ' at the start of each test. A 'Y' response will initiate the test; a 'N' response will return to the command prompt.

- SEQUENTIAL READ—(SEQ) READ

  The SEQUENTIAL READ test reads the data on all selected drives sequentially and compares the data pattern against what was written. The program types "READING" at the beginning of each pass. This test continues until the operator types "CTRL R".

- RANDOM READ/WRITE—(RA)NDOM R/W

  The RANDOM READ/WRITE test selects a random sector of a selected drive, then reads or writes it. It checks data when appropriate. This test continues until the operator types "CTRL R".

- READ RANDOM—(REA)D RANDOM

  The READ RANDOM test reads randomly selected sectors. Data is checked following each read. This test continues until the operator types "CTRL R".

EXAMPLE (for 480)

```
#COMMAND: FORMAT
SEQUENTIAL SECTOR FORMAT?
(Y OR N) : Y
Density                     Type   Supported
                                   On
DEC SD (IBM SD 2-128)        0     480
                                   440
                                   210
                                   110
DEC DD                       1     480
                                   440
DEC SD (ALL OF DISK)         2     880, 480,
                                   470, 430,
                                   4140
DEC DD (ALL OF DISK)         3     880, 480,
                                   470, 430,
                                   4140
IBM SD (92-256)              4     480
IBM SD (2-512)               5     480
IBM DD (2D-256)              6     480
IBM DD (2D-512)              7     480
IBM DD (2D-1024)             8     480
DESIRED SELECTION? (0 to 8) : 4
DO YOU WISH TO DO SIDE #0? (Y OR N) : Y
DO YOU WISH TO DO SIDE #1? (Y OR N) : Y
ARE YOU SURE? (Y OR N) : Y
# COMMAND:
```

MEDIA MODIFICATION COMMANDS

- REFORMAT—(FO)RMAT

  This function is used to rewrite diskette headers, as well as all the other data on a particular diskette. It also prompts for confirmation, unit, and sequential or interleaved format. Either the entire diskette (Formats 2 through 8) or just a portion of the diskette (Format 0 through 1) may be formatted. If a partial format is selected, the track range to be formatted is specified by the set track command. The sides to be formatted can also be specified.

  FLPEXR is designed to support the full range of formats available throughout the product line. However, not all units are capable of writing all formats. If an inappropriate format is selected, an error message will be output. If the unit is not capable of IBM format modes, they will not be output in the selection menu.

  Typically, the operator should format new diskettes by Formats 2 for single density diskettes and 3 for double density diskettes.

- SET MEDIA DENSITY—(SET M)EDIA DENSITY

  This function enables the operator to initialize a diskette to single density or double density format. The function prompts for function confirmation, unit, and desired density. To select single density, respond with an "S". Type "D" to select double density.

  The SET MEDIA DENSITY command is used to implement this function, therefore, no headers are rewritten. The prompt is issued when this function is complete. This function causes any status saved on track 0, sector 1 to be erased.

  ```
  #COMMAND: SET MEDIA DENSITY
  DO A SET MEDIA ON ALL DEVICES? (Y OR N) :
  N
  UNIT: 1: SET DENSITY TO (S,D) : S
  ARE YOU SURE? (Y, N) : Y
  ```

PROGRAM CONTROL VALUE COMMANDS

- SET UNIT—(SET U)NIT

  This command enables the operator to specify which drives are to be accessed by the various test functions. The default drives are units 0 and 1. The currently selected units are printed first. It prompts with "UNIT:", expecting a number between 0 and 3, inclusive. Unit numbers are accepted as long as they are valid. When a non-number is typed to a unit request, the units currently selected are prompted and FLPEXR returns to command prompt.

Note

1. *If using a two drive system, then selection of units 2 and 3 is invalid and may cause an error.*
2. *If units are selected by "SET DEVICE", they will override "SET UNIT". See the "SET DEVICE" command for more information.*

EXAMPLE

```
"SET DEVICE" overriding "SET UNIT"
#DD COMMAND: SET UNIT
— LOADED BY SET DEVICE FLAGS
  UNITS SELECTED 1
```

- SET TRACK—(SET T)RACK

  This command enables the operator to specify lower and upper track limits for all other test functions. The default lower track limit is track 1 and upper track limit is track 76. The "COMMAND" prompt is issued after the entry of valid new limits. The lower limit must not exceed the upper limit.

EXAMPLE

```
"SET TRACK" used to set track range from
track 1 to track 10
#COMMAND: SET TRACK
FROM 1 : THROUGH 14: 10
```

- SECTOR INCREMENT—(SEC)TOR INCREMENT

  This command enables the operator to specify the sector increment value. The number is added to the present sector address to determine the next sector address in the functions that read multiple sectors on a single track. If this number were 1 and the diskette did not have an interleaved format, an entire revolution would be required to read each sector. On LSI-11 processors, the default increment value is 3. On PDP-11 processors, the default increment value is 2. The "MODE:" prompt is issued after the new value has been entered.

```
#DD COMMAND: SECTOR INCREMENT
=3-2
#DD COMMAND: SECTOR INCREMENT
=2-3
```

- SET INTERRUPT STATUS—(I)NTERRUPT

  The SET INTERRUPT STATUS command enables the operator to test the disk system with interrupts either enabled or disabled. If interrupts are enabled, the FLPEXR ensures that an interrupt occurs whenever it is appropriate. The operator enters a D to disable interrupts and an E to enable interrupts. This function is also used in ACCEPTANCE and VERIFY to set "Interrupts Enabled" and "Interrupts Disabled".

EXAMPLE

```
#DD COMMAND: INTERRUPT
CURRENTLY INTERRUPTS ARE DISABLED
(D)
INPUT NEW STATUS (ENABLE OR
DISABLE)
(E OR D) : D
```

- DENSITY LOCKUP—(DE)NSITY LOCKUP

  The "DENSITY LOCKUP" function allows the operator to lock the current disk density during the various tests. This feature is useful when testing for a problem that occurs in one density only, or when the disk density cannot be changed by a SET MEDIA DENSITY function.

EXAMPLE

```
#DD COMMAND: DENSITY LOCKUP
DENSITY IS CURRENTLY UNLOCKED
DO YOU WISH TO LOCK THE DENSITY (Y
OR N): Y
#DD COMMAND:
```

- SET DEVICE—(SET D)EVICE

  This function facilitates testing controllers that are not configured at the standard device I/O address and interrupt vector. It also enables the FLPEXR test program to simultaneously exercise multiple controllers. The function protocol asks you for device address, interrupt vector, and flag word. If a space is typed, the program steps past that field, leaving it intact. Return to "COMMAND:" is by input of a "CR" (carriage return) in response to "RXCS:". The flag word is organized as follows:

15	14	13	12	11	10	09	08
			D4120	DMA	D85	DBS	DDN

07	06	05	04	03	02	01	00
	US3	US2	US1	US0			

When set to a 1, the bit labeled:

D4120 indicates the 4120 device is set.
DMA indicates the device should be tested as an RX02.
D85 indicates 850 timing should be used (else 800 timing).
DBS indicates that double sided operation is enabled.
DDN indicates double density operation is enabled.
US3 indicates this device contains a drive unit 3.

US2 indicates this device contains a drive unit 2.
US1 indicates this device contains a drive unit 1.
US0 indicates this device contains a drive unit 0.

US0, US1, US2, US3 do an implicit "SET UNIT" function when set. The normal flag variable for RX02 mode is 4400 (octal). The normal flag variable for RX01 is 0000 (octal). The normal flag for double sided RX02 operation is 7400 (octal).

EXAMPLE SET DEVICE

```
#COMMAND:  SET DEVICE
SET THE DEVICE FLAGS FOR EACH
SYSTEM AS FOLLOWS:
  10000: SETS FLAG FOR 4120 DEVICE
   4000: ENABLES DMA OPERATION IF
         AVAILABLE
   2000: SETS 850 (FAST) SEEKING
         TIMING (ELSE 800, SLOW)
   1000: ENABLES DOUBLE-SIDED
         OPERATION IF DOUBLE-SIDED
         DRIVE AND DISK USED
    400: ENABLE DENSITY SWITCHING
         IF RX02/440/480
     20: ENABLE UNIT #1 ON CURRENT
         DEVICE
     10: ENABLE UNIT #0 ON CURRENT
         DEVICE
RXCS @ 177170: INT @ 264 INTVEC = 264
FLAGS: 4400 6410
RXCS @ 0:
```

- HELP

  The HELP command causes all the valid "MODE:" responses to be displayed on the console terminal. The "MODE:" prompt is typed when this function is complete.

