BOSS/IX Diagnostics and Error Log Manual

July, 1989
Document Reorder Number 006204-003

M6204C

PAGE STATUS

Section	Pages	Effective Date
Cover		July 1989
Title Page		July 1989
Page Status	STATUS-1	July 1989
Table of Contents	CONTENTS-1 thru -11	July 1989
List of Figures	CONTENTS-12	July 1989
List of Tables	CONTENTS-12	July 1989
Preface	PREFACE-1	July 1989
Section 1	1-1 thru 1-4	July 1989
Section 2	2-1 thru 2-32	July 1989
Section 3	3-1 thru 3-28	July 1989
Section 4	4-1 thru 4-12	July 1989
Section 5	5-1 thru 5-12	July 1989
Section 6	6-1 thru 6-151	July 1989
Section 7	7-1 thru 7-91	July 1989
Section 8	8-1 thru 8-76	July 1989
Section 9	9-1 thru 9-16	July 1989
Section 10	10-1 thru 10-10	July 1989
Appendix A	A-1 thru A-5	July 1989

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

Page

Appendix A - Installing Diagnostics

LIST OF FIGURES

LIST OF TABLES

PREFACE

This manual describes the diagnostic programs provided for MAI 2000/2500/3000/4000 running the MAI BOSS/IX operating system, version 7.4. This information is provided as an aid for field service personnel in locating hardware problems to the board and socketed IC level.

The major topics covered in this manual and the changes made for this revision, are:

Section 1 Introduction

Section 2 Error Logging

Section 3 BASS Diagnostics

Section 4 Diagnostic Executive

Section 5 System Interaction Test (SIT)

Section 6 Logic Tests

- Added Dual SCSI Controller Logic Test

Section 7 Function Select Tests

- Added Dual SCSI Controller Function Select Test

Section 8 Micro-Diagnostic System (MDS)

Section 9 Disk Utilities (DUTIL)

Section 10 SCSI Disk Utilities (FAVI)

- New Section

Appendix A Installing Diagnostics

- New Appendix

SECTION 1
INTRODUCTION

1.1 OVERVIEW

This manual describes the diagnostic features developed for the MAI 2000, 2500, 3000 and 4000 systems. The diagnostics described are for release 7.4A and above, though many apply to earlier releases as well. The features consist of a system error logging facility, which maintains a file of boot and runtime errors on the system, and a set of programs to aid in diagnosing system problems to various levels.

The error logging facility is included in the base system software, and so is available on all systems (unless the program files are deleted). Error logging is started automatically during the normal boot procedure. Access to the error log is available on-line, during normal system operation.

Most of the diagnostic programs are distributed on tape or floppy. The exception is the Micro-Diagnostic System (MDS), which resides in system PROMS on MAI 2500, 3000 and 4000 systems (MDS is not available on the MAI 2000).

The Basic All-purpose Service System (BASS) diagnostic is an on-line system exerciser, and is the highest level set of diagnostic programs. All other diagnostic programs are off-line diagnostics, and require a special boot procedure (alternate load) or, in the case of MDS, invocation of the PROM routine from the boot menu.

The System Interaction Test (SIT), Logic tests, Function Select tests and Disk Utility (DUTIL) all operate under the control of the diagnostic Executive program (Section 4). These programs are distributed on the bootable diagnostic media (DIA).

Instructions for installing the diagnostic Executive, SIT, Logic tests, Function Selects and BASS on the fixed disk are provided in Appendix A.

1.2 DIAGNOSTIC STRATEGY

The diagnostic programs are designed to provide an aid in trouble shooting at different functional levels. For instance, the BASS tests exercise system peripheral devices under the control of the operating system, and may be run in a multi-user environment. This is the highest level of diagnostic, and is mostly useful for detecting failure of the peripherals themselves.

At a lower level, SIT simulates operation in an operating system environment, but in a closely controlled manner. It checks for interaction problems between competing devices, and helps isolate system problems to a suspect board or boards. Logic tests, at a deeper level, further diagnose problems at the board level, followed by Function Select tests which allow a degree of diagnostic to the component level.

MDS is the lowest level diagnostic, allowing testing of the system logic.

Given this hierarchical structure of the diagnostic programs, a general diagnostic strategy would be as follows:

1. Check the error log to determine which parts of the system are failing. Errors may be logged that have gone unnoticed during system operation.

2. If a peripheral device is suspect, use the BASS tests to exercise the device, and check for errors.

3. If the problem appears to involve the controllers, use SIT and/or Logic tests to check the functioning of the suspect controller either in interaction with other controllers or alone.

4. Use the Function Select Tests to identify failing chips on controller boards.

5. Use MDS to diagnose the system when errors prevent the system from booting.

Steps 1 through 3 and 5 can be performed in the field, and can reduce the amount of hit-or-miss board swapping. Step 4, involving Function Select tests, is generally appropriate only at the repair depot level.

The DUTIL diagnostic is primarily used only for formatting fixed disks. It does have some functionality useful as a diagnostic as well. It is run similar to the function selects, but may be of use at the Logic test level (step 3).

1.3 RELATED PUBLICATIONS

The following manuals provided related information.

- BOSS/IX Technical Reference Manual, M6227
- MAI 2000 Service Manual, M8079
- MAI 2500/3000 Service Manual, M8108
- MAI 4000 Service Manual, M8205

In addition, there are several related controller service manuals, as listed in the system service manuals.

1.4 NOTATIONAL CONVENTIONS

A few notational conventions have been adopted in this manual.

1.4.1 Keys and Key Sequences

Individual keys on the keyboard are referred to by their legend. Letter keys, for example the Individual keys on the keyboard are referred to by their legend. Letter keys, for example the **K** key, is referred to by the letter in normal type. When the reference occurs in running text, it is shown in bold type, as in the previous sentence. If it is shown on a displayed line, it is not bolded, e.g.:

K

Occasionally, especially for non-letter symbols, we further enclose the key name in double quotation marks for added clarity, e.g., "/".

Some keys are either not marked, for instance the **SPACE** bar, or have different names depending on the keyboard, such as the **ENTER** or **RETURN** key. In this later case, we always refer to it as the **ENTER** key. When there are more than one possible markings for a key, we have picked one to use through out the text which we believe is unambiguous.

MAI Basic Four terminals typically have a series of four keys, marked "I", "II", "III" and "IV", sometimes called "Motor Bars" and sometimes "Control Keys." We refer to these keys as **CTL-I**, **CTL-II**, **CTL-III** and **CTL-IV**.

A sequence of keys is occasionally used by the diagnostic programs to perform special functions. These sequences consist of pressing and holding down the **CTRL** (or **CONTROL**) key while pressing some other key. These sequences are described by joining the two keys with a "plus" sign, e.g., **CTRL + C**.

1.4.2 Entered and Displayed Information

Most messages displayed by they diagnostics are shown on displayed lines in normal type, for instance:

This is a message

In some cases we use a reduced, non-proportional type, primarily for size and formatting reasons:

```
This is a message
```

Rarely, a single word message is shown in running text enclosed in quotation marks, e.g., "ERROR."

Entering information, such as commands, into the system consist of typing the information followed by pressing the **ENTER** key. Rather than say that, we usually simply say enter:

the information

We mean, type the information shown and then press **ENTER**. On infrequent occasions we say something like, "Type **Y** and press **ENTER**." They mean the same.

1.4.3 Command Format Descriptions

Several diagnostics have commands associated with them. Commands typically consist of the command name followed by one or more arguments or parameters:

command param1 param2 ...

Parameters may be shown either by constants or variables. A constant is the actual value of the parameter, and is shown in normal type. A variable parameter may take any of several values, such as numeric values, and is shown in the format in *italics*. The legal values for the parameter will be described in the text. For example:

command A *value*

The parameter A is constant, and *value* is variable.

If a parameter is optional, it is enclosed in braces, for example:

command {*value*}

If two or more parameters are alternates (at most one can be used per command), they are separated by a vertical bar (|). If selection is optional, the parameters are enclosed in braces:

command {A | B}

In this case, either A or B or neither may be used. If one of the alternatives must be used, they are enclosed in parentheses:

command (A | B)

1.5 ABBREVIATIONS

The following are some common abbreviations used in this manual. Others may be used, as explained in the text.

- BOT Beginning of Tape
- CMB Central Microprocessor Board
- CRC Cyclical Redundancy Check
- DMA Direct Memory Access
- EOT End of Tape
- IOPB I/O Parameter Block
- KB/MB Kilobyte (1024 bytes)/Megabyte (1048576 bytes)
- SCC Serial Communications Controller
- VFC Vertical Forms Control

SECTION 2
BOSS/IX ERROR LOGGING

2.1 OVERVIEW

The BOSS/IX operating system supports error logging as a standard feature. Error logging is carried out in two stages. Errors are first logged to memory. Up to eight errors can be logged in memory. In addition, an error log process periodically updates an error log file on the fixed disk. This error log process runs as a background process started automatically when the system enters multi-user mode.

A system command, **/sys/errlog**, displays the error log record and sets attributes of the error logging process.

The usual error log file is **/etc/error.log**. This file can only be displayed by using the errlog command. The **p** and **cat** commands do not provide meaningful output.

2.2 THE ERROR LOG COMMAND

A single command is provided to display the contents of the error log and to set attributes of the error logging process. The general syntax of the command is:

/sys/errlog {*parameter1 ... parameterN*}

where the parameters are selected from the following:

file	The name of the error log file, usually /etc/error.log.
-quiet	Used when starting the error monitoring and file update as a background process.
-initial	Initializes the specified error log file, clearing out all existing entries. The file is initialized to the size and starting number specified by the size= and errors= parameters.
size=	Sets the maximum number of errors recorded in the error history log. The default size is 100. The size can be set only when initializing the error log file.
errors=	Sets the current starting error number in the error history log. The default is 1. The error number can be set only when initializing the error log file.
-mem	Displays memory errors only.
-tape	Displays tape errors only.
-term	Displays terminal port errors only.
-printer	Displays printer errors only.
-comm	Displays communications errors only.
-disk	Displays disk errors only.
-text	Used only with the -disk option, this displays descriptive text for each disk error.

2.2.1 Displaying the Error Log

Use the errlog command to display the error log disk file or the memory error log. To display the disk file, you must specify the file name, usually /etc/error.log. If you do not specify the file name, only the errors logged in memory are reported.

Display the entire disk error log file using this command:

```
/sys/errlog /etc/error.log
```

To display only errors of a certain type (e.g., memory, disk, tape, and so on), use the appropriate command line option as listed in section 1.2. For example:

```
/sys/errlog /etc/error.log -tape
```

2.2.2 The Error Log Display

The error log file contains two types of error records: "By Type" and "Chronological." The By Type log is at the beginning of the error log file, followed by the Chronological log.

The By Type records contain 2 entries for each type of error. The first entry indicates the number of occurrences of the error. The second entry describes the last occurrence of this type of error. The error types are very broad, such as "disk error" or "memory parity error." For example, an entry for Winchester disk errors is shown:

```
Winchester disk          First: Apr 14 1987 09:30:43    Count:  1
 Minor:   35   Type: -2    Last: Apr 14 1987 09:30:43      Rev:
A:   8241    B: 1         C:  0          D:  0           E:  0
T:   28      U: 95992931  V:  40                    Text:
```

The first line indicates the major device by common name (Winchester disk), the number of errors (1) of this type, and the date of the first occurrence. The following lines give specific information on the latest error of this type, including the minor device number and error type. The information provided is described in later sections.

For Winchester disk errors only, the additional information can be displayed as text. This is displayed by the -disk and -text options on the errlog command:

```
/sys/errlog /etc/error.log -disk -text
```

The Chronological log maintains a chronological list of errors that have occurred since the error log was last initialized. Errors are logged up to the maximum number specified when the file was initialized (default = 100). Older errors are then replaced with new errors. The errors are displayed starting with the most recent.

The format of entries in the Chronological log similar to that of the By Type log, except that an entire log entry for each error is displayed. For example:

```
Number:  6     Number since boot:  6      Date: Thu Apr 16 1987 15:04:43
Cartridge tape  Minor:  0   Type: -62    Class:  F        Rev:  A3
      A:  39        B:  7      C:  0         D:  0          E:  16
      T:  86A0      U:  4000   V:  0                     Text:

Number:  5     Number since boot:  5      Date: Wed Apr 15 1987 14:54:06
illegal instruction         Type: -32767 Class:           Rev:
      A:  44        B:  1      C:  0         D:  0          E:  0
      T:  AC1A      U:  0      V:  0                     Text:

Number:  4     Number since boot:  4      Date: Wed Apr 15 1987 13:10:47
CMB             Minor:  1   Type: -32767 Class:           Rev:
      A:  0         B:  0      C:  0         D:  0          E:  0
      T:  10        U:  0      V:  0                     Text:

Number:  3     Number since boot:  3      Date: Wed Apr 15 1987 13:10:46
CMB             Minor:  1   Type: -2     Class:           Rev:
      A:  0         B:  0      C:  0         D:  0          E:  0
      T:  10        U:  0      V:  0                     Text:

Number:  2     Number since boot:  2      Date: Tue Apr 14 1987 09:30:43
Winchester disk Minor:  35  Type: -2     Class:            Rev:
      A:  8241      B:  1      C:  0         D:  0          E:  0
      T:  B6F4      U:  0      V:  0                     Text:

Number:  1     Number since boot:  1      Date: Tue Apr 14 1987 07:57:00
memory management           Type: -32767 Class:           Rev:
      A:  193       B:  1      C:  337       D: -1191165876  E: -64510
      T:  B5F4      U:  0      V:  0                     Text:
```

2.3 ERROR INTERPRETATION

The following subsections describe the interpretations of the error reports for each error type.

Fields A through E are decimal fields. Fileds T, U and V are hexadecimal fields. When interpreting the hexadecimal fields, bit 0 is the least significant bit (LSB). For example, fields A and U in error number 6 in the previous example have the following interpretation:

A: 39	A decimal value, indicating block 39 is in error
U: 4000	A hexadecimal value, bit 14 is ON indicating "end of data".

2.3.1 System Trap Errors

There is a large variety of errors that may be assigned to major device number 0. These are generally caught by a system trap. The common names displayed for device number 0 is determined by the minor device number, as follows:

Minor Dev Number	Name
0	bus error
1	illegal instruction
2	bpt/trace (not used)
3	iot (not used)
4	power fail
5	system call
6	trap instruction
7	parity (MAI 2000 only)
8	PIRQ (not used)
9	floating point (not used)
10	memory management
11	odd address
12	divide by 0
13	check
14	overflow
15	privilege
16	error halt
17	format
18	illegal inst. (f-line) (MAI 2500/3000/4000 only)
19	coprocessor protocol (MAI 2500/3000/4000 only)
20	PMMU configuration (MAI 2500/3000/4000 only)
21	PMMU illegal operation (MAI 2500/3000/4000 only)
22	PMMU access level (MAI 2500/3000/4000 only)
23	multiple bit memory (MAI 2500/3000/4000 only)
24	single bit memory (MAI 2500/3000/4000 only)
25	cache parity (MAI 2500/3000/4000 only)
26	calendar (MAI 2500/3000/4000 only)
27	UPS transition (MAI 2500/3000/4000 only)
28	EIA trans. parity (expansion interface)
29	EIB trans. parity (expansion interface)
30	power fail (expansion unit) (MAI 4000 only)
31	UPS transition (expansion unit) (MAI 4000 only)

The following paragraphs contain detailed descriptions of the error log field values for each of these errors. The errors are groups based on common field value interpretations.