PROGRAM STATUS COMMANDS

- MAP ADDRESS—(M)AP ADDRESS

  The MAP ADDRESS command causes a memory and device address map of the system to be displayed on the console terminal. This is the same map displayed when the FLPEXR program is first loaded. In addition, the interrupt vector address associated with each disk interface is displayed. The "COMMAND:" prompt is typed when this function is complete.

Note

This example indicates that a device is installed at location 177170 with interrupt vector at location 264.

EXAMPLE

```
#DD COMMAND: MAP ADDRESS
(0 – 157776)
( 160100 – 160106 )
( 165000 – 165776 )
( 171000 – 171776 )
( 172300 – 172316 )
( 172340 – 172356 )
( 172520 – 172536 )
( 173000 – 173776 )
( 176700 – 176746 )
( 177170 – 177172 )
( 177510 – 177516 )
( 177546 – 177546 )
( 177560 – 177616 )
( 177640 – 177656 )
( 177776 )
DEV: 177170 INT @ 264
```

- STATUS—(ST)ATUS

  The STATUS function causes all the current status information including hardware errors, data errors, and pass counts to be displayed on the console terminal. Displaying status information does not reset the status counts. The "COMMAND:" prompt is typed when this function is complete.

EXAMPLE

```
#COMMAND: STATUS
UNIT #0 #BAD = 3 #RD/WRT = 2049
#XFERS = 0 B – DATA = 0 ST = 110 # = 3
```

- RESET STATUS—(RES)ET STATUS

  The RESET STATUS function first displays all the available status counts. Next, the display will ask whether all of the status counts need resetting. A "Y" will cause all of the error, pass, etc. counts to be reset to zero. The "COMMAND:" prompt is output when this function is complete.

- SAVE STATUS—(SA)VE STATUS

  The SAVE STATUS command causes all the status counts associated with a particular drive to be written on track 0, sector 1 of the diskette in that drive. Only the SET MEDIA DENSITY commands over-write track 0, so the status data associated with each drive can be safely stored away. This function is used by the acceptance test so that it can survive a loss of main computer CPU memory without any loss of cumulative error data. The "COMMAND:" prompt is typed when this function is complete.

- RECOVER STATUS—(REC)OVER STATUS

  The RECOVER STATUS routine performs the opposite function performed by the SAVE STATUS function.

The status data stored away on track 0, sector 1 of the diskette in each drive is transferred back from the diskette to the status/counter variables in memory. The "COMMAND:" prompt is displayed when this function is complete.

- DISPLAY CIRCULAR OUTPUT BUFFER—(DUMP C)IR BUFFER

  The DUMP C function is used to display the output buffer associated with all console terminal output. This function is useful on systems where the console terminal is CRT. Messages previously output can be re-examined on the console. The buffer can be cleared after it is displayed by this function.

DATA UTILITIES COMMANDS

Note

The SECTOR INCREMENT function may be used to specify sector sequencing for the duplicate and compare commands. For the dump commands, a sector increment of 1 is always assumed.

- DUPLICATE—(DUP)LICATE

  The DUPLICATE command enables the operator to make a duplicate copy of a diskette. The function prompts for a source drive unit number and a destination drive unit number. For each possible sector address, the function performs a READ SOURCE SECTOR, WRITE DESTINATION SECTOR, READ DESTINATION SECTOR, and COMPARE DATA.

EXAMPLE

```
#DD COMMAND: DUPLICATE
SOURCE UNIT: 0
TO DESTINATION UNIT: 1
#DD COMMAND:
```

- COMPARE—(CO)MPARE

  The COMPARE command enables the operator to compare two diskettes starting at a specific address. The function prompts for: SOURCE UNIT, STARTING TRACK, STARTING SECTOR, NUMBER OF SECTORS, and DESTINATION UNIT. Any differences in data will be output.

- OCTAL DUMP BY SECTORS—(DUMP O)CTAL

  This command enables the operator to cause an octal dump of specified sectors to the console terminal. The function prompts for: UNIT, STARTING TRACK, STARTING SECTOR, SIDE, and NUMBER OF SECTORS.

EXAMPLE

```
#DD MODE: DUMP OCTAL
SOURCE UNIT: 0TRACK: 0 SECTOR: 1 #
SECTORS: 2
[DDEN DRIVE #0 AT TRACK 0, SECTOR 1,
SIDE 0]
SC=1
```

```
  0: 0 0 0 3776 0 0 0 0
 20: 0 0 0 0 0 0 0 0
 40: 0 0 0 0 0 0 0 0
 60: 0 0 0 0 0 0 0 0
100: 0 0 0 0 0 0 0 0
120: 0 0 0 0 0 0 0 0
140: 0 0 0 0 0 0 0 0
160: 0 0 0 0 0 0 0 0
200: 0 0 0 3722 0 0 0 0
220: 0 0 0 0 0 0 0 0
240: 0 0 0 0 0 0 0 0
260: 0 0 0 0 0 0 0 0
300: 0 0 0 0 0 0 0 0
320: 0 0 0 0 0 0 0 0
340: 0 0 0 0 0 0 0 0
360: 0 0 0 0 0 0 0 0
[DDEN DRIVE #0 AT TRACK #0, SECTOR
#2 SIDE #0]
SC=2
  0: 0 0 0 0 0 0 0 0
 20: 0 0 0 0 0 0 0 0
 40: 0 0 0 0 0 0 0 0
 60: 0 0 0 0 0 0 0 0
100: 0 0 0 0 0 0 0 0
120: 0 0 0 0 0 0 0 0
140: 0 0 0 0 0 0 0 0
160: 0 0 0 0 0 0 0 0
200: 0 0 0 0 0 0 0 0
220: 0 0 0 0 0 0 0 0
240: 0 0 0 0 0 0 0 0
260: 0 0 0 0 0 0 0 0
300: 0 0 0 0 0 0 0 0
320: 0 0 0 0 0 0 0 0
340: 0 0 0 0 0 0 0 0
360: 0 0 0 0 0 0 0 0
```

- BYTE DUMP BY SECTORS—(DUMP B)YTE

  This command enables the operator to cause an octal dump of specified sectors to the console terminal. The function prompts for: UNIT, STARTING TRACK, STARTING SECTOR, SIDE, and NUMBER OF SECTORS.

- ASCII DUMP BY SECTORS—(DUMP A)SCII

  This utility command enables the operator to cause an ASCII dump of specified sectors to the console terminal. The function prompts for: UNIT, STARTING TRACK, STARTING SECTOR, SIDE, and NUMBER OF SECTORS.

APPENDIX D

RLEXR USER'S MANUAL

RLEXR USER'S MANUAL

INTRODUCTION

All DSD systems having an LSI-11 or PDP-11 interface board are shipped with a diskette containing an interactive diagnostic program called RLEXR. This manual explains the operation of this compehensive set of tests and utility programs. The manual assumes the user is familiar with DSD 880 operations and terminology.

RLEXR is designed to test and verify all functions of the DSD 880 winchester drive subsystem in normal and extended (if applicable) mode. It runs as a stand-alone program (with bootstrap) and is capable of handling multiple drives and systems. Both display console and hard copy terminals with full X-on, X-off output control are supported. To facilitate unattended operation, all terminal output is retained in a circular text buffer that is configured to use all available memory. This buffer may be displayed or reset at any time by use of a single command. Test commands fully exercise system functions while detecting and reporting any faults or bad disk areas. The acceptance tests provide total reliability testing and are suitable for both system burn-in/exercise and quality control checks.

PROGRAM LOADING

RLEXR requires a standard console device, an LSI-11 or PDP-11 computer, and at least 16K words of memory. Loading RLEXR can be accomplished by two methods. One method is to bootstrap the diagnostic diskette. This loads RLEXR into memory automatically. The other method requires an RT-11-compatible directory and file structure. The files on the diagnostic diskette can be accessed using standard RT-11 procedures. For example, RLEXR can be run from an RT-11 system by typing:

RU <DEV:> RLEXR <CR>

where<DEV:>might be DX0:, DX1:, or DY1:, as appropriate.

On a system running other operating systems (e.g., RSX-11M, IAS, PSTS, etc.), the distribution diskette must be bootstrapped into memory. Once the diagnostic diskette has been bootstrapped into memory, the following appears on the screen:

DSD DIAGNOSTIC MONITOR VXX

DSDMON>

to run the RLEXR program, type:

RLEXR <CR>

Since both bootstrap and diagnostic programs handle RX01 and RX02 protocols, RLEXR diagnostic diskettes may be used with any DEC-compatible disk system.

Once the RLEXR diagnostic program has been loaded into memory, the diagnostic diskette should be removed from the drive so it is not erased.

One high quality, write enabled formatted floppy diskette, single- or double-density, single- or double-sided, should be installed in the drive before proceeding with any of the tests.

After RLEXR is loaded into memory, a brief description is displayed on the terminal which includes a memory map and preliminary usage instructions. The memory map indicates the ranges of the address space which responds with SSYNC or BRPLY when accessed by the host computer. The following example shows the text initially output.

After you have run RLEXR by typing:

DSDMON> R RLEXR<CR>

The following text will be printed on the screen:

```
( 000000 - 157776 )
( 171000 - 171776 )
( 172300 - 172316 )
( 172340 - 172356 )
( 172516 - 172516 )
( 173000 - 173776 )
( 174400 - 174406 )
( 177150 - 177152 )
( 177170 - 177172 )
( 177560 - 177566 )
( 177572 - 177616 )
( 177640 - 177656 )
( 177776 )
```

REMOVE THE DISTRIBUTION DISKETTE

TYPE: A TO DO AN ACCEPTANCE TEST
This will do a short acceptance test followed by a full acceptance test.

TYPE: H FOR LIST OF VALID COMMANDS

CTRL C RETURNS TO COMMAND PROMPT

CTRL R ABORTS FUNCTION AND RETURNS TO COMMAND PROMPT

ALL NUMERIC INPUTS/OUTPUTS ARE IN OCTAL

INSERT ONE TEST DISKETTE PER SYSTEM

ENTER DEVICE TYPE (0, 1, 2) OR<CR> FOR LIST: <CR>

Type	Device
0	880x/8
1	880x/20
2	880x/30

ENTER DEVICE TYPE (0, 1, 2) OR CR FOR LIST: 2

Another memory map is then printed:

```
DSD RLEXR VXX

( 000000 - 157776 )
( 171000 - 171776 )
( 172300 - 172316 )
( 172340 - 172356 )
( 172516 - 172516 )
( 173000 - 173776 )
( 174400 - 174406 )
( 177150 - 177152 )
( 177170 - 177172 )
( 177560 - 177566 )
( 177572 - 177616 )
( 177640 - 177656 )
( 177776 )
```

FULL OR PARTIAL TESTING (F, P)? P

This option is asking whether to run the diagnostic over the entire disk, or only part of the disk. Partial testing preserves tracks 00 through 10 so that testing can be performed without wiping out the diagnostic programs.

SET CLASS SWITCH TO 0
PUSH BUTTON AND TYPE A CHARACTER

This means set the switch marked CLASS on the HyperDiagnostic panel to 0 and depress the EXECUTE pushbutton. Type any character on the keyboard to signal the program to proceed.

ENABLE HALT ON ERROR (Y, N)? N

A yes means that the program will halt on the first error encountered. No means the program will store all error messages in a circular buffer. These messages can be recovered using the DUMP C command.

COMMAND:

PROGRAM EXIT

If RLEXR was loaded via RT-11 operating system or DSDMON, direct return to the monitor may be possible. A control input of CTRL C will cause RLEXR to output, EXIT TO RT-11? A yes response will cause the return to RT-11 monitor. Exit to the monitor may not function if:

1. There is insufficient memory available.
2. The system device is not located at 177170.
3. The system device is not available.

If direct monitor exit is not possible, the operating system must be rebooted.

PROGRAM COMMANDS

Legal response to

COMMAND:

The valid responses to this prompt are listed in Table 1 and grouped by class of command. Only the characters enclosed in parenthesis need to be typed. The parenthesis should NOT be typed. When the typed string is recognized, the terminal bell will sound, at which time <CR>should be typed. The program will fill in the remaining characters and then proceed to execute the function.

RLEXR also recognizes various control character inputs. Table 2 lists the control input and the associated action. This input can be performed at any time, even while a test is in progress.

Table 1. RLEXR Commands

Command	Description
Comprehensive Tests	
• (A)CCEPTANCE	General Exerciser
• (SH)ORT ACCEPTANCE	Short Exerciser
Individual Tests	
• (INTE)RFACE TEST	Interface Test
• (INTR) TEST	Interrupt Test
• (SC)AN	Scan
• (SEE)K RANGE	Seek
• (E)XTENDED MODE TEST	Extended Mode Test
• (SEQ W)/R TEST	Sequential Write/Read Test

NOTE

The following three tests require a sequential write pass.

Command	Description
• (SEQ R)EAD	Sequential Read Test
• (RANDOM R/)W	Random Read/Write Test
• (RANDOM RE)AD	Random Read Test
Program Control Utilities	
• (SET D)EVICE	Set Device
• (SET U)NIT	Set Unit
• (SET T)RACK	Set Track
• (SET I)NTERRUPT STATUS	Set Interrupt Status
• (SET M)ODE	Set Mode

Table 1. RLEXR Commands (Cont)

Command	Description
Program Status	
• (H)ELP	Provides List of Commands
• (M)AP ADDRESS	Memory and Device Map
• (ST)ATUS	Display Status Information
• (SA)VE STATUS	Save Status on Diskette
• (RES)ET STATUS	Clear Status
• (DUMP C)IR BUFFER	Display Contents of Circular Buffer
• (REC)OVER STATUS	Retrieve Status
Data Utilities	
• (RD) WITHOUT HEADER	Read Without Header
• (DUMP S)ECTOR	Display Disk Sectors

Table 2. Control Inputs

Input	Meaning	Notes
CTRL R	Aborts current test, restarts at command	
CTRL S	Freeze terminal output until another character is typed	
CTRL O	Throws away all output until another character is typed	
CTRL P	Throws away all output, except errors, until another character is typed	
CTRL Q	Causes output to resume	1
<LF>	Types current track and sector status	2
CTRL C	Asks EXIT TO RT-11?. If RT-11 monitor is available, type Y to exit. If RT-11 monitor not available, action is similar to CTRL R. If in ODT, may return control to program	3
CTRL D	Causes control transfer to ODT	3,4
CTRL T	Causes control transfer to ODT with stack trace	3,4
RUB or DEL	Deletes previous character in input string	

NOTES

1. Actually, any character being input will perform this function.

2. This command always functions; however, for some tests, the track and sector information should be disregarded (e.g., fill empty test).

3. Exit to monitor and control transfer to debug may not function if there is not enough memory available, or if booted from a device other than a 177170.

4. Control transfer from ODT back into RLEXR is accomplished by CTRL C. If this does not work, the program may be restarted by XXXX;G, where XXXX is the appropriate restart address.

Full testing will set the lower track limit to 0. Partial testing will set it to 10 (octal). Partial testing is recommended if diagnostics or other files are already on the RL. If system file RT-11 is on the RL, the lower track limit should be set much higher. The default upper track limits are:

Type	Device	Limit
0	880x/8 - normal mode	376
0	880x/8 - extended mode	576
1	880x/20	776
2	880x/30	776

Selection of the next higher tracks, (377, 577, or 777) may result in the bad block map being destroyed. The bad block map may be rewritten by using the WINEXR utility program. The set mode command may only be executed by the 880x/8 (type 0) device to change modes from normal to extended mode, or from extended mode to normal mode.

RLEXR then prints the name and version number of the program, DSD RLEXR V1A. RLEXR prints <CRLF># when starting, and then attempts an initialize sequence. When the initialize instruction is successfully completed, the program prints the prompt word, # COMMAND:. This prompt allows the operator to input a command. A list of all the available commands may be obtained by typing H (help).

RLEXR has several restart addresses that can be used to restart the program if necessary. They are:

1104 - Normal start/restart address

1110 - Start address from monitor call

1114 - Start at command prompt without performing an initialize sequence on the device

1100 - Return address from ODT after CTRL D dispatch

PROGRAM INPUT/OUTPUT

All data input and output (except status counters) are in octal format, unless otherwise specified.