2.3.1.1 BUS ERROR, MEMORY MANAGER AND ODD ADDRESS

Major: 0

Minor: 0 = Bus Error
10 = Memory Manager
11 = Odd Address

Class: N/A

REV: N/A

A: process id

B: process mode:
1 = user mode
0 = supervisor mode

C: MAI 2000 - Reference classification

7	interrupt acknowledge
6	supervisor program
5	supervisor data
4	not used
3	not used
2	user program
1	user data
0	not used

MAI 2500/3000/4000 - 68020 special status word

Note: This field is expressed in decimal notation, but is a bit oriented value. You must first convert the decimal value to a hex value, then interpret the bits as follow:

Bit	
15	Fault on Stage C of the instruction pipe
14	Fault on Stage B of the instruction pipe
13	Return flag for Stage C of the instruction pipe*
12	Return flag for Stage B of the instruction pipe*
11-9	0
8	Fault/Return flag for data cycle*
7	Read/Modify/Write on data cycle
6	Read/Write for Data Cycle: 1 = Read; 2 = Write
5-4	Size code for data cycle
3	0
2-0	Address space for data cycle

* 1 = Rerun faulted bus cycle, or Run pending prefetch
0 = Do not rerun bus cycle

D: MAI 2000 - Type of access

1	Read
0	Write

MAI 2500/3000/4000 - Data cycle fault address

E: MAI 2500/3000/4000 - Value of PMMU status register

Note: This field is expressed in decimal notation, but is a bit oriented value. You must first convert the decimal value to a hex value, then interpret the bits as follow:

Bit	
15	Bus error. A bus error was returned to the PMMU from physical memory during a table search.
14	Limit violation. The table index exceeded a limit field.
13	Supervisor violation. An attempt was made to access a supervisor only page.
12	Access level violation. Access fields have been exceeded. (Access level protection is not used.)
11	Write protected. An attempt was made to write to a read-only page.
10	Invalid page. A page descriptor was fetched which was marked "invalid."
9	Modified page. This bit is set when a page has been modified.
8	Not used.
7	Globally shared. Shared bit is set in the logical address descriptor.
6-4	Not used.
3-0	Level number. Number of tables used during the translation.

T: Program counter at error.

MAI 2000. The program counter value, which is the address of the instruction that was executing at the time the fault was detected, could be advanced by up to 5 words.

MAI 2500/3000/4000. The program counter value, which is the address of the instruction that was executing at the time the fault was detected, is not necessarily the instruction that caused the error to occur, due to the overlapped execution performed by the CPU.

U: MAI 2000 - Instruction being processed at the time the error occurred (see explanation for field T.)

MAI 2500/3000/4000 - Stage B instruction fault address.

V: MAI 2000 - Address being accessed when the error occurred.

MAI 2500/3000/4000 - Instruction pipe Stage C fault address

Z: Not used.

2.3.1.2 MISCELLANEOUS TRAPS

These error log field descriptions apply to the following errors:

Major: 0

Minor:

1	illegal instruction
4	power fail
5	system call
6	trap instruction
12	divide by 0
13	check
14	overflow
15	privilege
16	error halt
17	format
18	illegal inst. (f-line) (MAI 2500/3000/4000 only)
19	coprocessor protocol (MAI 2500/3000/4000 only)
20	PMMU configuration (MAI 2500/3000/4000 only)
21	PMMU illegal operation (MAI 2500/3000/4000 only)
22	PMMU access level (MAI 2500/3000/4000 only)
30	power fail (expansion unit) (MAI 4000 only)

CLASS: N/A

REV: N/A

A: process id

B: process mode:
1 = user mode
0 = supervisor mode

C: not used

D: not used

E: not used

T: Program counter at error.

MAI 2000. The program counter value, which is the address of the instruction that was executing at the time the fault was detected, could be advanced by up to 5 words.

MAI 2500/3000/4000. The program counter value, which is the address of the instruction that was executing at the time the fault was detected, is not necessarily the instruction that caused the error to occur, due to the overlapped execution performed by the CPU.

U: not used

V: not used

Z: not used

2.3.1.3 PARITY ERROR (MAI 2000 ONLY)

Major: 0

Minor: 7

Type: -32767, nonexistent memory was specified.

CLASS: N/A

REV: N/A

A: process id

B: N/A

C: N/A

D: N/A

E: N/A

T: Address indicating the 128 KB bank that failed. The following addresses (in hex) are for full 256 KB boards. If the system contains 128 KB boards, adjust the table accordingly. (Boards are determined by switch setting, not by position in the stack.)

Board	Address Range
1	00000 to 3FFFF
2	40000 to 7FFFF
3	80000 to BFFFF
4	C0000 to FFFFF
5	100000 to 13FFFF
6	140000 to 17FFFF
7	180000 to 1BFFFF
8	1C0000 to 1FFFFF

U: Parity status register

Bit	
7	0: user mode when failure occurred 1: supervisor mode when failure occurred
6	1: parity error occurred in upper byte
5	1: parity error occurred in lower byte
0	0: parity error on CMB access 1: parity error on system bus controller access

Note: Bits 5 and 6 are undefined for 2+ artwork CMB's.
If the error occurred during access by a system bus controller, bit 7 is undefined.

V: N/A

Z: N/A

2.3.1.4 SINGLE AND MULTIPLE BIT MEMORY ERRORS (MAI 2500/3000/4000 ONLY)

Major: 0

Minor: 23 (multiple bit)
24 (single bit)

Type: -32767, nonexistent memory was specified.

CLASS: N/A

REV: N/A

A: process id

B: process mode:
1 = user mode
0 = supervisor mode

C: Count of single bit errors (single bit memory only).

D: Bus cycle:
0 = DMA cycle
1 = CPU cycle

E: Number of single bit error restarts, where a restart occurs when single bit error logging is disabled.

T: Upper physical address lines at time of error. Range is 100000 to 3F00000.

U: Low order 7 bits of the memory status register.

Bit	
2	0: data cache inhibited
	1: data cache operating
1	0: no single bit error occurred
	1: a single bit error occurred ans was corrected
0	0: no multiple bit error occurred
	1: a non-correctable multiple bit error occurred

V: Syndrome byte. The syndrome bits are interpreted according to the following table. Bit 7 is always 1.

Z: N/A

Table 2-1. Syndrome Bit Decoding

MSB (bits 7-4)								
LSB (3-0)	1000	1001	1010	1011	1100	1101	1110	1111
0000	u/c	2-b	2-b	DB0	2-b	u/c	u/c	2-b
0001	2-b	u/c	u/c	2-b	u/c	2-b	2-b	DB16
0010	2-b	DB29	DB7	2-b	u/c	2-b	2-b	u/c
0011	u/c	2-b	2-b	u/c	2-b	DB13	DB23	2-b
0100	2-b	DB28	DB6	2-b	u/c	2-b	2-b	DB17
0101	u/c	2-b	2-b	DB31	2-b	DB12	DB22	2-b
0110	u/c	2-b	2-b	u/c	2-b	DB11	DB21	2-b
0111	2-b	DB27	DB5	2-b	u/c	2-b	2-b	CB3
1000	2-b	DB26	DB4	2-b	u/c	2-b	2-b	u/c
1001	u/c	2-b	2-b	u/c	2-b	DB10	DB20	2-b
1010	DB31	2-b	2-b	u/c	2-b	DB9	DB19	2-b
1011	2-b	DB25	DB3	2-b	DB15	2-b	2-b	CB2
1100	u/c	2-b	2-b	u/c	2-b	DB8	DB18	2-b
1101	2-b	DB24	DB2	2-b	u/c	2-b	2-b	CB1
1110	2-b	u/c	u/c	2-b	DB14	2-b	2-b	CB0
1111	DB30	2-b	2-b	CB6	2-b	CB5	CB4	none

u/c Uncorrectable multi-bit error.
CBx Error in Check Bit x.
DBx Error in Data Bit x.
2-b Double bit error.

2.3.1.5 DATA CACHE PARITY (MAI 2500/3000/4000 ONLY)

Major: 0

Minor: 25

Type: -32767 nonexistent memory was specified.

CLASS: N/A

REV: N/A

A: process id

B: mode
1 = user process
0 = supervisor process

C: Count of data cache errors. The count reported may be less than the actual count, since the data cache can be turned off after a specified number of errors and then turned back on.

D: N/A

E: N/A

T: Data cache parity status register.

Bit	
7	1 = Cache Parity Error in byte 0 (D00-D07) 0 = No Cache Parity Error in byte 3
6	1 = Cache Parity Error in byte 1 (D08-D15) 0 = No Cache Parity Error in byte 3
5	1 = Cache Parity Error in byte 2 (D16-D23) 0 = No Cache Parity Error in byte 3
4	1 = Cache Parity Error in byte 3 (D24-D31) 0 = No Cache Parity Error in byte 3
3	Address bit 11 at time of error.
2	0 = Cache set to 1 1 = Cache set to 0
1	0 = Parity error in tag bits A18-A11 1 = No error in tag bits A18-A11
0	0 = Parity Error in tag bits A25-A19 1 = No error in tag bits A25-A19

U: N/A

V: N/A

Z: N/A

2.3.1.6 CALENDAR (MAI 2500/3000/4000 ONLY)

Major: 0

Minor: 26

Type:
- -32 Calendar chip is not keeping correct time.
- -62 Timeout reading calendar chip.
- -132 Battery is low or dead.

CLASS: N/A

REV: N/A

A: Calendar Chip Time (in seconds since midnight, 1/1/80)

B: System Time (in seconds since midnight, 1/1/80)

C: N/A

D: N/A

E: N/A

T: N/A

U: N/A

V: N/A

Z: N/A

2.3.1.7 UPS TRANSITION (MAI 2500/3000/4000 ONLY)

Major: 0

Minor: 27 Base unit UPS
31 Expansion unit UPS

Type: N/A

CLASS: N/A

REV: N/A

A: on/off flag
1 = UPS on
0 = UPS off

B: total count of UPS transitions since boot.

C: N/A

D: N/A

E: N/A

T: N/A

U: N/A

V: N/A

Z: N/A

2.3.1.8 EXPANSION INTERFACE TRANSMISSION ERRORS (MAI 3000/4000 ONLY)

Major: 0

Minor:
28 EIA transmission parity error
29 EIB transmission parity error

Type: N/A

CLASS: N/A

REV: N/A

A: process id

B: process mode:
1 = user process
0 = supervisor process

C: N/A

D: N/A

E: N/A

T: program counter at error

U: Expansion interface interrupt 7 status register

Bit	
0	Transmission parity error has occurred in the low byte
1	Transmission parity error has occurred in the high byte
2	Error occurred during DMA cycle
3	Error occurred during read cycle
4	Error occurred during interrupt acknowledge cycle
5-7	not used

V: N/A

Z: N/A

2.3.2 1/2 Inch Streamer

Major: 2

Minor: 0

Type:

-2	I/O error
-26	Device is busy
-33	Bad system call
-62	Time out
-89	Device off-line
-98	Tape media error
-111	Parity error
-129	Failed attempt to open
-32767	Nonexistent memory specified by system call

These errors are all fatal, and so are logged. Non-fatal errors, such as read overflows, file mark read, no write-ring, etc., are not logged.

CLASS: F
(All logged errors are classified as fatal (F).)

REV: N/A

A: Bytes processed since BOT.

B: Total files processed since BOT.

C: Soft errors since BOT.

D: Block size at error.

E: Operation in progress:

Value	
0	test unit ready
1	rewind
3	get sense (status)
5	read block size
8	read
10	write
16	write file mark(s)
17	space tape
18	inquiry
19	verify
20	recover buffered data
21	mode select
25	erase
26	mode sense
27	load/unload

T: Drive status

Bit	
31	Hard error
30	Corrected error
29	Filemark encountered
28	ID Mark sensed
27	EOT sensed
26	Load point sensed
25	Formatter busy (tape active, streaming)
24	Data busy (tape active, command in progress)
23	Tape loaded and ready
22	Write protected
21	High speed selected
20	Tape is On-line
19	Tape is rewinding
18	not defined
17	Unit number of selected tape drive
16	Unit number of selected tape drive
15	Tape TC
14	SCSI TC
13	Block underrun
12	SCSI DMA generated parity
11	Tape overrun
10	Parity in
9	Tape DMA parity error
8	SCSI DMA parity error
7	Tape DMA length terminal count
6	SCSI DMA length terminal count
5	Manual SCSI DMA request
4	Manual Tape DMA request
3	Dynamic RAM refresh enable
2	SCSI to Tape
1	Tape DMA enable
0	SCSI DMA enable

U: Controller Status

This is a 4 byte field. The first and third bytes are sense bytes, and interpreted as follows:

Bit	
7	Filemark
6	End of Tape
5	Requested block length mismatch
4	not defined
3-0	Sense key, as follows:

0	No further information
1	Recovered error
2	Not ready
3	Medium error
4	Hardware error (device failure)
5	Illegal request
6	Unit attention
7	Write protected
8	Blank tape
9	not defined
A	not defined
B	Aborted command
C	not defined
D	Volume overflow
E	Miscompare
F	not defined

The second and fourth bytes contain secondary error codes, interpreted as follows:

Value	
00	No error
20	Illegal SCSI command
43	Data buffer parity error
50	No data detected
51	Function did not complete in specified time
52	Tape position error - BOT was not indicated after a Rewind, Load or Long Erase.
53	An error occurred before the requested tape drive command was completed.
54	Data buffer is not empty
55	Fixed bit is set while in variable block mode or is not set while in fixed block mode.
56	Data transfer error occurred, host to controller
57	Data transfer error occurred, controller to host
58	Verify command does not support byte compare mode
59	Space command does not support spacing to physical End of Data
5A	Diagnostic Self Test not supported
5B	Command Sequence Error
5C	Unit Select Error
5D	Variable block length greater than 64 KB
5E	Unable to obtain ownership of buffer
5F	Command parameter error
60	Status error from target (COPY command)

V: N/A

Z: N/A

2.3.3 Cartridge Tape

Major: 6

Minor: 0

Type:

-2	I/O error
-62	Operation has timed out
-63	Media is write protected
-89	Device is off-line
-91	End of data
-97	No cartridge
-98	Tape media error

CLASS:

F	Fatal error. Fatal errors are: controller errors, tape read/write abort errors, and device timeouts.
N	Non-fatal error. Non-fatal errors are: no cartridge, write protected cartridge, and drive not ready.
S	Summary of tape statistics. Summary statistics are logged whenever that tape cartridge is rewound. It is not an error, and is not logged in the "by type" log. The log fields have different meanings for type S entries, as described below.