The DEL or RUB key may be used during input to remove the previously input character. On some output devices, the cursor will be backspaced one position for each deletion. On others, a / will be output, followed by the characters being deleted. Normal input may be resumed at any time.

The program fully supports X-on, X-off protocol (CTRL S, CTRL O and CTRL Q) to enable output to be suspended and restarted.

Disk data are accessed via a combined address of unit, side, track, and sector values. Various commands are provided to specify the limits of the address components to be used by the tests. Default values are preset following the initial program load.

Input is typically terminated by a <CR> or <SP>. Validation input (Y, N)? typically does not require termination.

DETAILED DESCRIPTION OF STATUS AND ERROR DISPLAYS

RLEXR types out error and status information under a wide variety of circumstances. All printouts to the console terminal are sent to a circular buffer in memory as well. The buffer size is determined by available memory. The circular buffer is useful if a hard copy console terminal is not being used, and the error printouts are longer than can be displayed on the CRT screen. The display output buffer function (DUMP C) is used to examine messages in the circular buffer.

Status Variables Displayed

The status variables that might appear on the console terminal are explained below:

DEV XXX	Is printed only when running multiple controllers. XXX are the six octal digits of the CS address for the system whose error/status data is being displayed.
UN U	U represents the logical drive unit number for which the error/status data are being displayed.
TRACK= TK	Track address at time of status/error printout.
SECTOR= SC	Sector address at the time of status/error printout.
SIDE 1	Indicates the status or error relates to side one (first or second side of the disk).
RLCS= XY	Shows the contents of the command and status register.
RXCS= XY	Shows the contents of the floppy control and status register.

#BAD= XX	This variable indicates the number of status errors detected.
#RD/WRT= XX	This variable indicates the number of read and write operations performed error free.
B-TRACK= XX	This variable indicates the number of bad tracks detected.
B-DATA= XX	Number of data errors where a byte or word of data did not compare with the value the program was expecting. This is different from the CRC error, which would be counted as bad status. There can be up to 128 data errors in one sector.

Error Messages and Meanings

1 * No Bus Response *

ADDRESS

17XXXX

This indicates no SSYN acknowledge to memory access within 200 milliseconds (interface test only).

2 * Status Error *

RLCS RLBA RLDA RLMP STAT

XXXX XXXX XXXX XXXX XXXX

This indicates fault or error during operation indicated in RLCS. Parameters in address registers and status should give exact nature of error (all tests).

3 * No Interrupt *

RLCS RLBA RLDA RLMP STAT

XXXX XXXX XXXX XXXX XXXX

An expected interrupt did not occur after completion of the function in RLCS (interrupt test).

4 * Read/Write Error *

ADDRESS READ EXPECTED

17XXXX XXXX XXXX

5 * Bus Reset Error *

ADDRESS	READ	EXPECTED
17XXXX	XXXX	XXXX

A bus reset instruction did not clear all expected bits in a specific register at address indicated (interface test).

6 * Time Out Error *

RLCS
XXXX

Indicates that a function was not completed within the required time.

7 * Header CRC Error *

DEVICE	UNIT	SECTOR	SIDE	TRACK	EXPECTED	CALCULATED
17XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX

The CRC calculated by software did not compare to that written by hardware during a format operation (scan test).

8 * Non Consecutive Header Error *

DEVICE	UNIT	PREV	PRES	SIDE	TRACK
17XXXX	XXXX	XXXX	XXXX	XXXX	XXXX

Sector header information for two adjacent sectors was incorrect (scan test).

9 *Data Compare Error *

DEVICE	UNIT	SIDE	TRACK	SECTOR	EXPECTED	READ	WORD-#
17XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX

During a sequential or random read, data read did not match that expected (written). Multiple errors may indicate a bad sector or track. Refer to WINEXR utilities program for rewriting the bad track map.

10 * Bad Track Detected *

DEVICE UNIT SIDE TRACK

17XXXX XXXX XXXX XXXX

Results from multiple data compare errors on the same track.

11 * Write Protect Error *

DEVICE UNIT

17XXXX XXXX

Drive was write protected during a write operation (sequential or random write tests).

12 * Drive Select Error *

RLCS RLBA RLDA RLMP STAT

XXXX XXXX XXXX XXXX XXXX

A nonexistent drive unit was selected (all tests).

13 * Spin Error *

DEVICE UNIT RLCS

17XXXX XXXX XXXX

Indicates the drive was not up to speed during operation in RLCS (all tests).

14 * Nonexistent Memory *

DEVICE UNIT RLCS RLBA

17XXXX XXXX XXXX XXXX

15 * Seek Time Out *

DEVICE UNIT RLCS

17XXXX XXXX XXXX

A seek operation did not complete in 200 milliseconds (all tests).

16 * Write Check Error *

RLCS	RLBA	RLDA	RLMP	STAT
XXXX	XXXX	XXXX	XXXX	XXXX

Data read from disk did not compare to that originally written. Usually indicates a bad block or track (sequential read/write test).

17 * Header Not Found *

DEVICE	UNIT	RLDA
17XXXX	XXXX	XXXX

Seek to sector and track in RLDA could not be completed in 200 milliseconds due to invalid or nonexistent disk address (all tests).

18 * Header CRC Error *

DEVICE	UNIT	RLCS	RLDA
17XXXX	XXXX	XXXX	XXXX

A CRC error was detected on the header field (scan test).

19 * Data CRC Error *

DEVICE	UNIT	RLCS	RLBA	RLDA
17XXXX	XXXX	XXXX	XXXX	XXXX

A CRC error was detected during a data transfer (scan, sequential write/read, and random write/read tests).

20 * AC Power Low *

RLCS
XXXX

AC voltage is below normal, or interface cable is not connected (all tests).

Examples of Error Output

The following are examples of the RLEXR diagnostic program outputs to the console under varying circumstances:

Example 1: Operator requests status of currently selected drive during a test by typing <LF>.

```
DRIVE #0  SIDE 0  AT TRACK  155  SECTOR 0  # BAD=0
# RD/WRT=0  B-TRACK=0  B-DATA=0
```

Example 2: Operator requests status of both drives using the status command.

```
UNIT#0  #BAD=0  #RD/WRT=0  B-TRACK=0  B-DATA=0
UNIT#1  #BAD=0  #RD/WRT=0  B-TRACK=0  B-DATA=0
```

Example 3: Disk was write protected.

* Write Protect Error *

DEVICE	UNIT
174400	1

Example 4: Bad block found during read/write test.

* Data Compare Error *

DEVICE	UNIT	SIDE	TRACK	SECTOR	EXPECTED	READ	WORD
174400	1	1	207	31	14761	14561	2

DETAILED DESCRIPTION OF COMMANDS

Comprehensive Tests:

- (A)CCEPTANCE

This test does one pass of a short acceptance test on the first seven tracks and then resets the limit variables back to the default values. It then induces an automatic CTRL P to inhibit all but error printout, and initiates the longer test. This test will run until terminated by a CTRL R.

Example:

```
# COMMAND: A <CR>
SCRATCH DISKS INSTALLED? (Y,N)? Y
TEST NOW STARTING
SCAN CRC CHECKED WRITING READING
INTERRUPTS ENABLED
WRITING READING
```

- (SH)ORT ACCEPTANCE

This interactive program changes the track range used by the acceptance test so that only the first seven tracks of each selected drive are tested. This test will run until terminated by a CTRL R.

Individual Tests:

- (INTE)RFACE TEST

  Checks for response of all interface registers and issues a response error if a bus time out occurs. All read/write bits in each register are verified to be individually set and cleared without affecting other bits. A no-op or maintenance-op code is checked along with a bus reset.

- (INTR) INTERRUPT TEST

  All RL op codes (except write) are executed with interrupts enabled. If an interrupt does not occur, an interrupt error message will appear. This test runs until terminated by a CTRL R.

- (SC)AN

  The scan test reads all sectors on all selected drives sequentially, and checks for CRC errors. No direct data checking takes place in this test; only status is checked. After all units are scanned once, the command prompt is displayed on the console.

- (SEE)K RANGE

  The seek test function is a versatile drive test that performs all possible seeks within the operator specified track and seek length boundaries. Thus, it is a worst-case test of the drive stepper motor and head setting. Status information will be continuously displayed during execution of the test indicating the seek length currently being used (x) and direction of seek (∧ = outward). An ! will be printed at the conclusion of each pass. This test will run until terminated by a CTRL R.

  Example:

  ```
  # COMMAND: SEE<CR>
  SEEK LENGTH (1): 3 THROUGH (40): 7
  COVERING TRACKS (0): 10 THROUGH (776): 40
  ```

- (SEQ W)/R TEST

  The sequential write/read test writes pseudo-random data sequentially on all selected tracks. The test then reads and checks all the data. The message WRITING is typed on the console terminal when the test starts writing the data. The message READING is typed when the test starts reading the data. This test continues until the operator types CTRL R.

- (E)XTENDED MODE TEST

  Checks implied seek capability of controller during large inter-track data transfers. This test will not execute if the 880x/8 device (type 0) has been selected, and if the extended test mode was selected.

NOTE

> The following three tests require a sequential write pass be done first in order to initialize the pseudo-random data. If this is not done, data compare errors are reported.

- (SEQ R)EAD

  This test reads the data on all selected drives sequentially, and compares the data pattern against what was written. The program types READING at the beginning of each pass. This test continues until terminated by typing a CTRL R.

- (RANDOM R)/W

  This test selects a random sector of a selected drive, then reads or writes it. It checks data when appropriate. This test continues until terminated by a CTRL R.

- (RANDOM RE)AD

  This test reads randomly selected sectors. Data are checked after each read. This test continues until the operator types CTRL R.

Program Control Utilities:

- (SET M)ODE

  This test may be executed only on an 880x/8 device. The test allows selection of normal or extended mode of operation. Extended mode will allow access of tracks 0 through 576 (octal) and is selected in normal mode, class 1. Normal mode (normal switch, class 0) allows access to tracks 0 through 376 (octal). After setting class select switch to 0 or 1, depress EXECUTE pushbutton BEFORE typing a character. After typing a character, it prompts ENABLE HALT ON ERROR? If an error occurs, the error message will be printed followed by * HR *. This allows the LED to continue flashing the current error.

- (SET U)NIT

  This comand enables the operator to specify which drives are to be accessed by the various test functions. The default drive is unit 0. The currently selected units are printed first. It prompts with UNIT:, expecting a number between 0 and 3, inclusive. Unit numbers are accepted as long as they are valid. When an invalid number is typed as a response to a unit request, the units currently selected are prompted and the program returns to a command prompt.

NOTE

If using a two-drive system, selection of units 2 and 3 is invalid and may cause an error. If units are selected by a set device command, they will override the set unit command. See the set device command for more information.

- (SET T)RACK

This command allows the operator to specify lower and upper track limits for all other tests. The default lower track limit is 0. The default upper track limits are as follows:

Type	Device	Limit
0	880x/8 - normal mode	376
0	880x/8 - extended mode	576
1	880x/20	776
2	880x/30	776

If the last physical track is selected (377, 577, or 777), the bad block map might be destroyed and would have to be rewritten (refer to WINEXR User's Guide). A warning message will be output if this happens. Nothing will be destroyed until testing begins. The command prompt is issued after the entry of valid new limits. The lower limit must not exceed the upper limit.

- (SET I)NTERRUPT STATUS

This command enables the operator to test the disk system with interrupts enabled or disabled. If interrupts are enabled, the program ensures that an interrupt occurs whenever appropriate. This test is also used in the acceptance tests to set interrupts enabled or disabled. A <CR> response is a no answer.

Example:

```
# COMMAND: SET I<CR>
CURRENTLY INTERRUPTS ARE DISABLED (D)
ENABLE INTERRUPTS (Y,N)?
```

- (SET D)EVICE

This function facilitates testing controllers that are not configured at the standard device input/output address and interrupt vector. It also enables the test program to simultaneously exercise multiple controllers. The function protocol asks you for device address, interrupt vector, and flag word. If a space is typed, the program steps past that field, leaving it intact. Return to the command prompt is by input of a <CR> in response to RLCS@0:. The flag word is organized as follows:

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
									US3	US2	US1	US0			

When set to a 1, the bit labelled:

US3 indicates this device contains a drive unit 3.
US2 indicates this device contains a drive unit 2.
US1 indicates this device contains a drive unit 1.
US0 indicates this device contains a drive unit 0.

US0, US1, US2, and US3 do an implicit set unit function when set.

Example:

```
# COMMAND: SET D<CR>
SET THE DEVICE FLAGS FOR EACH SYSTEM AS FOLLOWS:
 10: ENABLE UNIT 0 ON CURRENT DEVICE
 20: ENABLE UNIT 1 ON CURRENT DEVICE
 40: ENABLE UNIT 2 ON CURRENT DEVICE
RLCS @ 174400: INT @ 160 INTVEC=160  FLAGS: 70
RXCS @ 177170
RLCS @ 0:
```

Program Status Commands:

- (H)ELP

The help command causes all valid command responses to be displayed on the console terminal. The command prompt is typed when this function is complete.

- (M)AP ADDRESS

The map address command causes a memory and device address map of the system to be displayed on the console terminal. This is the same map displayed when the RLEXR program is first loaded. In addition, the interrupt vector address associated with each disk interface is displayed. The command prompt is typed when this function is complete.

Example:

```
# COMMAND: M<CR>

( 000000 - 157776)
( 160100 - 165776)
( 171000 - 171776)
( 172300 - 172316)
( 172340 - 172356)
( 172520 - 172536)
( 173000 - 173776)
( 176700 - 176746)
( 177170 - 177172)
( 177510 - 177516)
( 177546 - 177546)
( 177560 - 177616)
( 177640 - 177656)
( 177776 )

DEV: 174400 INT @ 160
```

NOTE

The previous example indicates that a device is installed at location 177170 with interrupt vector at location 160.

- (ST)ATUS

The status command causes all the current status information, including hardware errors, data errors, and pass counts to be displayed on the console terminal. Displaying status information does not reset the status counts. The command prompt is printed when this function is complete.

Example: # COMMAND: ST<CR>
UNIT #0 #BAD=3 #RD/WRT=2049 B-DATA=0 B-TRACK=0
COMMAND:

- (RES)ET STATUS

The reset status command first displays all the available status counts. Next, the display will ask whether all the status counts need resetting. A yes response will cause all of the error, pass, etc., counts to be reset to zero. The command prompt is output when this function is complete.

- (SA)VE STATUS

This command causes all the status counts associated with a particular drive to be written on track 0, sectors 1, 2, and 3 of the diskette in that system. This function is used by the acceptance test so that it can survive a loss of main computer memory, without any loss of cumulative error data. The command prompt is displayed when this function is completed.

- (REC)OVER STATUS

This command performs the opposite function performed by the save status command. The status data stored on track 0, sectors 1, 2, and 3 of the diskette in each drive is transferred back from the diskette to the status/counter variables in memory. The command prompt is displayed when the function is complete.

- (DUMP C)IR BUFFER

This command is used to display the output buffer associated with all console terminal outputs. This function is useful on systems where the console terminal is a CRT. Messages previously output can be re-examined on the display. The buffer can be cleared after it is displayed by this command.

Data Utilities Commands:

- (DUMP S)ECTOR

This command enables the operator to cause an octal, or ASCII dump, at a specified sector to the console terminal. This function prompts for unit, cylinder, sector, side, ASCII or octal format, and exit from this function.

Example:

```
# COMMAND DUMP S<CR>
ALL PARAMETERS ARE IN OCTAL
UNIT (0,2)? 2<CR>
CYLINDER (0,776)? 23 <CR>
SECTOR (0,47)? 5 <CR>
SIDE (0,1)? 1<CR>
DUMP IN ASCII OR OCTAL WORD FORMAT (A,0)? 0<CR>
.....
EXIT (Y,N)? Y
```

- (RD) WITHOUT HEADER

This command performs the same function as the dump sector command.

APPENDIX E

WINEXR USER'S MANUAL

CAUTION

The WINEXR utility program should only be used if the system will not pass the RLEXR program, and the entry of new bad tracks, or reformatting, is indicated.

WINEXR is a utility that allows direct access to all winchester tracks. Used improperly, it can destroy the existing bad track map, rendering the RL01 or RL02 emulation inoperative.