REV: N/A

A: Block number at error (error types F and N)
Total number of blocks (512 bytes) processed (error type S).

B: Total files processed since BOT.

C: Soft error count as returned by the drive.

D: Underrun count as returned by drive.

E:

S	not used
F,N	Operation in progress:

Value	Command
18	append
17	raw write
16	raw read
5	read drive status
4	erase
3	retension
2	rewind
1	space forward tape
0	controller initialize

T: S not used
F,N Drive status

Bit	Status
15	Set if any bits in high byte are set
14	Cartridge is not in place
13	Catastrophic error
12	Write protected cartridge
11	End of media
10	Unrecoverable data error
9	Bad block not located
8	File mark read
7	Set if any bits in low byte are set
6	Illegal command
5	No data detected
4	Marginal block detected
3	Beginning of media
2	Bus parity error
1	End of recorded media
0	Power on/Reset

U: S not used
F,N Controller status

Bit	Status
15	Operation successful
14	End of data
13	IOPB parameter error
12	Chain processing terminated
11	Filler sent
10	Read to end of media
9	Bad data transfer
8	No cartridge
7	File mark detected
6	Write protected
5	Aborted
4	Not erased in area of append
3	Wrote to end of media
2	Powerfail/reset
1	Catastrophic error
0	Error, drive status required

V: N/A

Z: N/A

2.3.4 Floppy Disk (MAI 2000 only)

Major: 7

Minor: 0 - 31 = drive 0
32 - 63 = drive 1

The number indicates the disk partition number, usually only 0 or 32.

Type:
-63 Write protected
-86 Drive seek error or data transfer error
-99 Originally mounted diskette was not found

CLASS: N/A

REV: N/A

A: Starting block of transfer

B: Number of blocks requested

C: N/A

D: N/A

E: N/A

T: Driver status

Bit	Command
8	interrupt
7	error
4	seek
3	recal
2	write
1	read

U: Controller status

Bit	seek/recal	read	write
7	not ready	not ready	not ready
6	write protect	N/A	write protect
5	head loaded	N/A	write fault
4	seek error	record not found	record not found
3	CRC error	CRC error	CRC error
2	track 0	lost data	lost data
1	index pulse	data request (DRQ)	data request (DRQ)
0	busy	busy	busy

V: Drive control

Bit	Command
5	enable write verification
4	disable error messages
3	not used
2	retires disabled
1	not used

Z: N/A

2.3.5 Printer Filter (Printer)

Major: 8

Minor:

	MAI 2000	MAI 2500/3000/4000
0	CMB parallel port	CMB parallel port
1	CMB serial port A	CMB serial port A
2	CMB serial port B	CMB serial port B
3	port 1 on the first 4-way	port 1 on the first 4-way/8-way
4	port 2 on the first 4-way	port 2 on the first 4-way/8-way
5	port 3 on the first 4-way	port 3 on the first 4-way/8-way
6	port 4 on the first 4-way	port 4 on the first 4-way/8-way
7	port 1 on the second 4-way	port 5 on the first 8-way
8	port 2 on the second 4-way	port 6 on the first 8-way
9	port 3 on the second 4-way	port 7 on the first 8-way
10	port 4 on the second 4-way	port 8 on the first 8-way
11	port 1 on the third 4-way	port 1 on the second 4-way/8-way
12	port 2 on the third 4-way	port 2 on the second 4-way/8-way
13	port 3 on the third 4-way	port 3 on the second 4-way/8-way
14	port 4 on the third 4-way	port 4 on the second 4-way/8-way
15	port 1 on the fourth 4-way	port 5 on the second 8-way

and so on

Type:
- -62 Operation has timed out
- -108 Printer I/O transmission error

CLASS: N/A

REV: N/A

A: Current page number since open

B: Current line number

C: Current column position, usually the end of the line

D: MBF printer protocol:
- 1 on
- 0 off

E: Printer type

Bit	Printer
5	Okidata 82a
4	Diablo
3	MVP
2	Whisper
1	Tritel
0	Printronix

T: Status byte 1

Bit	
7	0
6	busy
5	paper out
4	VFC error
3	offline
2	not used
1	default VFC in effect
0	power up

U: Status byte 2

Bit	
7	0
6	acknowledge alternator flag
5	not used
4	buffer overrun
3	LRC error
2	receive overrun error
1	parity error
0	frame error

V: N/A

Z: N/A

2.3.6 CMB

Major: 9

Minor: 0 = CMB serial port A
1 = CMB serial port B

Type: 1

CLASS: N/A

REV: N/A

A: N/A

B: N/A

C: N/A

D: N/A

E: N/A

T: SCC status (type 1 errors; special receive conditions)

Bit	Error
7	end of frame
6	CRC/framing error
5	receive overrun error
4	parity error
3-0	not used

U: N/A

V: N/A

Z: N/A

2.3.7 Four and Eight Way Boards

Major: 10

Minor:

	MAI 2000	MAI 2500/3000/4000
0	port 1 on the first 4-way	port 1 on the first 4-way/8-way
1	port 2 on the first 4-way	port 2 on the first 4-way/8-way
2	port 3 on the first 4-way	port 3 on the first 4-way/8-way
3	port 4 on the first 4-way	port 4 on the first 4-way/8-way
4	port 1 on the second 4-way	port 5 on the first 8-way
5	port 2 on the second 4-way	port 6 on the first 8-way
6	port 3 on the second 4-way	port 7 on the first 8-way
7	port 4 on the second 4-way	port 8 on the first 8-way
8	port 1 on the third 4-way	port 1 on the second 4-way/8-way
9	port 2 on the third 4-way	port 2 on the second 4-way/8-way
10	port 3 on the third 4-way	port 3 on the second 4-way/8-way
11	port 4 on the third 4-way	port 4 on the second 4-way/8-way
12	port 1 on the fourth 4-way	port 5 on the second 8-way

and so on.

Type:

1	Special receive conditions
2	Controller errors
3	Command timeout
4	Z-80 busy

CLASS: N/A

REV: N/A

A: Controller type (4 or 8)

B: Firmware revision level (8-way only)

C: Hardware revision level (8-way only)

D: N/A

E: N/A

T: SCC status (type 1 errors; special receive conditions)

Bit	Error
7	end of frame
6	CRC/framing error
5	receive overrun error
4	parity error
3-0	not used

4-way controller status (type 2, 3 and 4 errors)

Code	Error
81	illegal command (software error)
83	bus error (hardware error)
	All other errors are not defined

U: MSW: Command (Type 2, 3 and 4 errors only)

01	configure port
02	data transfer (out)
03	status
04	load default configuration
05	load X-on
06	load X-off
07	single byte output
08	set X-on/X-off flow control
09	set DTR flow control
0A	enable/disable high bit mask
0B	download
0C	revision status
0D	port lump sum set-up
0E	DMA to host

LSW: Transfer length or byte value, if single byte transfer

V: Data address (type 2, 3 and 4 errors only)

Z: N/A

2.3.8 LAN

Major: 11

Minor: 0

Type:

-665	Board disable (see field D:)
-670	Hardware generated error or retry count exhausted
-671	Length too long for receive socket
-672	Uninitialized socket
-673	Invalid transmit control length
-674	Invalid socket number
-675	Receive socket busy
-676	Destination host specification bad

CLASS: N/A

REV: N/A

A: N/A

B: N/A

C: N/A

D: Condition causing LAN controller disable (error -665):

Code	Reason
1	bus error
11	LAN write timeout
21	another controller within the LAN driver has the same address

E: Address of station to which a transmission failed (errors -670 through -676).

T: N/A

U: N/A

V: N/A

Z: N/A

2.3.9 Winchester Disk

Major: 14

Minor:

0 - 31	= drive 0
32 - 63	= drive 1
64 - 95	= drive 2 (MAI 3000/4000 only)
96 - 127	= drive 3 (MAI 3000/4000 only)
128 - 159	= drive 4 (MAI 4000 only)
160 - 191	= drive 5 (MAI 4000 only)
192 - 223	= drive 6 (MAI 4000 only)
224 - 255	= drive 7 (MAI 4000 only)

The specific number indicates the disk partition.

Type:

-2	I/O error
-26	Device busy. The controller was busy when it was not expected to be (prior to outputting a command). The information in the remaining fields applies to the previous command.

CLASS:

A	ST506 error
B	SCSI sense error
C	SCSI driver detected error
D	SCSI driver fatal errors

REV: N/A

A: Starting block of transfer

B: Blocks requested

C: Software retry count
0 = last try
1 = first try

D: Controller status register (class A errors only).

Bit	
7	command line active
6	busy line active
5	message line active
4	reset line active
3	input register empty
2	operation complete
1	output register empty
0	bus error

E: N/A

T: Command that was being executed when the error occurred.

Class 00 commands:

Code	Command
0	test unit ready
1	rezero unit
3	request sense
4	format unit
8	read
10	write
11	seek
13	translate
19	write buffer
20	read buffer
21	mode select
26	mode sense
27	start/stop unit
28	receive diagnostic
29	send diagnostic

Class 01 commands:

Code	Command
25	read capacity
28	read
2A	write
2E	write and verify
2F	verify
31	search data equal

U: Error Status

Class A errors (ST506):

Bit	Code	Error	Possible failing component
31	0	invalid logical block address	
	1	valid logical block address	
30-24	0	no sense	N/A
	1	no index signal	drive interface, controller
	2	no seek complete	drive
	3	write fault	drive
	4	drive not ready	drive or drive interface
	6	no track 0	drive or drive interface
	10	ID CRC error	controller
	11	uncorrectable data error	media error
	12	ID address mark not found	media error or controller
	13	data address mark not found	media or controller
	14	record not found	media error or controller
	15	seek error	drive or media
	18	data check without retry	media
	19	ECC error during verify	media
	1A	interleave error	system software
	1C	blown format on drive	unformatted drive
	1D	self test failed	controller
	1E	defective track	media
	20	invalid command	system software
	21	illegal block address	system software
	23	volume overflow	software
	24	bad argument	system software
	25	invalid logical unit number	system software
23-21		reserved	
20-0		logical block address	

Class B errors (SCSI sense):

Bit	Code	Error	Possible failing component
31	0	invalid logical block address	
	1	valid logical block address	
30-24	0	no sense	N/A
	1	no index	drive interface, controller
	2	no seek complete	drive
	3	write fault	drive
	4	drive not ready	drive or drive interface
	6	rezero command failed	drive, drive interface
	10	ID check sum error	media
	11	uncorrectable data error	media
	12	ID sync error	media
	13	data sync error	media
	14	block not found	media
	15	seek fault	drive
	17	data retry (retries)	media
	18	data retry (ECC)	media
	19	grown defect list not found	media
	1C	primary defect list not found	drive
	20	invalid opcode	system software, controller
	21	illegal logical block address	system software, controller
	24	invalid field (CDB)	system software, controller
	25	invalid logical unit	system software, controller
	26	invalid field (parameter)	system software, controller
	27	write protected	drive
	28	medium change	drive
	29	power on/reset/bus device reset	drive
	2A	mode select parameters changed	drive
	31	format corrupted/unformatted	media
	32	no defect spare locations	drive
	40	buffer RAM failure	drive
	41	serial data path error	drive, drive interface
	43	message reject retry failed	drive, controller
	44	SCSI hardware/firmware error	drive, controller
	45	select/reselect error	drive, controller
	47	bus parity error	drive, drive interface
23-0		logical block address	

Class C errors (SCSI driver detected errors):

Bit	Code	Error	Possible failing component
31-24	B1	bus error	controller
	B2	busy	drive, controller
	B3	unsolicited target disconnect	drive
	B4	partial data transfer	drive
	B5	parity error (disk to host)	drive interface
	B6	parity error (host to disk)	drive interface
	B7	hardware error (SBIC)	controller
	B8	selection timeout	drive, drive interface
	BA	invalid parameters	system software

Class D errors (SCSI driver detected errors):

Bit	Code	Error	Possible failing component
31-24	D1	drive shut down	drive, controller
23-16	XX	Class B or C error code	
8-15	XX	Class B or C error code	
0-7	0	0	

V: Drive control

Bit	Command
6	enable overlapped seeks
5	not used
4	enable write verification
3	disable error messages
2	queue optimization disabled
1	retries disabled, report ECC
0	ECC and retries disabled

Z: N/A

SECTION 3
BASIC ALL PURPOSE SERVICE SYSTEM (BASS)

3.1 INTRODUCTION

BASS diagnostics are a set of programs designed to exercise peripheral devices through the operating system. The tests are designed to exercise and test a peripheral device while allowing normal system operation to continue. Accordingly, these tests assume the MAI 2000/2500/3000/4000 system is operating normally, with the possible exception of one or more peripherals.

The programs are written in Business BASIC, with the exception of the disk and tape exercisers which use some BOSS/IX commands, and the LAN exerciser which is written in C. The system consists of the BASS Monitor, which provides the central menu driven interface and records test results, and the following exercisers:

- Disk Exerciser
- Magnetic Cartridge Streamer (MCS) Exerciser
- Printer Exerciser
- Terminal Exerciser
- 1/2 Inch Magnetic Tape Streamer (MTS) Exerciser

Instructions for running the BASS Monitor and each exerciser are given in the following subsections. The entire BASS system is menu driven for ease of use.

3.2 SYSTEM REQUIREMENTS

3.2.1 Hardware Requirements

The BASS diagnostics require the following minimal hardware:

- A CPU with at least 128 pages of user memory
- A functioning VDT

The VDT need not be the system monitor. There must be at least 128 pages of memory available to this terminal for running BASS.

3.2.2 Software Requirements

The BASS diagnostics require the following minimal software:

- The BOSS/IX operating system, level 7.2A or higher.
- BASS software package, revision 7201 or higher.

3.3 RUNNING BASS

The BASS diagnostic programs are all in the **/sys/bass** directory. The BASS system is then run from BASIC with a START size of at least 128 pages of memory.

3.3.1 Starting BASS

To start BASS, do the following:

1. Log on and enter BOSS/IX command mode.

2. Change your working directory to **/sys/bass**.

   At the command prompt, enter:

   ```
   chdir /sys/bass
   ```

   All characters must be lower case.

3. Enter BASIC command mode.

   At the command prompt, enter:

   ```
   basic
   ```

   All characters must be lower case.

4. Start the necessary number of pages. At the BASIC command prompt, enter:

   ```
   START 128
   ```

   A larger number may be entered, but is not usually necessary.

5. Run the BASS diagnostics.

   At the BASIC command prompt, enter:

   ```
   RUN "BASS"
   ```

   The program name, BASS, must be in upper case letters.

The BASS menu is then displayed, as shown in Figure 3-1.

```
Basic/Four All-Purpose Service System   Rev: 72XX   time   date

    1.  B/4 Service System

    2.  Inspection Cycle

    3.  Burn-In Cycle

    4.  Display/Print Previous Test Results

    5.  Change System Time/Date

       Enter Your Selection, <CR> to Exit:
```

Figure 3-1. BASS Diagnostics Menu

3.3.2 BASS Testing Options

The first three options on the menu provide system testing:

- B/4 Service System allows you to select specific tests to run on a device. This option is generally used after the faulty device has been identified by either the Inspection Cycle or the Burn-In Cycle. You then select individual tests to run. These tests are run in a loop, in the order specified.

- Inspection Cycle provides automatic selection of tests for specified devices. The tests execute through one complete pass, then stop.

- Burn-In Cycle provides long term testing of the system. The testing is the same as for the Inspection Cycle except that the cycle repeats continuously, and special long term tests are included.

- Display/Print Previous Test Results allows you to report results from previously run BASS tests that are stored in the error log file.

To select a testing option, type the number of the option and press **ENTER**.

The Change System Date and Time option is self-explanatory, and so is not described in this document. Only the system administrator can change the date or time.

3.3.3. Test Result Summary Screen

The BASS summary screen reports the testing start time, the current time, date and the terminal from which BASS is being run, followed by a list of each test run and any errors that occurred during the test.

The summary screen is displayed continuously while the tests are running, and updated following each test. The test summary can also be output to a printer, as described in Section 3.3.6.

3.3.4 B/4 Service System Procedure

The B/4 Service System option allows you to select the tests to perform on a device already found to be defective. You select the tests from a menu listing all of the available tests. When selecting tests, you also have the option to display or print a description of the test before adding it to the test list.

The individual tests are described in sections 3.4.

To run the B/4 Service System, do the following:

1. **Select B/4 Service System.**

   At the BASS Diagnostics Menu, select option 1, "B/4 Service System."

2. **Do you want program descriptions?**

   When you select to run the B/4 Service System, you are first asked if you want a description of selected tests. This allows you to view or print descriptions of tests, and then to optionally include them in the test list.

   If you want program descriptions, type **Y** and press **ENTER**; otherwise, press **ENTER** alone.

   A menu of test programs is then displayed.

3. **Enter printer name.**

   (This step is skipped if you do not what program descriptions.)

   If you want the program descriptions printed, type the name of the printer and press **ENTER**. If you only want the descriptions displayed on the screen, press **ENTER** alone.

4. **Select Numbers:**

   Then menu lists the names of BASS test programs. Any currently selected items are marked with an asterisk (*).

   Enter the numbers of all tests you want to select or view, separated by commas, and press **ENTER**. During the execute cycle, programs are run in the order entered.

   If you are not viewing or printing program descriptions, the options prompt is displayed (step 5).

   If you are viewing or printing program descriptions, the description of each program is displayed or printed in the order contained in the list. After each description, you are asked:

   Include this program in the run list?

   To include the program, type **Y** and press **ENTER**; otherwise, press **ENTER** alone. When all program descriptions have been displayed, the options prompt is displayed (step 5).

5. **Enter One of the Following:**
a = Add Entries, d = Delete Entries, r = Review, <CR> = Execute:

A list of the currently selected test numbers is displayed.

If you select to Add or Delete entries, the appropriate prompt is displayed. Enter the numbers to be added or deleted, separated by commas. This prompt is then repeated.

If you select to Review your selections, test program menu is displayed and the Select Numbers prompt is repeated. (Repeat step 4.)

To begin the test cycle, press **ENTER** alone.

6. **Log Errors to a Printer?**

When you select the Execute option, you are asked if you want errors logged to a printer.

If you want the errors printed, type **T** and press **ENTER**. You are then asked for the name of the printer. Type the printer name and press **ENTER**.

7. **Log Errors in Error File?**

Whether or not you have errors logged to a printer, you also have the option of having them logged to an error file.

If you want errors logged, type **Y** and press **ENTER**; otherwise, press **ENTER** alone.

If you choose to log errors to the file, you are next asked if the file should be cleared first:

Clear error file?

To clear the file, type **Y** and press **ENTER**; otherwise press **ENTER** alone.

8. **Test Execution.**

Following the last prompt, the tests execute in the order entered. Individual tests are described later in this section.

3.3.5 Inspection and Burn-In Cycle Procedures

The Inspection and Burn-In Cycles run a series of tests on selected peripheral devices. You select the devices, but not the tests.

To run the Inspection or Burn-In Cycle, do the following:

1. **Select Cycle.**

   At the BASS Diagnostics Menu, select option 2 for the Inspection Cycle, or option 3 for the *Burn-In* Cycle.

2. **Select Tape Device.**

   You are first asked whether you want to test the 1/4 inch Magnetic Cartridge Steamer (MCS). To test this device, type **Y** and press **ENTER**; otherwise press **ENTER** alone.

   Next, you are asked whether you want to test the 1/2 inch Magnetic Tape Streamer (MTS). To test this device, type **Y** and press **ENTER**; otherwise press **ENTER** alone.

3. **Select Disk Device(s).**

   A list of configured disk drives is displayed, and you are asked to specify which drive(s) to test:

```
Configured Disks:
wd0  wd1  fd0

Enter Drive (<CR> to end):
```

   Type the menu number of each drive to test, separated by commas, then press **ENTER**. The system checks that the selected disks can be accessed by mounting them. Any disk that cannot be mounted is reported as such.

   For each disk that can be mounted, you are asked to select the filesystem(s) to be tested, for example:

```
Enter file system to be tested (<CR> to end):

1. wd0
2. wd1
3. fd0
```

   Type the number of each filesystem to be tested, separated by commas.

4. **Select Printers.**

A list of configured printers is displayed, and you are asked to specify which printer(s) to test:

```
Configured Printers:
LP  p0  p4

     Enter Number(s) of Device(s) to be tested:
```

For each printer, you are then asked for the printer type:

```
Enter Printer Type (<CR>=do not test) for Printer 'XX':
1 = 4201    5 = 4210     9 = 4214    13 = 4218
2 = 4203    6 = 4211    10 = 4215    14 = 4220
3 = 4206    7 = 4212    11 = 4216    15 = ISP
4 = 4208    8 = 4213    12 = 4217
```

Enter the type of each printer followed by **ENTER**.

5. **Select Terminals.**

Finally, you are asked for each terminal to test:

```
Enter Terminal <CR> to end:
```

The names of the configured terminals are shown on the screen. Type the name of a terminal to test and press **ENTER**. You are then asked for the terminal type:

```
Enter Terminal <CR> to end : t0
Enter Type: 1=7270/4301
            2=4309/4312
            3=4310/4313/4314
            4=4308 (Phase 1 ODT)
            5=4308 (Phase 2 ODT)
            <CR> = Do Not Test :
```

Type the number for the type terminal type followed by **ENTER**.

The testing cycle then begins. For the Inspection Cycle test, the cycle is executed once. For the Burn-In Cycle, the test is executed repeatedly.

3.3.6 Print/Display Previous Test Results

The Print/Display Previous Test Results option provides a report of the errors logged in the error file during a previously run test. The report can be displayed on the VDT or output to a printer.

To run this option, do the following:

1. **Select Print/Display Option.**

   Select the option 4 at the BASS Diagnostics Menu, "Print/Display Previous Test Results."

2. **Specify Output Device.**

   The error listing can be displayed on the VDT or output to a printer.

   To output the listing to a printer, type the name of the printer and press **ENTER**. To display the listing on the VDT, press **ENTER** alone.

3. **Printed Reports.**

   If you selected a printer as the output device, this message is displayed while it is being printed:

   Error File Being Listed to XX

   where XX is the two-character printer name.

   When printing is complete, this message is displayed:

   Printout Complete

   <CR> To Exit

   Press **ENTER** to exit and return to the BASS Diagnostics Menu.

4. **View Report on the Screen.**

   If you chose to view the report on the VDT, one screenful of information is displayed at a time. When a screen (page) is displayed, you are prompted:

   End of Page. <CR> to Continue, CTL-I to End:

   Press **ENTER** to view the next page, or **CTL-I** to end the report.

   If you press **CTL-I**, or are viewing the last page, you are prompted:

   Printout Complete

   <CR> To Exit

   Press **ENTER** to exit and return to the BASS Diagnostics Menu.

3.4 BASS TEST DESCRIPTIONS

The following paragraphs describe the tests available for selection in the B/4 Service System. The same tests are used during the Inspection and Burn-In Cycles, noted for each test.

Some of the tests can also be run directly from BASIC command mode. The program names are provided for this purpose.

3.4.1 DISK DEVICE TESTS

The Disk Device Tests are run by a single BASIC program: D01. This program can be run directly from BASIC command mode or by selecting it at the BASS Service System menu. The program in turn calls several overlay programs: D01-1 through D01-4. The overlay programs cannot be run directly, but must be called by D01.

There are four tests run on disk devices:

- Disk Dump Test (D01-1)
- Direct File Exerciser (D01-2)
- File Integrity Test (D01-3)
- Freespace Analysis and Filesystem Check Test (D01-4)

The test begins by determining what disks are configured on the system. When the disks have been determined, the user is asked to specify which to test:

```
Configured Disks:
fd0  fd1  wd0  wd1

Enter Drive (<CR> to end):
```

Each device selected is tested for availability by mounting. For each available device, the user is asked to specify the filesystem(s) to test. For example:

```
Enter filesystem to be tested (<CR> to end):

1. wd0
2. boot
3. root
```

Once the filesystems have been specified, the overlay programs are run. (The boot filesystem is not tested by the Direct File Exerciser (D01-2) or the File Integrity Test (D01-3).)

The individual tests are cycled through once for each device and filesystem before passing on to the next device or filesystem.

3.4.1.1 DISK DUMP TEST

The Disk Dump Test checks the availability of specific programs on the disk by using the BOSS/IX "dump" command. Programs are dumped in alphabetical order by directory.

The program searches the specified disks and filesystems for BASIC programs, and then dumps the programs. During the process the program displays the following messages, describing its current activity:

Determing Pathnames .....
Determing Filenames .....
Dumping Filenames .....

When the program is complete, it continues to the Direct File Exerciser.

3.4.1.2 DIRECT FILE EXERCISER

The Direct File Exerciser tests the system's ability to WRITE, READ and REMOVE records in a Direct file, using Business BASIC directives. The test is performed on each filesystem the user selected. The test is intended to verify data integrity on the disk.

During this test, a Direct file, named DRTFx, where "x" is the disk number, is repeatedly created, exercised, and erased. Each time the file is created, the key size, number of records, or record size is changed. READ, WRITE and REMOVE directives are executed in set patterns to test the file.

While a filesystem is being tested, "Direct File Exerciser" is displayed, followed by "Now Testing Device: XXX", where "XXX" is the name of the filesystem currently being tested.

All errors that occur are logged, indicating the time, BASIC command, key size, disk error, error status, statement number (in D01-2), and the key. If the data does not match what it should be, a message displays the expected data and the actual data read, along with the error parameters listed above.

3.4.1.3 FILE INTEGRITY TEST

The File Integrity Test checks the integrity of copies of BASIC programs between every directory in a filesystem. In this way, it tests the integrity of BASIC program storage and retrieval.

The test proceeds by first copying a BASIC program to a directory in the filesystem being tested, using the BOSS/IX **copy** command. The copy is then compared with the original using the BOSS/IX **bcompare** utility. The copy is then moved to another directory, using the BOSS/IX **move** command, and again compared to the original. The move and compare procedure is repeated until all directories have been tested. Finally, a new directory is created, the copy is moved to this new directory and compared to the original. The copy and new directory are then deleted from the system.

If an error occurs, the test continues by returning to the original program and copying it to the next directory. The routine then continues as before.

All errors that occur are logged, indicating the error which occurred, the time, the directory and the device.

3.4.1.4 FREESPACE ANALYSIS AND FILE SYSTEM CHECK TEST

The Freespace Analysis and File System Check Test verify the integrity of the filesystem.

A message is displayed indicating which filesystem is currently being checked:

Checking File system: XXX

where "XXX" is the filesystem name.

Freespace Analysis is performed first, using the BOSS/IX **freespace** utility. When the check is complete, a message reports the total space available, the space used, and the percentage of freespace remaining.

The File System Check is then performed, using the BOSS/IX **fschk** command. Messages report that the inodes, the block map and the reference count are checked and any problems.

Any errors are logged, indicating the time, the error which occurred, the device and the operation being performed when the error occurred.

3.4.2 Magnetic Cartridge Streamer Exerciser

The Magnetic Cartridge Streamer (MCS) Tests are run by a single BASIC program: M01. This program can be run directly from BASIC command mode or by selecting it at the BASS Service System menu. The program exercises the MCS using the BOSS/IX commands **tcompare, tlabel, tlist, tsave** and **trestore**.

Since the test writes on the MCS tape, a scratch tape must be used. Any data on the tape prior to the test will be destroyed.

The test runs for approximately 4 minutes for the Service System and Inspection Cycle tests, and approximately 20 minutes each pass of the Burn-In Cycle. The time required increases with system load.

When the test is run from the B/4 Service System or directly from BASIC command mode, this prompt is displayed:

> Insert Scratch Tape...<CR> when ready..

Insert a tape and press **ENTER**.

At the beginning of the Inspection and Burn-In Cycles, you are asked whether or not to run the test:

> Test Magnetic Cartridge Streamer? (y/<CR>):

To run the test, type **Y** and press **ENTER**; otherwise, press **ENTER** alone. If you choose to run the test, you are then prompted to insert a cartridge:

> Insert Scratch Tape...<CR> to Continue:

Insert a tape and press **ENTER**.

The individual tests are then performed in this order:

- MCS Label Test
- MCS Save Test
- MCS List Test
- MCS Compare Test
- MCS Restore Test

Any errors that occur are logged, including the name of the program and the statement number being executed when the error occurred, the time the error occurred, and a description of the error.

3.4.2.1 MCS LABEL TEST

The MCS Label test checks the system's ability to read the MCS label.

For the B/4 Service System, the label is read and displayed. You are then prompted for an action:

Write New Label, Insert New Tape, <CR> to Continue, MB-IV to Exit

(w/i/<CR>/MB-IV):

Writing a new label initializes the tape and continues. Inserting a new tape and entering I reads the label of the new tape. Press **ENTER** to continue with the next test.

For the Inspection and Burn-In Cycles, the test continues immediately if the label can be read.

For all testing cycles, if the label cannot be read, the MCS test sequence aborts.

3.4.2.2 MCS SAVE TEST

This test begins by creating and filling a 4 KB file on disk, and then saves the file to MCS using the **tsave** command.

If the file is written successfully, the MCS List test is executed. If the write is not successful, an error message is displayed and the MCS test aborts.

3.4.2.3 MCS LIST TEST

After the 4KB file is successfully written to tape, an attempt is made to read the file using the **tlist** command.