WINEXR USER'S MANUAL

INTRODUCTION

All DSD 880 flexible disk systems with an LSI-11 or PDP-11 interface board are shipped with a diskette containing an interactive diagnostic program called WINEXR. This manual explains the operation of this set of utility programs. The manual assumes the user is familiar with DSD 880 operations and terminology.

WINEXR supports the direct access mode of the DSD 880 and bad track map generation. It is a stand-alone program, capable of being bootstrapped into the processor. It performs auto-configuration of certain control parameters, determining both disk and CPU characteristics. It supports both hard copy and video display terminals with full X-on, X-off output control. In order to facilitate unattended testing, terminal output is also retained in a circular buffer, auto-configured to the full available memory. Commands are provided to display and reset this circular buffer. Commands are also provided for disk formatting, bad track mapping, and examination. Test commands fully exercise system capabilities with operational parameters being user selectable through commands. The acceptance test, drive test, and verify commands are suitable for both incoming quality control checks, and system exercise/burn-in.

PROGRAM LOADING

WINEXR requires a standard console device, an LSI-11 or PDP-11 computer and, at least, 16K words of memory. Loading WINEXR can be accomplished by two methods. One method is to bootstrap the diagnostic diskette. This loads DSDMEN. The other method requires an RT-11 operating system. The WINEXR diagnostic diskette has an RT-11-compatible directory and file space. The files on the diagnostic diskette can be accessed using standard RT-11 procedures. For example, WINEXR can be run from the RT-11 system by typing:

RU<DEV:>WINEXR<CR>

where<DEV:>might be DX0:, DX1:, DY0:, or DY1:, as appropriate.

On a system running other operating systems (e.g., RSX-11M, IAS, RSTS, etc.), the distribution diskette must be bootstrapped into memory.

After WINEXR is loaded into memory, a brief description is displayed on the terminal which includes a memory map and preliminary usage instructions. The memory map indicates the ranges of the address space which responds with SSYNC or BRPLY when accessed by the host composter.

This device type specification is used by WINEXR to set up internal control values that tailor the program's operation to specific DSD winchester product capabilities. A CR input, in response to a device-type prompt, will output the list of types as shown below.

TYPE	DEVICE
0	880x/8
1	880x/20
2	880x/30

Which type of device? (0, 1, or 2)

After the device type is selected, WINEXR will output the device flag being used as shown below:

Device flag being used is: XXXX
Use set device command to modify flag
Is unit in mode 0, class 2 or 7 ? (Y,N):
Is bad track map on disk? (Y,N):
Skip bad tracks during testing? (Y,N):

880x/8	mode 0, class 2
880x/20	
880x/30	mode 0, class 7

WINEXR then outputs the name and version number of the program.

DSD WINEXR

WINEXR types <CRLF># when starting the program, and then attempts an initialize instruction. When the initialize cycle is successful, the program types the prompt word command. This prompt string allows the operator to input a command. A list of all the available commands may be obtained by typing an H (help).

PROGRAM EXIT

If WINEXR was loaded via the bootstrap, the operating system must be rebooted. If WINEXR was loaded via the RT-11, or DSDMON operating system, direct return to the monitor may be possible. A control input of CRTL C will cause WINEXR to output EXIT TO RT-11? A yes response will cause return to the monitor. Exit to the monitor may not function if:

1. There is insufficient memory available.

2. The system device is not located at 177170.

3. The system device, or diskette, is not available.

If direct monitor exit is not possible, the operating system must be rebooted.

PROGRAM COMMANDS

Legal responses to:

COMMAND:

The valid responses to this prompt, # COMMAND:, are listed in Table 1 and grouped by class of command. Only the characters enclosed in parenthesis need to be typed. The parenthesis should NOT be typed. When the typed string is recognized, the terminal bell will sound, at which time <CR> should be typed. The program will fill in the remaining characters and then proceed to execute the function.

Table 1. WINEXR Commands

Commands	Description
Comprehensive Tests	
• (V)ERIFY	General Exerciser
• (A)CCEPTANCE	General Exerciser
• (D)RIVE	Drive Exerciser
Individual Tests	
• (FI)LL EMPTY	Fill/Empty Buffer
• (SEQ W)/R	Sequential Write/Read
• (SEQ) READ	Sequential Read
• (RA)NDOM R/W	Random Read/Write
• (REA)D RANDOM	Read Random
• (SC)AN	Scan
• (SEE)K RANGE	Seek Range
Media Modification	
• (RE-)FORMAT RL	Reformat Disk
• (B)AD TRACK MAPPING	Entry of Bad Track Map
• (P)RINT BAD TRACK MAP	Output Bad Track Map
• (T)RANSFORM	Transform RL Address to SA Address
• (RL) BAD SECTOR	Rewrite RL Bad Sector Map
• (DISC)OVERED BAD TRACKS	Output Discovered Bad Tracks
Program Control Values	
• (SET U)NIT	Set Unit
• (SET T)RACK	Set Track Limits
• (SET S)ECTOR INCREMENT	Specify Sector Interleave
• (SET D)EVICE	Set Device
• (H)ELP	Output List of Commands
• (SET P)RINTING	Printing Control
Program Status	
• (M)AP ADDRESS	Memory and Device Map
• (ST)ATUS DISPLAY	Display Status Information
• (RES)ET STATUS	Change Status
• (SA)VE STATUS	Save Status on Diskette
• (DUMP C)IR BUFFER	Display Circular Output Buffer
• (REC)OVER STATUS	Retrieve Status
Data Utilities	
• (DUMP O)CTAL	Data Dump in Octal Format
• (DUMP B)YTE	Data Dump in Byte Format
• (DUMP A)SCII	Data Dump in ASCII Format

WINEXR also recognizes various control character inputs. Table 2 lists the control character inputs and the associated action. This input can be performed at any time, even while a test is in progress.

Table 2. Control Inputs

Input	Meaning	Notes
CTRL R	Aborts current test, restarts at command	
CTRL S	Freeze terminal output until another character is typed	
CTRL O	Throws away all output until another character is typed	
CTRL P	Throws away all output, except errors, until another character is typed	
CTRL Q	Causes output to resume	1
<LF>	Types current track and sector and status counts	2
CTRL C	Asks EXIT TO RT-11?. If RT-11 monitor is available, type Y to exit. If RT-11 monitor not available, action is similar to CTRL R. If in ODT, may return control to program	3
CTRL D	Causes control transfer to ODT	3,4
CTRL T	Causes control transfer to ODT with stack trace	3,4
RUB or DEL	Deletes previous character in input string	

NOTES

1. Actually any character being input will perform this function.

2. This command always functions; however, for some tests, the track and sector information should be disregarded (e.g., fill/empty test).

3. Exit to monitor and control transfer to debug may not function if there is not enough memory available, or if booted from a device other than a 177170.

4. Control transfer from ODT back into WINEXR is accomplished by CTRL C. If this does not work, the program may be restarted by XXXX;G, where XXXX is the appropriate restart address.

WINEXR has several restart addresses that can be used to restart the program if necessary. They are:

1104 - Normal start - Restart address

1110 - Start address from monitor call

1114 - Start at command prompt, without performing initialize on device

1100 - Return address from ODT after CTRL D dispatch

PROGRAM INPUT/OUTPUT

All data input and output are in octal format, unless otherwise specified.

The DEL or RUB key may be used during input to remove the previously input character. On some output devices, the cursor will be backspaced one position for each deletion. On other devices, a / will be output, followed by the characters being deleted. Normal input may be resumed at any time.

The program fully supports X-on, X-off protocol (CTRL S, CTRL O, and CTRL Q) to enable output to be suspended and restarted.

Disk data are accessed via a combined address of unit, head, track, and sector values. Various commands are provided to specify the limits of the address components to be used for tests. Default values are preset following the initial program load.

Input is typically terminated by either a <CR> or <SP>. Validation input (Y,N)? typically does not require termination.

DETAILED DESCRIPTION OF STATUS AND ERROR DISPLAYS

WINEXR types out error and status information under a wide variety of circumstances. All printouts to the console terminal are sent to a circular buffer in memory as well. The buffer size is determined by available memory. The circular buffer is useful if a hard copy console terminal is not being used and error printouts, no longer on the face of the CRT screen, need to be examined. The display output buffer (DUMP C) function is used to examine messages in the circular buffer. The status variables that might appear on the console terminal are explained below.