If the file is read successfully, the contents are listed on the screen. If the read is not successful, an error message is displayed. In either case, the MCS test continues.

3.4.2.4 MCS COMPARE TEST

This test compares the file on MCS with the original file on disk using the **tcompare** command.

If the file compares successfully, the MCS Restore test is executed. If the write is not successful, an error message is displayed and the MCS test aborts.

3.4.2.5 MCS RESTORE TEST

This test first deletes the original 4 KB file on disk. Then an attempt is made to restore the file from MCS using the **trestore** command.

If the restore is successful, the file on tape and the restored file on disk are compared using **tcompare**.

If an error occurs, it is logged and displayed (optional). The file is also compared in BASIC, and any discrepancies are reported.

This concludes the Inspection Cycle testing.

3.4.2.6 ADDITIONAL BURN-IN CYCLE TESTING

Additional write/compare/restore/compare testing is performed by Burn-In Cycle testing. The general test cycle is as follows.

First, files are created on disk and then written to the MCS tape using tsave. These files are then compared to the originals on disk. If the compare is successful, then the original files are deleted from disk and the files on tape are restored using trestore. The files on tape and on disk are again compared using tcompare and are also compared in BASIC. Any errors are logged and optionally displayed.

This testing cycle is repeated three times for each of three sets of files, for a total of nine write/compare/restore/compare tests:

- Ten 1 byte files
- Two 10 KB files
- Six files of different lengths as follows: a 4 KB file, a 1 byte file, a 10 KB file, a 1 byte file, a 4 KB file, and a 10 KB file

3.4.3 Printer Exerciser

The Printer Exerciser Test is a series of four BASIC programs:

- Printer Quality Test (P01)
- Print Quality Check Test (P02)
- Printer Ripple Test (P03)
- Printer Function Test (P04)

These programs exercise any printer supported by the MAI 2000/2500/3000/4000 systems.

All functions supported by the printer and printer driver are tested. The programs are written in BASIC, and so use the printer driver through printer mnemonics. (Refer to the BB/86 Reference Manual (M6262) for a complete description of the printer mnemonics.)

All four tests can be selected at the BASS test menu in the B/4 Service System, or can be run directly from BASIC command mode. Tests are selected automatically by the Inspection and Burn-In Cycles.

The printer must be correctly set up before running these tests, including proper cabling, jumper settings and dip switch settings. Several ERROR 5's will occur if the printer is incorrectly set up.

3.4.3.1 PRINTER QUALITY TEST

This test provides a sample of basic print quality. It does not test printer features. This test is not included in the Burn-In Cycle.

3.4.3.2 PRINT QUALITY CHECK TEST

This test produces printed output providing a visual check of the printer's vertical and horizontal spacing accuracy. The results can be checked against the tolerances in the printer's service manual to determine if a fault exists.

The test begins by printing a form feed ('FF' mnemonic) followed by:

Line Spacing, Horizontal Line Skew, Vertical Registration and Horizontal Land Width Test:

A blank line ('LF') is then printed, followed by ten lines of E's. Another blank line is then printed, followed by the message:

Now Switching to 8 Lines per Inch.

Another blank line is printed, followed by ten more lines of E's.

Use the lines of E's to check line spacing, horizontal line skew, vertical registration and horizontal land width.

Next, the test prints five blank lines and the message:

Horizontal Registration Test

This is followed by a blank line and ten lines of H's.

Use the lines of H's to check the horizontal registration.

Another five blank lines are printed, followed by:

Vertical Land Width Test

This is followed by a blank line and ten lines of M's.

Use the lines of M's to check the vertical land width.

This ends the Print Quality Check test.

3.4.3.3 PRINTER RIPPLE TEST

This test prints an endless ripple pattern, shifting the character sequence by one column for each new line. Printing continues until you press **ESCAPE** on the keyboard or the printer is taken off-line.

This test is not run by either the Inspection or Burn-In Cycles. It can be selected at the B/4 Service System menu or can be run directly from BASIC command mode.

3.4.3.4 PRINTER FUNCTION TEST

This test checks all printer features supported by the printer drivers. The features available vary slightly with the printer and printer driver. Tests for features not supported by a printer are excluded from the test. For this reason, the test begins by asking for which printers are to be tested and for the printer types.

There are sixteen different subtests to exercise printer features. These are described in the following subsections.

When the Printer Function Test starts, a prompt is displayed asking for the names of printers to test:

Enter the Number of the Printer(s) to be Tested:

Type the names of the configured printers you want to test, separated by commas or spaces, e.g.:

P0,P2,P5

For each printer listed, you must specify the printer type. The printer type is used to select tests. Printer types are listed by model number in a menu:

```
Printer Types:
1 = 4201   5 = 4210  9  = 4214  13 = 4218
2 = 4203   6 = 4211  10 = 4215  14 = 4220
3 = 4206   7 = 4212  11 = 4216  15 = .ISP
4 = 4208   8 = 4213  12 = 4217

Enter Printer Type for 'P1':
```

Most printers also require that you specify the paper size:

```
Paper Sizes:
   1 = 11 3/4 x 8 1/2"
   2 = 9 1/2 x 11"
   3 = 14 7/8 x 11"

Enter Paper Size for Printer 'P1'
```

You are also prompted for the printer serial number:

Enter Serial # for Printer 'P1': (<CR> to Continue)

The serial number is optional. If you enter it, it is printed at the beginning of the test along with the printer name.

For some printers this message is displayed:

Printer 'XX' Must be Set With Default LPI. (<CR> to Continue)

Make sure the printer is set correctly before continuing.

When all of the required information has been entered, the first test is begun on the first printer.

If any BASIC errors occur during the tests, they are logged. The information entered in the log includes the time the error occurred, the printer it occurred on, the function being tested, the statement number the error occurred on, and the BASIC error number. Error information is displayed on the terminal when it occurs, and is output to a printer if you specified one for error logging.

The remaining paragraphs describe the individual printer tests.

3.4.3.4.1 Ripple Pattern Test

This exerciser tests the ability of the printer to print every printable ASCII character at every column position. Printable ASCII characters are characters 32 to 126 (7-bit ASCII values), excluding character 96, a total of 94 characters.

The test prints 94 rows, printing each character on each row. On each successive row the character pattern is advanced one column to the right.

3.4.3.4.2 Line Width Density Test

This test exercises the BASIC '10', '12' and '16' mnemonics. The test prints eight rows of the ripple pattern in either 12 characters per inch (cpi) or 16 cpi if the printer has that capability. The printer is then returned to the default 10 cpi using the '10' mnemonic and prints eight more rows of the ripple pattern.

3.4.3.4.3 Line Height Density Test

This test exercises the BASIC '8L' and '6L' mnemonics. The test prints eight rows of the ripple pattern in the default 6 lines per inch (lpi). It then shifts to 8 lpi using the '8L' mnemonic. If the printer supports the '6L' mnemonic, it shifts back to 6 lpi and prints eight more lines.

3.4.3.4.4 Underline Test

This test exercises the 'BU' and 'EU' mnemonics. The test outputs this line:

The Word Underline Should be Underlined

3.4.3.4.5 Carriage Positioning Test

This test exercises the 'CR' mnemonic by printing an entire row of upper-case H's. First it prints an H at the left margin, and then an H at the right margin. It then returns to print an H one character right of the first H, then one character left of the last H. This alternation continues until the entire row is filled. Inspection of the line will determine the printer's accuracy.

For the Burn-In Cycle, this test is continued for an entire page.

3.4.3.4.6 Expanded Ripple Pattern Test

This test exercises the printers ability to print every character in every position in expanded mode, using the 'EP' mnemonic. The same pattern used in the ripple pattern test is used. The test is accomplished in 96 lines.

3.4.3.4.7 New Line Test

This test exercises the BASIC 'NL' mnemonic. The output of the test is:

This is the Line Preceding the 'New Line'
This is the Line Following the 'New Line'

Verify that the lines are separated and aligned.

3.4.3.4.8 Overprint Test

This test exercises the BASIC 'OP' mnemonic. The test output is a line of zeros (0) with a line of X's printed over them:

3.4.3.4.9 Plot Mode Test

This test exercises the printer's ability to enter plot mode and decode plot data. Each line of data is preceded with the BASIC 'PM' mnemonic.

All plot data on a single row has the same value. This row is repeated ten times to make verification by visual inspection easy.

Plot data consists of six bits. The following possible bit patterns are plotted: all single bit patterns (6 possible), all two consecutive bit patterns (5 possible), all three consecutive bit patterns (4 possible), all four consecutive bit patterns (3 possible), all five consecutive bit patterns (2 possible), and the one six consecutive bit pattern, for a total of 21 patterns. The actual plot data used to create this pattern is (bit 7 is always set for plot data):

64,65,66,68,80,96,67,70,76,88,112,71,78,92,120,79,94,124,95,126,127

3.4.3.4.10 Ring Bell Test

This test exercises the 'RB' mnemonic. The test sends the mnemonic to the printer five times. The bell on the printer should ring once every second for five seconds.

3.4.3.4.11 Bold Print Test

This test exercises the BASIC 'SF' and 'SB' mnemonics. The test outputs the line:

The word **BOLD** should be in bold print.

3.4.3.4.12 Load/Test VFU Test

This test exercises the printer's ability to load a Vertical Form Unit (VFU) definition, and to slew to the defined lines. The VFU is defined with the following positions:

Channel 1	- Top of Form
Channel 2	- Row 5
Channel 3	- Row 9
Channel 4	- Row 13
Channel 5	- Row 17
Channel 6	- Row 21
Channel 7	- Row 25
Channel 8	- Row 29

Defining the VFU uses the 'SL' and 'EL' mnemonics.

The test then slews to all eight channels. After the slew is performed, a message is printed stating which channel has been slewed to. Slews are performed in this order: 'FF', 'S1', 'S2', 'S3', 'S4', 'S5', 'S6', 'S7', 'S8'. The vertical tab mnemonic, 'VT' is then tested by issuing a 'FF' followed by 'VT'. The vertical tab mnemonic slews to channel 6.

To verify the test results, check that each slew message is four lines below the previous slew message. The message "Slew to Channel 1" should be printed at the top of the first page. The message "Form Feed" should be printed at the top of the second page, follows by "Vertical Tab" at line 21.

3.4.3.4.13 Super/Subscript Test

This test exercises the 'SP' (superscript) and 'SS' (subscript) mnemonics. The printer should output these messages:

The Word ^{SUPER} Should be in Superscript

The Word $_{SUB}$ Should be in Subscript

3.4.3.4.14 Bin Sheet Feed Test

This test exercises the 'B1' and 'B2' mnemonics on letter quality printers with cut sheet paper feeders. The test prints the following lines on a sheet of paper from the appropriate bin:

This sheet should be supplied by Bin 1

This sheet should be supplied by Bin 2

3.4.3.4.15 Worst Case Pattern Test

This test generates a worst case pattern for letter quality printers by printing characters from opposite sides of the print wheel. Maximum travel of the print wheel is forced for each character. The pattern is printed on ten consecutive lines.

3.4.4 Terminal Exerciser

The Terminal Exerciser checks the functionality of all terminals supported by the MAI 2000/3000/4000 systems. The tests can be individually selected from the B/4 Service System menu or run directly from BASIC command mode. The tests are:

- Keyboard Echo Test (V01)
- General Terminal Exerciser (V02)
- 4308/4309/4312 Exerciser (V03)
- 4310/4313/4314 Exerciser (V04)
- 4310 Graphics Exerciser (V05)

Except for the Keyboard Echo Test, all tests are run without user intervention. The 4310 Graphic Exerciser requires visual inspection to verify functionality.

Terminals should be set with Xon/Xoff protocol for these tests. The ODT (model 4308) must be set in 7270 emulation mode with Auto Xon/Xoff protocol, and specified as a 7270/4301 terminal.

3.4.4.1 RUN INSTRUCTIONS

In the B/4 Service System, the test begins by asking which terminals are to be tested, as show in Figure 3-2 (test V03 in the sample). The screen indicates the terminals configured on the system and the terminal being used to run BASS. To select a terminal, type its name and press **ENTER**. You are then asked for its terminal type.

```
V03 - 4308/4309/4312 EXERCISER PROGRAM

      Number of Terminals Configured: 4
      Terminals Configured:                          BASS Console is T0
      T0  T3  T1  T2



                     Selected Terminals:


                                Enter Terminal <CR> to end:
```

Figure 3-2. Terminal selection screen.

With the exception of the Keyboard Echo Test, all tests need to know the type of terminal being tested.

```
Enter Type:  1 = 7270/4301
             2 = 4309/4312
             3 = 4310/4313/4314
             4 = 4308 (Phase 1 ODT)
             5 = 4380 (Phase 2 ODT)
             <CR> = Do Not Test
```

Type the number of the correct terminal type and press **ENTER**. Press **ENTER** alone to skip this terminal. The B/4 Service System, will verify your entry, and prompt for a correct entry if you select the wrong type.

For the Inspection and Burn-In Cycles, it is assumed that all terminals will be tested, and the terminal type prompt is displayed for all configured terminals. The system does not verify that you have entered the correct terminal type.

The tests begin as soon as the terminals have been identified.

3.4.4.2 TEST DESCRIPTIONS

3.4.4.2.1 Keyboard Echo Test (V01)

The Keyboard Echo Test is run only from the B/4 Service System menu or directly from BASIC command mode. The test requires user intervention.

When the test starts, the user is prompted with:

Press key, (<ESC> to exit):

When you press a key other than **ESCAPE**, the test displays the character's hex value, the character as it appears on the screen and the decimal value for the key as a value between 0 and 127.

When you press **ESCAPE**, you are asked to verify exiting the program:

Continue test? (Y,<CR>):

Type **Y** and press **ENTER** to continue with the keyboard test. Press **ENTER** alone to exit the test and display the BASS Summary Screen.

3.4.4.2.2 General Terminal Exerciser - Multiple VDT (V02)

This test exercises general features of all terminals supported by BOSS/IX. It is useful primarily in the Inspection Cycle.

The program performs four subtests as follow:

Print to every Position (V02-1)

This test verifies that printable characters can be displayed at each column on each line. This is done by printing a line of ripple pattern characters on every line.

Scrolling Test (V02-2)

This test verifies full scrolling of all 80 columns of the display. This test should be run when scrolling appears to be adding or dropping characters.

Print @ Test (V02-3)

This test verifies function of the BASIC print location mnemonic on terminals. A full screen is constructed using the @(x,y) positioning expression in BASIC. The screen pattern is then read, line by line, and compared against the pattern written. Characters that do not compare correctly are logged as errors.

Control Function Test

This test verifies the terminal control functions (clear screen, cursor home, foreground, background, etc.) and control characters (escape, etc.). The control functions and characters are tested for their intended effects, and any errors are logged.