Status Variables Displayed

DEV XXX — Is printed only when running multiple controllers. XXX are the three octal digits of the RXCS address for the system whose error/status data are being displayed.

UN U — U represents the logical drive unit number for which the error/status data is being displayed.

TRACK= TK	Track address at time of status/error printout.
SECTOR= SC	Sector address at the time of status/error printout.
RXCS= XY	Shows the contents of the command and status register.
RXDB= XY	Shows the contents of the data buffer register.
INTERRUPT ERROR	If X is less than 0, this indicates that an expected interrupt failed to occur. If X is greater than 0, more than one interrupt occurred.
#BAD= XX	Indicates the number of status errors detected.
#RD/WRT= XX	Indicates the number of sectors that were transferred error free.
#XFERS = XX	Indicates the number of fill/empty command cycles that were completed successfully.
B - DATA= XX	Number of data errors where a byte, or word of data, did not compare with the value the program was expecting. This is more difficult than a CRC error, which would be counted as bad status. There can be up to 128 data errors in one sector.
DEFSTT= DEFINITIVE ERROR STATUS	Error code associated with the error currently being displayed. The meaning of each error code can be found in the user's manual.

Error Activity Codes

A number of two-character activity codes are displayed in the context of error printouts. The codes listed below indicate what the diagnostic was doing when the error was detected.

Activity	Code	Meaning
FILL/EMPTY	FB	Problem loading sector buffer
FILL/EMPTY	E1, E2	Sector buffer data did not check during an empty buffer operation
FILL/EMPTY	FL, EL	DMA fill or empty error to low memory buffer
FILL/EMPTY	FD, ED	DMA fill or empty error to center memory buffer
FILL/EMPTY	FH, EH	DMA fill or empty error to high memory buffer
SEQ WRITE	SW, CW	Problem during sequential write
SEQ	SR	Problem during sequential read

Activity	Code	Meaning
RA	RW,RC,RR	Random (write, check, or read) activity when error was detected
ANY READ RETRY	XE	Empty buffer check before retrying read

DETAILED DESCRIPTION OF COMMANDS

Comprehensive Test Commands:

- (V)ERIFY

  The verify test does one pass of a short acceptance test on the first seven tracks, then resets the limit variables back to the normal default values. It then induces an automatic CTRL P to inhibit all but error printout, and initiates the acceptance test. This test will run until terminated by a CTRL R.

  Example:
  ```
  #DD COMMAND: V<CR>
  VERIFY TEST NOW STARTING
  WRITING - PASS CODE= 0 READING - PASS CODE = 0 RANDOM
  RD/WRT
  READING - PASS CODE = 0
  PASS FINISHED
  ```

- (A)CCEPTANCE

  This interactive program changes the track range used by the verify test so that only the first nine tracks of each selected drive are tested. This test will run until terminated by a CTRL R.

- (D)RIVE

  The functions in this command are similar to the verify test except it does not do seek range functions.

Individual Tests:

- (SC)AN

  The scan test reads all sectors on all selected drives sequentially, and checks for CRC errors. No direct data checking takes place in this test; only status is checked. After all units are scanned once, the command prompt is displayed on the console.

  Example:
  ```
  # COMMAND: SC<CR>
   CRC CHECKED
  # COMMAND:
  ```

- (SEE)K RANGE

The seek range function is a versatile drive test that performs all possible seeks within the operator specified track and seek length boundaries. Thus, it is a worst-case test of the drive stepper motor and head setting. Status information will be continuously displayed during execution of this test indicating head, the seek length currently being used (x), and direction of seek (^ = outward). An ! will be output at the conclusion of each pass. This test will run until terminated by a CTRL R.

Example:
```
#DD COMMAND: SEE<CR>
ALL TIMES ARE GIVEN IN OCTAL TENTHS OF MSEC
SEEK LENGTH ( 1 ): 3 THROUGH ( 27 ): 7
850 SEEK TIME (36):
850 SECTOR OFFSET: (4):
COVERING TRACKS (0): 1 THROUGH (114); 3 (HEAD: 0)
3  4  S  6  7   !  3  4 ...
```

- (FI)LL EMPTY

The fill/empty test checks the fill buffer and empty buffer controller commands. The controller does fill/empties into three different buffers to verify proper operation of all possible address bits. Fill/empties are done to cover the drives; the system will operate in silence. This test continues until the operator types a CTRL R.

- (SEQ W)/R

The sequential write/read test writes pseudo-random data sequentially on all selected tracks. The test then reads and checks all the data. The message WRITING is typed on the console terminal when the test starts writing. The message READING is typed when the test starts reading. This test continues until the operator types CTRL R.

NOTE

The following three tests require a sequential write pass be done first to initialize the pseudo-random data. Data compare errors are reported if this is not done. WINEXR prompts IS DISKETTE SEQUENTIALLY WRITTEN? (Y,N)? at the start of each test. A yes response will initiate the test. A no response will return to the command prompt.

- (SEQ) READ

  The sequential read tests reads the data on all selected drives sequentially and compares the data pattern against what was written. The program types READING at the beginning of each pass. This test continues until the operator types CTRL R.

- (RA)NDOM R/W

  The random read/write test selects a random sector of a selected drive and reads or writes it. It checks data when appropriate. This test continues until the operator types CTRL R.

- (REA)D RANDOM

  The read random test reads randomly selected sectors. Data are checked following each read. This test continues until the operator types CTRL R.

Media Modification Commands:

- (DISC)OVERED BAD TRACKS

  This command will accumulate information for bad tracks discovered during test execution. Any discovered bad tracks should be verified by specific tests, and the bad track map updated. This data are reset each time the program is initiated.

- (RL) BAD SECTOR

  This command is used to rewrite the RL bad sector data if it has become corrupted. In normal operation, the data should not be corrupted; however, diagnostic testing may have modified the data.

  Example:
  ```
  # COMMAND: RL<CR>
  WRITE RL BAD SECTOR: (Y,N)? Y
  WRITING RL BAD SECTOR
  RL BAD SECTOR COMPLETED
  # COMMAND:
  ```

- (B)AD TRACK MAPPING

  This command enables the operator to input bad tracks, or update the bad track map. The input prompt is issued after the operator selects decimal or octal input. A CR will terminate input mode. The operator is allowed to do editing on new bad tracks. It also allows formatting of the disk before writing the bad track map and the RL bad sector on the disk.

Example:

```
# COMMAND: B<CR>
ENTRY OF NEW BAD TRACK MAP? (Y,N)? Y
ARE YOU SURE? (Y,N)? Y
DSD 880 BAD TRACK MAP
LATEST UPDATE: 26-NOV-80
DRIVE SN: 1234567890
DATE FIRST ENTERED: 26-NOV-80
DECIMAL/OCTAL INPUT? (D,O)? O
***OCTAL INPUT***
TRACK: 1       HEAD: 1
TRACK: 2       HEAD: 2
TRACK: 101     HEAD: 3
TRACK: 202     HEAD: 2
TRACK: 303     HEAD: 3
TRACK:<CR>
ANY MORE INPUT? (Y,N)? N
```

TRACK		—HEAD	TRACK		—HEAD	TRACK		—HEAD
DECIMAL	OCTAL		DECIMAL	OCTAL		DECIMAL	OCTAL	
1	1	1	2	2	1	65	101	3
130	202	2	195	303	3			

```
EDIT INPUT? (Y,N)? Y
DECIMAL/OCTAL INPUT? (D,O)? D
***DECIMAL INPUT***
ADD (Y,N)? Y
TRACK: 10      HEAD: 1
TRACK: <CR>
ANY MORE INPUT? (Y,N)? N
DELETE? (Y,N)? Y
TRACK: 1       HEAD: 1
TRACK: <CR>
EXIT EDITING? (Y,N)? Y
```

TRACK		—HEAD	TRACK		—HEAD	TRACK		—HEAD
DECIMAL	OCTAL		DECIMAL	OCTAL		DECIMAL	OCTAL	
2	2	1	65	101	3	130	202	2
195	303	3	10	12	1			

```
EDIT INPUT? (Y,N)? N
```

TRACK DECIMAL	TRACK OCTAL	—HEAD	TRACK DECIMAL	TRACK OCTAL	—HEAD	TRACK DECIMAL	TRACK OCTAL	—HEAD
2	2	1	10	12	1	65	101	3
130	202	2	195	303	3			

FORMAT DISK? (Y,N)? N
FLOPPY DRIVE IN UNIT? (Y,N)? Y
SA800 FLOPPY DRIVE (DEFAULT SA850 DRIVE)? (Y,N)? Y
WRITE BAD TRACK MAP ON DISK? (Y,N)? Y
WRITING RL BAD SECTOR
BAD TRACK MAP COMPLETED

NOTE

Press EXECUTE pushbutton on HyperDiagnostic panel after rewrite/update of bad track map to signal the controller to read the new bad track information.