3.4.4.2.3 4308/4309/4312 Exerciser - Multiple VDT (V03)

This program exercises most features supported by the 4308 (ODT), 4309 (EVDT) and 4312 (EVDTB) terminals to determine if the terminal is functioning properly. All BASIC errors and data mismatch errors are logged.

The test consists of six subtests, as follow:

Character Display Test (V03-1)

This test displays a ripple pattern to the entire screen. Each character position is then read and verified against what was written. Any mismatches are logged as errors.

Attribute Test (V03-2)

This test displays lines with all possible combinations of attributes. Each attribute is displayed in two consecutive lines; the first line indicates which attribute is being tested, the second line prints a line of characters the entire width of the pattern.

Status Line Test (V03-3)

This test varies depending on the terminal type:

4308 - Each status line is displayed, and several fields within the status lines are changed to show alternate settings.

4309 - This test displays the setup screen, and then displays individual setup lines. Then the test writes a title and sets the time on the terminal status line.

Graphics Test (V03-4)

This test varies depending on the terminal type.

4308 - The test draws rectangular boxes using graphics characters. It also displays all other special characters. The test is performed in both 80-column and 132-column screens.

4309 - The test uses the G1 character set, consisting of block graphics and international characters. Rectangular boxes are drawn using the international characters. Displayable control codes are then printed in normal and double-wide sizes. Lines made up of every character in the block graphics and international character sets are printed in both normal and double-wide sizes.

Scroll Test (V03-5)

This test varies depending on the terminal type.

4308 - The test sets a scroll region, and then text is scrolled up and scrolled down in the region. This is done for jump scroll, smooth scroll and slow scroll. The test is first performed for an 80-column display, and then for a 132-column display.

4309 - The test sets a screen region of six lines, and then scrolls a ripple pattern within the region. After ten lines have been scrolled, the six-line screen region is then set down one line. The scrolling pattern is then repeated for another ten lines. The cycle of lowering the region and testing is repeated until all 24 lines have been used. Both normal and smooth scroll are tested.

Alignment Test (V03-6)

This test varies depending on the terminal type.

4308 - The test fills the screen with vertical lines in the 80 and 132 column modes.

4309 - The test fills the screen with vertical and horizontal lines.

3.4.4.2.4 4310/4313/4314 Exerciser - Multiple VDT (V04)

This program exercises most features supported by the 4310 (EDT), 4313 and 4314 terminals to determine if the terminal is functioning properly. All BASIC errors and data mismatch errors are logged.

The test consists of six subtests, as follow:

Character Display Test (V04-1)

This test displays a ripple pattern to the entire screen: Each character position is then read and verified against what was written. Any mismatches are logged as errors. This test is run in both 80-column and 132-column modes.

Attribute Test (V04-2)

This test displays lines with all possible combinations of attributes. Each attribute is displayed in two consecutive lines; the first line indicates which attribute is being tested, the second line prints a line of characters the entire width of the screen. This test is run in both 80-column and 132-column modes.

Status Line Test (V04-3)

This test writes a title and sets the time on the terminal status line.

Graphics Test (V04-4)

The test uses the G1 character set, consisting of block graphics and international characters. Rectangular boxes are drawn using the international characters. Displayable control codes are then printed in normal and double-wide sizes. A line made up of every character in the international character set is printed in both normal and double-wide sizes. This test is run in both 80-column and 132-column modes.

Scroll Test (V04-5)

The test sets a screen region of six lines, and then scrolls a ripple pattern within the region. After ten lines have been scrolled, the six-line screen region is then set down one line. The scrolling pattern is then repeated for another ten lines. The cycle of lowering the region and testing is repeated until all 24 lines have been used. Both normal and smooth scroll are tested. This test is run in both 80-column and 132-column modes.

Alignment Test (V04-6)

The test fills the screen with vertical and horizontal lines. This test is run in both 80-column and 132-column modes.

3.4.4.2.5 Graphics Exerciser (V05)

This test exercises the functionality of the 4310 graphics board. The test does not log errors and requires visual verification of the graphics patterns it creates.

The program produces three separate graphic displays:

- The first test plots a straight line, draws a circle, and draws a solid star.
- The second test displays fifteen lines, three each of the line types: dotted, dot-dashed, short dashed, long dashed, and solid. It then displays eight character strings showing the terminal's four character sizes in both character modes.
- The third test displays three circles with varying arc lengths and increment values. Then eight stars are displayed with each star moved to the right, filled with a different pattern and increased in size.

3.4.5 LAN Exerciser

The LAN exerciser demonstrate the basic functionality of a local area network (LAN). The goal of the exercisers is to learn as much as possible about the functioning of the LAN controller.

There are six programs in the LAN test: five test programs and one topography maintenance program. The programs can all be individually selected at the B/4 Service System menu. The programs are:

- LAN Initialization (L01) insures that the LAN hardware and software are present and minimally operational.
- LAN Who's There Test (L02) determines which LAN controllers are on the network and able to respond.
- Lan Network Test (L03) Exercises the connections between the initiating controller and all controllers responding to L02.
- Remote LAN Who's There Test (L04) performs a "who's there" sequence. It also sends, receives and echos a series of messages.
- Remote LAN Network Test (L05) exercises a selected pair of LAN controllers.
- Expected LAN Topography MAP Editor (L06) maintains a table describing the LAN system. This program should be run before any of the tests.

3.4.5.1 ADDITIONAL SETUP REQUIREMENTS

The LAN exerciser requires at least two systems to be running on the local area network. Three systems are required for L04 and L05. All systems must be active and ready.

All systems must have the same revision of the LAN software.

The Remote LAN Network Test (L05) also requires that the BASS files be located on one of the remote systems being tested.

The Expected LAN Topography MAP Editor (L06) uses the data files LMAP1 and LMAP2. If LMAP1 does not exists, execute this command at the BOSS/IX command prompt:

L06 MAP

You must be logged on as the system administrator to execute this command.

3.4.5.2 RUN INSTRUCTIONS

The LAN exercisers are run from the B/4 Service System menu, or automatically by the Inspection and Burn-In Cycles.

If the test is being run from the B/4 Service System, the Expected LAN Topography MAP Editor (L06) should be run FIRST. This sets up a table needed by the other tests. Then the LAN Initialization test (L01) should be run. These tests should be run only once.

3.4.5.3 TEST DESCRIPTIONS

3.4.5.3.1 LAN Initialization (L01)

This test insures that the LAN hardware and software on the system initiating the exerciser is present and functioning well enough to test. If the test fails, an off-line diagnostic is required to isolate the fault. Proceeding with other on-line tests will produce unpredictable results.

Four items are checked by this test, including the commands **who**, **getstat**, and **flushq**. The files "lmap1" and "lmap2" are also checked.

3.4.5.3.2 LAN Who's There Test (L02)

This test determines what LAN controllers are on the network and capable of responding. The actual results are maintained in a table and displayed. The results are compared to the values in the Expected Topography table created by L06. Any discrepancies are reported.

Testing is performed on those controllers detected as being active.

3.4.5.3.3 LAN Network Test (L03)

This test exercises the connections between the initiating controller and all controllers found to be active by L02. A number of messages are sent to and echoed back from each of the active controllers. At the end of the test sequence, the success rate for each controller is displayed.

3.4.5.3.4 Remote LAN Who's There Test (L04)

This test starts a "who's there" sequence at a remote controller. The results are displayed, but no table is kept.

A series of messages is then sent, returned and echoed. The results are compared and displayed.

3.4.5.3.5 Remote LAN Network Test (L05)

This test exercises a selected pair of LAN controllers. The test requires user intervention to select the two stations to test. Remote services are used, so it is necessary for the BASS files to be located on one of the remote system to be used.

3.4.5.3.6 Expected LAN Topography MAP Editor (L06)

This is the first program that should be run. It is a utility program that maintains a table mapping the LAN system. The map is needed by the remaining tests (L01 through L02).

3.4.6 Magnetic Tape Streamer Exerciser

The Magnetic Tape Streamer (MTS) Tests are run by a single BASIC program: T01. This program can be run directly from BASIC command mode or by selecting it at the BASS Service System menu. The program exercises the MTS using the BOSS/IX commands **tcompare**, **tlabel**, **tlist**, **tsave** and **trestore**.

Since the test writes on the MTS tape, a scratch tape must be used. Any data on the tape prior to the test will be destroyed.

The test runs for approximately 2 minutes for the Service System and Inspection Cycle tests, and approximately 10 minutes each pass of the Burn-In Cycle. The time required increases with system load.

When the test is run from the B/4 Service System or directly from BASIC command mode, this prompt is displayed:

> Insert scratch tape... <CR> when ready..

Insert a tape and press **ENTER**.

At the beginning of the Inspection and Burn-In Cycles, you are asked whether or not to run the test:

Test Magnetic Tape Streamer? (y/<CR>):

To run the test, type **Y** and press **ENTER**; otherwise, press **ENTER** alone. If you choose to run the test, you are then prompted to insert a cartridge:

Insert Scratch Tape... <CR> to Continue:

Insert a tape and press **ENTER**.

The individual tests are then performed in this order:

- MTS Label Test
- MTS Save Test
- MTS List Test
- MTS Compare Test
- MTS Restore Test

Any errors that occur are logged, including the name of the program and the statement number being executed when the error occurred, the time the error occurred, and a description of the error.

3.4.6.1 MTS LABEL TEST

The MTS Label test checks the system's ability to read the MTS label.

For the B/4 Service System, the label is read and displayed. You are then prompted for an action:

Write New Label, Insert New Tape, <CR> to Continue, MB-IV to Exit

(w/i/<CR>/MB-IV):

Writing a new label initializes the tape and continues. Inserting a new tape and entering **I** reads the label of the new tape. Press **ENTER** to continue with the next test.

For the Inspection and Burn-In Cycles, the test continues immediately if the label can be read.

For all testing cycles, if the label cannot be read, the MTS test sequence aborts.

3.4.6.2 MTS SAVE TEST

This test begins by creating and filling a 4 KB Indexed file on disk, and then saves the file to MTS using the tsave command.

If the file is written successfully, the MTS List test is executed. If the write is not successful, an error message is displayed and the MTS test aborts.

3.4.6.3 MTS LIST TEST

After the 4KB file is successfully written to tape, an attempt is made to read the file using the tlist command.

If the file is read successfully, the contents are listed on the screen. If the read is not successful, an error message is displayed. In either case, the MTS test continues.

3.4.6.4 MTS COMPARE TEST

This test compares the file on MTS with the original file on disk using the tcompare command.

If the file compares successfully, the MTS Restore test is executed. If the write is not successful, an error message is displayed and the MTS test aborts.

3.4.6.5 MTS RESTORE TEST

This test first deletes the original 4 KB file on disk. Then an attempt is made to restore the file from MTS using the trestore command.

If the restore is successful, the file on tape and the restored file on disk are compared using tcompare.

If an error occurs, it is logged and displayed (optional). The file is also compared in BASIC, and any discrepancies are reported.

This concludes the Inspection Cycle testing.

3.4.6.6 ADDITIONAL BURN-IN CYCLE TESTING

Additional write/compare/restore/compare testing is performed by Burn-In Cycle testing. The general test cycle is as follows.

First, files are created on disk and then written to the MTS tape using tsave. These files are then compared to the originals on disk. If the compare is successful, then the original files are deleted from disk and the files on tape are restored using trestore. The files on tape and on disk are again compared using tcompare and are also compared in BASIC. Any errors are logged and optionally displayed.

This testing cycle is repeated three times for each of three sets of files, for a total of nine write/compare/restore/compare tests:

- Ten 1 byte files
- Two 10 KB files
- Six files of different lengths as follows: a 4 KB file, a 1 byte file, a 10 KB.file, a 1 byte file, a 4 KB file, and a 10 KB file

SECTION 4
THE DIAGNOSTIC EXECUTIVE

4.1 OVERVIEW

This section describes the procedures for loading and using the Diagnostic Executive. It also describes the commands provided by the executive, which are also available during the Logic and Function Select Tests, and error messages displayed by the executive.

The Diagnostic Executive is a stand alone, bootable program which must be loaded prior to running the System Interaction Test (SIT), Logic Tests, Function Select Tests, and a few other special purpose diagnostic programs. SIT, the Logic Tests and the Function Select Tests are described in Sections 5, 6 and 7, respectively.

4.2 LOAD DIAGNOSTIC EXECUTIVE

To load the diagnostic executive, you must perform an alternate system load to the diagnostic medium. This is usually either MCS or MTS, unless the diagnostics have been installed on the fixed disk.

To load the diagnostic executive, do the following:

1. **Shut down the system to the reboot point.**

   If the system is running, perform a system shutdown. If the system is in multi-user mode, use the command:

   ```
   shutdown 0
   ```

   This will either shut down the system to the reboot point or to single user mode. The reboot point is either the reboot prompt (MAI 2000) or the reboot menu (MAI 2500/3000/4000).

   If the system is in single user mode, as indicated by the ADMIN prompt, press **CTRL + D**, and then enter **shutdown** to complete the shutdown.

2. **Mount the diagnostic medium.**

   If you are booting fixed-disk resident diagnostics, there is no medium to mount.

   If you are booting floppy, MCS or MTS based diagnostics, mount (insert) the diagnostic medium and make sure it is ready. The medium is labelled: DIA.

3. **Select the Alternate Load Procedure.**

   For the MAI 2000, press **CTRL + C** at the reboot prompt.

   For the MAI 2500/3000/4000 systems, select item 2, "Alt-load", at the reboot menu.

4. **Specify Boot Device.**

   You must specify the device containing the diagnostic programs at the prompt:

   Boot Device:

   Type the name of the appropriate device and press **ENTER**. The device names are:

   fd0 - floppy drive (MAI 2000 only)
   cs - MCS drive
   ts - MTS drive
   wd0 - fixed disk

5. **Specify System File.**

   You must specify the system file at the prompt:

   System file:

   For fixed disk resident diagnostics type:

   diag

   (/sys/diag on the MAI 2000) and press ENTER.

   If you specified the floppy, MCS or MTS as the boot device, press **ENTER** alone. The system file is the default file for these devices.

The system now loads the diagnostic executive and displays its prompt:

<exec>

At this point you can execute any of the executive commands, or load and run SIT, Logic Tests, or Function Select Tests.

4.3 EXECUTIVE COMMANDS

4.3.1 Executing Commands

The Diagnostic Executive supports several commands. To execute any of these commands, the user types the command name, followed by any required and optional parameters, and then presses ENTER. The command name and parameters may be entered in either upper case or lower case letters.

The commands can be entered any time the executive prompt, <exec>, is displayed. The commands may also be entered at the prompt for any Logic Test or Function Select Test. The commands are not available when SIT is loaded.

For example, the executive command used to load another diagnostic test, is the **LOAD** command. It is entered, followed by the name of the test to load:

<exec> load mcs

All commands are terminated by pressing **ENTER**.

Some Logic tests are manual intervention tests. These tests require that the operator respond to a prompt before the test is run. They are usually either potentially destructive to data (e.g., perform disk writes) or require special equipment (e.g., an external loopback plug). Manual intervention tests are run only if the **OPTIONS MANUAL** command has been executed.

4.3.2 Command Descriptions

The following commands are available through the diagnostic Executive. Some are of use only by Logic or Function Select tests, as noted in the descriptions.

ALL

This command starts running all logic tests. Each logic test is successively loaded and run. Refer to Section 6 for a description of the logic tests.

APNDTAPE

This command retensions the MCS tape, seeks the current end of tape, and copies the entire disk in drive 0 to the cartridge. The volume label, loader, and Exec are not written to the tape.

AUTO

This command executes all logic tests except those requiring manual intervention. (See the ALL command, and refer to Section 6.)

BUILD {*x*}

This command begins building a test loop of Function Select tests. The parameter is either "1" or "2", and specifies which test loop is being constructed. The prompt is changed to <loop*x*>, where *x* is either "1" or "2". Refer to Section 7 for information on building Function Select test loops.

CACHE *ep*

This command is available only for MAI 2500/3000/4000 systems. The command enables/disables the various caches available on these systems. The parameter is one byte of hexadecimal data.

The low nibble, *p* (bits 1-4) is loaded directly into the processor instruction cache. The bits are used as follows:

bit	use
1	enable (1)/disable (0) instruction cache
2	freeze cache (1)
3	clear entry (1)
4	clear cache (1)

The high nibble *e*, (bits 5-8) controls the CMB cache. Only bit 5 is used to enable (1) or disable (0) the cache.

For example, "cache 19" flushes the processor cache and enables the processor and external instruction cache.

CLEAR_ECC

This command zeros the single bit ECC counter tables for the system.

CMD_FILE

This command permits the user to build an ASCII command list for execution. No syntax or command validity checking is performed. The command list is ended by a blank command (ENTER alone on a command line).

CONSOLE *ddn*

This command changes the current console device to that specified by the parameter, where *dd* is a device mnemonic, and *n* is the device unit number. If the device specified does not exist, the console device remains unchanged. Legal devices and unit numbers are as follows:

Device	Mnemonic	Unit Numbers
SCC (CMB serial port)	SC	0,1
Four-way	FW	0-3, 10-13, 20-23, 30-33
Eight-way	EW	0-7, 10-17, 20-27, 30-37, 40-47, 50-57

COPYDISK

This command copies all files from disk 0 to disk 1. The copy destroys any data on the target disk, disk 1. The user is prompted to verify the action before copying begins.

COPYTAPE

This command is available on MAI 2000 systems only. The command destructively copies all files from diskette 0 to MCS tape. The user is prompted to verify the action before copying begins. The tape is first erased, and the volume label and boot program are written prior to copying files to the tape. When the copy finishes the diskette, the user is prompted for another diskette to append to the tape. If another diskette is desired, the user is prompted to replace the current diskette with the new one.

CRUNCH *n*

This command consolidates all unused disk space to provide a larger contiguous space for file storage. The parameter *n* indicates the drive unit number for crunch.

DCOPY *fff ggg*

This command copies one file, named *fff* from floppy disk 0 to floppy disk 1, naming the file *ggg*. If the destination file already exists, the user is prompted to verify overwriting the file.

DEBUG

This command immediately drops the user to the ROM based debugger. Refer to Section 8, MDS, for debugging commands.

DEVICE *ddn*

This command changes the current load device to that specified. In the parameter, *dd* is the two-character device mnemonic, and *n* is the device unit number. If the device specified does not exist, the load device remains unchanged. Legal devices and unit numbers are as follows:

Device	Mnemonic	Unit Numbers
SCC* (CMB serial port)	SC	0, 1
Four-way*	FW	0-3, 10-13, 20-23, 30-33
Eight-way	EW	0-7, 10-17, 20-27, 30-37, 40-47, 50-57
Floppy disk	FD	0, 1
Cartridge Streamer	CS	0
1/2 Inch Tape	TS	0
Winchester Disk	WD	0, 1, 10, 11

Devices marked with an asterisk (*) are download ports, and require host support.

DELETE *n fff*

This command marks the file named *fff* on disk *n* as deleted.

DINIT *n*

This command initializes the directory of disk *n*, and writes the bootstrap program to the disk. The user is prompted to verify disk deletion, and is asked for the new disk name.

DISASSEMB *m n*

This command is available on MAI 2500/3000/4000 systems only. The command disassembles and displays the mnemonics for the instructions from memory location *m* to memory location *n*. The display pauses after each screenful of data.

ECC_OUT

This command is available on MAI 2500/3000/4000 systems only. The command displays the single bit ECC count for each board and bank configured on the system. Memory ECC errors are only counted when the ECC option is selected (refer to the OPTION command in this section).

END

This command terminates building a Function Select test loop. Refer to Section 7 for instructions on building test loops.

EOT

This command repositions the tape to the logical end of media.

FORMAT *n* {*v*}

This command formats and verifies the floppy diskette specified by *n*. If the *v* parameter is non-zero, the disk is not verified following formatting. Service mode must be activated prior to executing this command (refer to the SERVICE command in this section).

On the MAI 2500/3000/4000 systems, the user must know the drive type of the disk being formatted. If the drive to be formatted has defects, use the DUTIL formatting utility, described in the system service manual and in the BOSS/IX Technical Reference Manual (M6227).

HELP

This command lists the executive commands on the console. The display pauses after each screenful.

HELP_WIZ

This command displays the Diagnostic Executive environment information on the console. This information includes the processor type, the amount of RAM available, the location of the executive in memory, the devices found in the system, and the test program currently loaded.

INIT

This command initializes the current test program. The total error count and pass number are also initialized, and all tests are enabled to run.

INSTALL {*fff*}

This command copies all files from the MCS or MTS boot device tape to the diagnostic partition of the fixed disk drive, destroying the existing files in the diagnostic directory. If the optional filename is specified, only that file is copied from tape to the disk.

ITERATION *n*

This command sets the iteration count of a test loop, overriding the default. This command is useful in obtaining a failure percent from a loop which ordinarily executes only a few times. Setting *n* to zero returns the iteration count to the default.

LDISK *n*

This command displays the directory of disk *n* on the console.

LIST_COMMANDS

This command lists the commands available in the currently loaded diagnostic program.

LIST_TESTS

This command lists all tests available in the currently loaded diagnostic program, regardless of their selection or manual intervention status.

LTAPE

This command lists the MCS tape directory on the console.

LOAD *n fff*

This command loads file *fff* from the current load device. If *n* is specified, the file is loaded from an alternate unit of the same device type. After loading, the program name and assembly time and date are displayed. If the file is a diagnostic program with an initialization routine, the routine is automatically run.

MTSCOPY

This command initializes the 1/2 inch MTS tape with a Volume Label and Loader. It then copies all diagnostic programs from disk 0 to the tape.

MTSLIST

This command lists the diagnostic programs on the MTS tape to the console. The assembly date and time and the revision are also displayed.

OPTIONS {*list*}

This command selects or deselects options, or displays the current options list. The list of options, *list*, consists of one or more of the options described below, separated by a space. If no parameters are specified, the currently selected options are displayed. If a parameter is preceded by a minus sign, "-", the option is deselected. Any option specified without the minus sign is selected. The options available are:

B14, B15	The meaning of these options is determined by the individual diagnostic. Refer to the descriptions of commands for the diagnostics later in this manual.
BELL	This option causes the terminal bell or beep to sound when an error occurs. This option is independent of the NO-PRINT option.
CLOCK	This option causes the system clock to be initialized and enabled. Each time the RUN command is executed, the elapsed time counter is reset. The elapsed time is displayed upon completion of each test pass.
ECC	This option enables logging of corrected single bit memory errors (MAI 2500/3000/4000 systems only).
HELP	This option displays a list of available options on the console.
LOOP	This option causes a test loop to return to the top of the loop upon occurrence of an error.
MANUAL	This option enables running of manual intervention tests. Manual intervention tests require operator input or other special conditions, such as installation of external loopback plugs.

NO_PRINT	This options suppresses display of individual error messages. The error count continues to be updated.
NONE	This option deselects all options. All other options included in the list are ignored.
NUMBER	This option displays the test number and title of each test when it is run.
PARITY	This option enables the detection and reporting of memory parity errors. The message displays the parity error register to aid in isolation of the faulty memory. (MAI 2000 systems only.)
PAUSE	This option causes the test to pause on each error. The user is given th option of continuing the test (SPACE), turning pause OFF and continuing (period, "."), or aborting the test (ESCAPE).
PERCENT	The percentage of errors versus passes through a test loop is printed after the loop is complete. If the LOOP option is set and an error occurs, the loop will not complete, but immediately loop back to the top.
REMOTE	This option is not supported for field and depot diagnostics. It enables remote diagnostics through a download port.
SP	This option causes all console output to be echoed to the serial printer. If an error is detected in the operation of the serial printer, the option is disabled.
STOP	This option causes test execution to suspend if an error occurs, and returns control to the executive. Program may be continued by using the RESUME command. STOP overrides the PAUSE command.
TEST_PCT	This option displays the percentage of errors versus the number of passes through the test loop when the test loop is complete.

PARSE *n*

This command determines the search order of the command tables. The default is to search executive command list first followed by the current test command list, is specified by a zero argument. A non-zero argument reverses the search order, so the test command list is searched first followed by the executive command list.

PMMU *n*

This command enables/disables the PMMU. A zero argument disables the PMMU, and a non-zero argument enables the PMMU. With the PMMU enabled makes address translation transparent for all memory addresses.

RELEASE

This command is used to build a diagnostics release, and so is for MBF use only.

RENAME *n fff ggg*

This command renames the file *fff* on disk *n* to *ggg*. Rename may also be used to change the volume name of the disk without initializing the disk.

RESET

This command executes a soft system reset. Following the reset, the vector table is initialized and the Diagnostic Executive resizes.

RESUME

This command continues execution of a test which has suspended on an error or been interrupted by ESCAPE. Other executive commands may be executed before the test is resumed.

RUN *x*

This command runs the current test through the number of passes specified by *x*. If the parameter is not specified, one pass is executed. If the parameter is zero, the tests will run continuously until interrupted by **ESCAPE**. The test order is determined by the selection order of the tests.

SAVE *name start end*

This command is not supported (MBF use only).

SBINIT *x*

This command initializes the super block of a formatted disk. All current O.S. partition information is deleted and a new diagnostic partition of the size specified by *x* is created. This command requires that SERVICE mode be enabled.

SERVICE

This command invokes service mode. The user is prompted for the password before service mode is activated. Service mode is required for Function Select tests and certain other commands and programs.

SHUTDOWN

This command invokes the ROM shutdown sequence, returning the system to the reboot prompt or menu. Execution of all programs is terminated.

TCOPY *fff*

This command copies the file *fff* from disk 0 to the MCS tape at the logical end of tape. The test prompts to user to specify whether to rewind and/or retension the tape prior to the copy.

TESTS {*list*}

This command selects or deselects tests to include in the test list, as specified in "list", for subsequent RUNs. Parameters are the test numbers to be selected/deselected. If no parameters are supplied, the currently selected tests are displayed on the console. If a test number or range is preceded by a minus sign, "-", the test or tests are deselected. If the minus sign is not used, the test or tests are added to the test list. A parameter of zero deselects all tests. Parameters are processed from left to right. For example:

tests 0,5	Selects only test 5
tests -4-7	Deselects tests 4 through 7
tests 1-6,-5	Selects tests 1 through 6, then deselects test 5

TINIT

This command initializes the MCS tape cartridge. The tape is erased and a new volume label and boot program are written on the tape, followed by a filemark. The user is prompted for a new volume name. All previous data on the tape is lost.

VERIFY *n*

This command verifies the format on disk *n*. All logical sectors of the disk are read in sequence.

WBOOT *n*

This command writes a new boot block to disk *n*. No other portion of the disk is modified.

WFD *n fff*

This command is not supported (MBF use only).

WHEREIS *ddn* {{,}ddn}

This command displays the physical hardware address of the controller card specified by *ddn*, where *dd* is the device mnemonic and *n* is the unit number. Any number of device identifiers may be included in parameter list, each separated by a space or a comma.

XCMD

This command causes the executive to begin parsing and executing commands that were entered using the CMD_FILE command.

XLOOP *x*{,*y*}

This command begins execution of the Function Select test loop *x* (1 or 2). If the second parameter is specified, the loop is executed y times before returning control to the executive. If the second parameter is not specified, the loop runs continuously until interrupted by **ESCAPE**.

4.4 ERROR REPORTING

Error messages displayed by the Executive are general in nature and usually pertain to an abnormal condition detected in a peripheral device. Specific error determination is handled by the diagnostic programs themselves.

4.4.1 Error Codes

The following error codes apply to disk and MTS errors detected by the Executive.

Code	Meaning
20	Timeout waiting for the bus to be free
21	Timeout waiting for command phase status after select
22	Timeout waiting for status or message phase
23	Skipped status phase
24	Timeout waiting for message phase

4.4.2 Disk Error Messages

- Seek Error

  A problem was detected seeking a particular sector (or track), indicating a bad or unformatted disk.

- Read Error

  A sector on the disk was not readable, indicating the data has been corrupted.

- Write Error

  A problem was detected writing a sector of data to the disk, indicating the disk is write protected.

- Directory Read Error

  The disk file directory was unreadable or has an incorrect format, indicating the disk is uninitialized or corrupted.

- Directory Write Error

  A problem was detected writing the disk file directory, indicating the disk is write protected or unformatted.

- File Already Exists

  An attempt was made to rename a file, where the new name already exists in the disk file directory. Use **LDISK** to view the directory.

- File Not Found

  The file requested does not exist in the disk file directory. Use LDISK to view the directory.

4.4.3 Tape Error Messages

- Tape Not Ready

  The tape was not initially ready prior to a tape operation, indicating the controller is "hung." Clear the condition with the **RESET** command.

- Tape Error

  An abnormal terminating status from a tape command was detected. Check to see if the tape is write protected. Use the **LTAPE** command to list the tape directory.

- Tape Not Initialized

  An attempt was made to read an uninitialized tape (no volume label or corrupted data). Use TINIT to initialize the tape.

4.4.4 Other Error Messages

- Checksum Error

  In the process of loading a file, an incorrect checksum was calculated.

- Invalid Hex Char

  In the process of loading a file, an incorrect hexadecimal character was read.

- Power Failure

  A system power failure was detected and reported via the non-maskable interrupt.

- Illegal EXEC I/O Call

  A user trapped to the executive service routine with an invalid service request.

SECTION 5
THE SYSTEM INTERACTION TEST

5.1 OVERVIEW

This section describes the System Interaction Test (SIT) diagnostic and provides instructions for its use and interpretation.

SIT is designed to detect interaction problems occurring between competing devices (e.g., Magnetic Cartridge Streamer), controller boards (e.g., LAN and 4-way controllers), and memory. If problems are found, SIT also provides some capabilities for isolating the source of the problem.