- (P)RINT BAD TRACK MAP

This command prints the existing bad track map on the CRT or printer.

Example: # COMMAND: P <CR>
DSD 880 BAD TRACK MAP
LATEST UPDATE: 16-DEC-80
DRIVE SN: A10533
DATE FIRST ENTERED: 16-DEC-80
FORMAT: 2
FLOPPY DRIVE - SA850 SYSTEM TYPE - 880x/xx

TRACK DECIMAL	TRACK OCTAL	—HEAD	TRACK DECIMAL	TRACK OCTAL	—HEAD	TRACK DECIMAL	TRACK OCTAL	—HEAD
8	10	3	9	11	3	10	12	3
24	30	1	24	30	2	25	31	0
25	31	1	25	31	2	26	32	0
26	32	1	26	32	2	27	33	0
27	33	2	28	34	2			

- (RE-)FORMAT RL

This command allows the user to reformat the winchester disk. The program responds with:

2-WAY OR 3-WAY INTERLEAVE ? (2,3)?

The two-way interleave provides faster throughput in most instances. Units are shipped with two-way interleave. The DMA burst length jumper on the LSI-11 interface must be in the eight-word burst mode for two-way interleaving to work on an LSI-11. The three-way interleaving could improve performance on systems that have a lot of DMA activity, or systems with a lot of operating system overhead. The two-way interleaving is best for most DEC software systems. The program next asks:

FULL FORMAT (HEADERS AND DATA)? (Y,N)?

A yes causes the program to write the data fields as well as the header fields. This format takes longer, but writes the entire disk. A no causes the program to write headers only. This format is faster, but might cause problems for programs that attempt to read sectors which have not been written upon previously. A sequential write in RLEXR will fill in the data fields for this type of format. The program then types:

FORMAT FROM TRACK 1 THROUGH 777
ON SURFACE 0 THROUGH 7

This prompt tells the user what portions of the disk will be formatted. The set track command should be used to change these parameters. Two sections of the disk are of special interest, the hardware bad track map which resides on track 0, and the RL bad sector maps. To preserve both types of maps, reset the lower track limit to 0 using the set track command. When this parameter is 0, the program automatically saves and restores the bad track and bad sector maps if allowed to run to completion.

WARNING

> If you abort the reformat operation before its completion, the bad track and bad sector information will be lost.

The program then types:

INCREMENT IS 7 SECTORS, ENTER NEW INCR:

Increment refers to the sector offset from track-to-track.

FAST FORMAT OF RLXX

This refers to your earlier choice of no to the full format question.

- (T)RANSFORM

This command is used to map cylinder, surface, and sector of the RL01/02 winchester disk drives. The computed winchester cylinder and surface are adjusted to take bad tracks into account. The bad track map is examined for bad tracks up to and including the target track. Each bad track encountered causes the target surface and cylinder to be incremented by one surface.

Example: (If no bad track)

```
Unit: 0    RL TRACK: 12    RL HEAD: 3    RL SECTOR:  4
           DA TRACK: 10    DA HEAD: 0    DA SECTOR: 34
           RL TRACK:<CR>
```

Program Control Values Commands:

- (SET U)NIT

  This command enables the operator to specify which drives are to be accessed by the various test functions. The default drive is unit 2. The currently selected units are printed first. It prompts with UNIT:, expecting a number between 0 and 3, inclusive. Unit numbers are accepted as long as they are valid. When a non-number is typed to a unit request, the units currently selected are prompted and WINEXR returns to command prompt. Note that the single winchester 880 systems default to unit 2 and do not allow unit selection.

- (SET T)RACK

  This command enables the operator to specify lower and upper track limits for all other test functions. The default lower track limit is 1 and the upper track limit is 377. The command prompt is issued after the entry of valid new limits. The low limit must not exceed the upper limit.

  Example: Set track used to set track range from 1 to 100 on heads 1 and 2.

  ```
  # COMMAND: SET T<CR>
  FROM (0): 1 THROUGH (377): 100
  HEAD FROM (0): 1 THROUGH (3): 2
  ```

  NOTE

  880x/20 maximum track is 577, maximum head is 5.
  880x/30 maximum track is 777, maximum head is 7.

- (SET P)RINTING

  This command enables the line printer for output device. The printer device address is LpCS = 175564, LpDB = 175566.

- (SET S)ECTOR INCREMENT

  This command enables the operator to specify the sector increment value. The number is added to the present sector address to determine the next sector address in the functions that read multiple sectors on a single track. The prompt is issued after the new value has been entered.

  Example:

  ```
  # COMMAND: SET S<CR>
  INCREMENT IS 7 SECTORS
  ENTER NEW INCR: 6
  ```

- (SET D)EVICE

  This function facilitates testing controllers that are not configured at the standard device input/output address and interrupt vector. It also enables the WINEXR test program to simultaneously exercise multiple controllers. The function protocol asks you for device address, interrupt vector, and flag word. If a space is typed, the program steps past the field, leaving it intact. Return to command is by input of a<CR>in response to RXCS:.

- (H)ELP

  The help command causes all the valid command responses to be displayed on the console terminal. The command prompt is typed when this function is complete.

Program Status Commands:

- (M)AP ADDRESS

  The map address command causes a memory and device address map of the system to be displayed on the console terminal. This is the same map displayed when the WINEXR program is first loaded. In addition, the interrupt vector address associated with each disk interface is displayed. The command prompt is typed when this function is complete.

  Example: #DD COMMAND: M<CR>

```
( 000000 - 157776 )
( 160100 - 160106 )
( 165000 - 165776 )
( 171000 - 171776 )
( 172300 - 172316 )
( 172340 - 172356 )
( 172520 - 172536 )
( 173000 - 173776 )
( 176700 - 176746 )
( 177170 - 177172 )
( 177510 - 177516 )
( 177546 - 177546 )
( 177560 - 177616 )
( 177640 - 177656 )
( 177776 )

<DEV:> 177170 INT @ 264
```

NOTE

The previous example indicates that a device is installed at location 177170 with interrupt vector at location 264.

- (ST)ATUS DISPLAY

  The status function causes all the current status information including hardware errors, data errors, and pass counts to be displayed on the console terminal.

Displaying status information does not reset the status counts. The command prompt is typed when this function is complete.

Example: # COMMAND: ST <CR>
UNIT #0 #BAD=3 #RD/WRT=2049 #XFERS=0
B-DATA=0 ST = 110 # = 3

- (RES)ET STATUS

  The reset status function first displays all the available status counts. Next, the display will ask whether all of the status counts need resetting. A yes will cause all the error, pass, etc., counts to be reset to zero. The command prompt is output when this function is complete.

- (SA)VE STATUS

  The save status command causes all the status counts associated with a particular drive to be written on track 0, sector 1 of the diskette in that drive. Only the set media density has command over write track 0, so the status data associated with each drive can be safely stored away. This function is used by the acceptance test. It can survive a loss of main computer CPU memory without any loss of cummulative error data. The command prompt is typed when this function is complete.

- (REC)OVER STATUS

  The recover status routing performs the opposite function performed by the save status function. The status data stored away on track 0, sector 1 of the diskette in each drive is transferred back from the diskette to the status/counter variables in memory. The command prompt is displayed when this function is complete.

- (DUMP C)IR BUFFER

  This command is used to display the output buffer associated with all console terminal outputs. This function is useful on systems where the console terminal is a CRT. Messages previously output can be re-examined on the console. The buffer can be cleared after it is displayed by this function.

<u>Data Utilities Commands:</u>

NOTE

The set sector increment function may be used to specify sector sequencing for the duplicate and compare commands. For the dump commands, a sector increment of one is always assumed.