As an interaction test, SIT is not intended to identify faults in individual devices and controllers. However, failures of these devices and controllers will cause errors in SIT. If errors occur in SIT related to a device, the fault should be pursued further by running Logic Tests for that device (refer to Section 5).

When you load SIT, the program performs a process of "autosizing." During this procedure, the program checks to see which devices and controllers are installed on the system. It then defines a system configuration for those devices and controllers.

The test proceeds by exercising the devices and controllers in a semi-random fashion, attempting to create as many conflicts and worst-case situations as possible. The SIT Sequencer controls the order in which devices and controllers are exercised, insuring that processes are started sometimes in random order and sometimes in a determined order.

While the devices and controllers are being exercised, SIT keeps track of any problems that occur, such as multiple interrupts, surges on the power supply and overlapping memory accesses. Periodic displays provide a running report of the test results.

If a problem is determined to exist, a knowledgeable user can employ the SELECT command to specify individual boards or groups of boards for further testing. Running tests on specific boards makes it possible to narrow down the possible sources of the problem.

A number of further commands allow a more knowledgeable user to isolate the problem even further, to a portion of the fixed disk for example.

5.1.1 When to Use SIT

There are several situations in which it is useful to run the SIT diagnostic program.

1. You may wish to run SIT periodically to check the state of your system. Developing problems in the hardware can then be caught before they have a chance to cause serious damage and loss of data. This can be done periodically as part of a regular maintenance program.

2. If you are upgrading your operating system software, or re-installing the operating system, it is a good idea to run SIT before you do the installation. This will verify that the system hardware is in good working order prior to the installation.

3. If your system is having problems, the SIT is useful as a trouble-shooting tool. It can help you to eliminate possible causes and to isolate the source of the problem. If you are using the SIT as a diagnostic, you should leave the interpretation of the results to a trained service representative.

5.2 SIT CONFIGURATION REQUIREMENTS

5.2.1 Hardware Requirements

In order to run the SIT, the system must meet the following minimum configuration requirements:

1. A video display terminal configured on port 0. The terminal may be any of these types:

   - Ergonomic Video Display Terminal (EVDT)
   - Ergonomic Display Terminal (EDT)
   - 7270 VDT

2. Minimal RAM

   - 512 KB on MAI 2000 systems
   - 1 MB on MAI 2500/3000/4000 systems

5.2.2 Software Requirements

The only software requirement to use the System Interaction Test is the bootable Diagnostic Executive and the System Interaction Test.

5.3 STARTING AND RUNNING SIT

This subsection describes the procedures for using the System Interaction Test. These procedures include booting the diagnostic executive, loading the SIT program and executing commands.

5.3.1 Load Diagnostic Executive

SIT runs under control of the diagnostic Executive. Refer to Section 4 for instructions on performing an alternate load to the diagnostic Executive.

5.3.2 Start Printer Output

Diagnostics messages can be output to a parallel printer or to a serial printer with either an industry standard or Basic Four interface. The printer must be configured as the system printer. The messages are printed in addition to being displayed on the terminal screen.

To print messages to the serial printer, at the <exec> prompt, enter:

option sp

For the parallel printer, enter:

option lp

Make sure the printer is on-line. Now proceed with the tests.

5.3.3 Loading SIT

Once the diagnostics executive has been loaded, you are ready to load the SIT program. At the diagnostics executive prompt type:

LOAD SIT

in either upper or lower case, and then press **ENTER**.

The diagnostics executive then begins loading the SIT program into memory. While it is loading the program, it displays several rows of dots on the screen. When it is finished loading, it displays:

SYSTEM INTERACTION TEST (SIT) Rev. XX, wait { = 30 seconds for auto sizing

The SIT program is now operating. It begins by "auto sizing", during which it searches for the devices that are ready to be used during the test. It then builds a configuration file for those devices for use during the test.

Note that SIT will configure a floppy drive for testing only if there is a diskette in the drive and the lock button is pressed in. Similarly, it will configure the MCS or MTS drive for testing only if there is a cartridge or tape installed and not write-protected.

When SIT is finished auto sizing, it displays a report of the controllers and devices present and the status of each, as in Figure 5-1.

```
SYSTEM CONFIGURATION TABLE
cont  status   device  status    properties
lnlx  present  ln10    runnable  1 megahertz CSMA Local Area Network
fw0x  present  fw00    runnable  serial comm port-chan a
fw0x  present  fw01    runnable  serial comm port-chan b
fw0x  present  fw02    runnable  serial comm port-chan c
fw0x  present  fw03    runnable  serial comm port-chan d
fw1x  present  fw10    runnable  serial comm port-chan a
fwlx  present  fw11    runnable  serial comm port-chan b
fw1x  present  fw12    runnable  serial comm port-chan c
fw1x  present  fw13    runnable  serial comm port-chan d
mc0x  present  mc00    cant run  streamer cartridge tape
wd0x  present  wd00    runnable  Hard Disk, #cyl=830, #heads=6
fd0x  present  fd00    runnable  double sided, double density, 5 1/4
fd0x  present  fd01    cant run  double sided, double density, 5 1/4
sccx  present  scc0    runnable  serial communications port
sccx  present  scc1    runnable  serial communications port
MEMORY MAP:  SIT area=000000-034800,freespace=034800-0BFFF
```

Figure 5-1. SIT Configuration Report

In the figure, the column labeled "cont" lists the controller boards, the column labeled "device" lists the devices. The kind of board is indicated by a two or three letter designation. The number of the board is indicated by the first digit, and the number of the device on that board is indicated by the second digit. E.g., "ln10" indicates the Local Area Networking controller board number l as the controller, and port 0 as the device, "fw12" is the four-way controller board number 1, and port number 2. The exceptions to this are the sccx ports, which reside on the Central Microprocessor Board (CMB).

Note that controller boards and devices are numbered 0,1,2,..., so that "0" is the first, "1" is the second, etc.

If the status of a device is "cant run", the device was found but not ready for use, as in the case of a tape drive that did not have a tape properly in place.

Below the configuration table, the memory addresses are displayed where the SIT program is contained and the memory addresses that are free.

5.3.4 Using SIT Commands

Using SIT as a basic diagnostic is a simple matter of selecting tasks and running them. When SIT first starts, all tasks are selected by default. To run all tests, enter the RUN command. Later, you can stop tests to narrow down the possible source of the system's problems.

A knowledgeable user can use SIT's commands and options to place special conditions on certain tasks. To display a list of the available commands, use the LIST command at the SIT prompt. LIST is a diagnostic Executive command available in SIT.

To execute a command, type the name of the command at the SIT prompt, together with a task name or option as required, and press **ENTER**. The commands may be entered in either upper or lower case.

5.4 SIT TASK DESCRIPTIONS

This section describes the tasks run by SIT. The variables "x" and "y" are used in task names to indicate changing values.

5.4.1 LAN Task

The **LNx0** task exercises the Local Area Networking controller board and port. "x" is the number (0 or 1) of the controller board. There is only one port per board.

The task broadcasts a message between 1 byte and 1024 characters in length for each pass. This is a "write only" test, since no return message can be expected. The LAN cable does not need to be connected for this test.

5.4.2 Four-way Task

The **FWxy** task exercises the four-way serial communications controllers and their ports. "x" is the number of the controller (0-5) and "y" is the number (0-3) of the port.

The task causes four-way controller x to transmit a message to port y. The message is a multiple of 80 characters long, between 80 and 1920.

If a CRT is attached to the port, the port is configured for 9600 baud transmission and no loopback occurs. If an external loopback plug is installed in the port, the data is looped back. If there is no attachment to the port, the port is configured for 1200 baud and internal loopback. Internal loopback data is compared to the transmitted patterns to verify system integrity.

5.4.3 Eight-way Task

The **EWxy** task exercises the eight-way serial communications controllers and their ports. "x" is the number of the controller (0-5) and "y" is the number (0-7) or the port.

One pass of the task causes the controller to transmit a message of 80 to 1920 characters to the port.

If a CRT is attached to the port, the port is configured for 9600 baud transmission and no loopback occurs. If an external loopback plug is installed in the port, the data is looped back. If there is no attachment to the port, the port is configured for 1200 baud and internal loopback. Internal loopback data is compared to the transmitted patterns to verify system integrity.

5.4.4 MCS Task

The **MC00** task exercises the MCS controller board and its port. Only one MCS is supported.

This is a read/write test. One pass writes between 1 and 17 blocks (512 Bytes), reads back what was written, and then compares the messages. Each successive pass skips to the end of data prior to each write until all of the tape is used, preventing excessive wear on any specific area of the tape.

The sequencer may set a flag for a pass that causes the drive motor to be turned on and off for each access. This causes a "worst case" drain on the power supply.

5.4.5 MTS Task

The **MT00** task exercises the MTS controller and drive. Only one controller/drive pair is supported.

One pass of the MTS task writes, reads and compares 1 to 17 blocks (512 bytes per block). The writes and reads are performed in a progressive manner. Each successive pass skips to the end of data prior to each write until all of the tape is used, preventing excessive wear on any specific area of the tape.

5.4.6 Floppy Drive Task

The **FD0y** task exercises the floppy controller and drive (available on the MAI 2000 only). "y" is the number of the drive (0 or 1).

This is a write, read and compare test, and can be run only if a diskette is in the drive. Errors will occur if the diskette is write protected or if the diskette is damaged, as well as if there is a problem with the controller or drive.

One pass causes one block (512 Bytes) to be written, read back, and the message compared. The drive motor is periodically turned off and back on (approximately one out of ten accesses) causing a worst case drain on the power supply.

5.4.7 Winchester Disk task

The **WDxy** task exercises the specified fixed disk controller and drive. "x" is the controller number (only 0 is supported), and "y" is the number of the drive (0 or 1).

One pass of the task writes, reads and compares 1 to 17 blocks (512 bytes).

If the KILL option is not used with the RUN command, access is restricted to the diagnostic cylinder, so the test does not destroy disk data. If the KILL option is used with the RUN command, the sector addresses are selected from all logical disk space past sector 5 (the superblock).

5.4.8 Memory Task

The **MEMM** task exercises the system RAM (random access memory). The area of memory from the end of SIT's program and data areas to the top of RAM are tested.

This is a read/write test. One pass involves two write, read and compare cycles. The first cycle writes the word addresses for each word of memory to the address, reads it back and compares the two. The second cycle writes, reads, and compares the complement of the word address to the address.

5.4.9 Floating Point Task

The **FPCP** task is exercises the floating point processor, if it is detected during auto-sizing. This task is not available on the MAI 2000.

This task does not start automatically, but must be specifically selected using the START command. Each pass of the task computes and compares e^pi, sqrt 2 and cosh^2 - sinh^2 for random values.

5.4.10 Serial Communications Controller Task

The SC0x exercises the SCC chip on the CMB. "x" is either 0 or 1, indicating CMB serial port 0 or 1.

The task configures the SCC chip for 1200 baud internal loopback transmission. Internal loopback data is compared to the transmitted patterns to verify system integrity.

5.5 SIT COMMANDS

The SIT commands are listed in two sets, the Basic Commands and the Advanced Commands.

5.5.1 Basic Commands

RUN

Format: run {{-}kill}

The **RUN** command starts all currently selected tasks. The user control task is then put to sleep, and error and news reporting begins.

When SIT initially starts, all tasks are selected that were found to be runnable during the autosize, except the floating point task. Use the SELECT command to add and remove tasks from the selected list. The user control task is then put to sleep, and error and news reporting begins.

The **KILL** option begins write access to the user data area of the disk, and so is destructive to data and programs on the disk. The **-KILL** option restricts the task to accessing the diagnostics cylinder.

CAUTION

Using the "RUN" command with the "KILL" option destroys any data on the disk.

SELECT

Format: select {all | none | show | {-}*taskname*}

The **SELECT** command specifies the tasks to be started by the "RUN" command.

If SELECT is entered without options, you are prompted for each task corresponding to a device found during autosizing. For instance:

fw03 (n=no,<cr>=yes)?

To include a task, press **ENTER** alone. To exclude a task, type **N** and press **ENTER**.

The ALL option selects all tasks, and the NONE option deselects all tasks.

The "*taskname*" option selects the specified task, and the "-*taskname*" deselects the task. Only one task may be selected or deselected per command line.

The SHOW option displays a list of the currently selected tasks.

SLEEP

Format: sleep

The **SLEEP** command releases the console for use by other tasks, while errors and news updates are reported. SLEEP is automatically invoked by the RUN command.

To execute further commands, SLEEP must be interrupted. To interrupt SLEEP, press the **ESCAPE** key.

START

Format: start *taskname*

The **START** command begins execution of the specified task. Only one task may be started per command line.

START begins execution of the task in addition to those already running. To begin new updates and error message displays for the task, use the SLEEP command.

STOP

Format: stop *taskname*

The **STOP** command ends execution of the task specified by task. It is used to stop one task while continuing execution of others. Only one task may be specified per command line.

TASKS

Format: tasks

The **TASKS** command displays a list of the tasks corresponding to devices that were found during autosizing. These are the tasks that are available for selection with the SELECT command.

SHUTDOWN

Format: shutdown

The **SHUTDOWN** command shuts down the system from SIT. This is the only method available for exiting SIT. When the system is shutdown, the re-boot prompt or menu is displayed:

In order to return from SIT to the Diagnostic Executive or the operating system, it is necessary to shut down the system and then reboot.

5.5.2 Advanced Commands

BFERR

Format: bferr {off | *taskname*}

The **BFERR** command causes a break if an error is detected during a comparison of read and write buffers. If the OFF option is specified, no break occurs.

If the optional taskname is not specified, any data miscompare between read and write buffers causes a break. If a taskname is specified, the break occurs only on a miscompare of the read and write buffers for that task.

When a break occurs, the screen displays the prompt:

<bferr>

At this prompt, the user must enter one of the following:

- c - to continue testing
- d - to dump the read/write buffers to the screen.

CYLINDER

Format: cylinder *taskname lowerbound* {*upperbound* {A}}

The **CYLINDER** command causes the specified task to access the specified cylinder or cylinders only, overriding the sequencer.

Lowerbound and upperbound may be any decimal number, and specify entire cylinders on the disk (they are not checked for legal values).

If only the lowerbound is specified, only that cylinder is exercised by the task. If both the lowerbound and the upperbound are specified without the A option, then the inclusive range of cylinders are tested. Adding the A option causes the task to alternate between the two cylinders.

DEBUG

Format: debug

The **DEBUG** command is used to enter the ROM debugger (refer to MDS, Section 7).

DELAY

Format: delay {*taskname*} *value*

where "value" is any integer greater than or equal to zero.

By default, a task is made available to the sequencer for a new pass as soon as it has completed the previous pass. The **DELAY** command causes a delay of the value times 100 milliseconds before the specified task is made available for another pass. If no task is specified, the delay applies to all tasks.

DLTH

Format: dlth *taskname length*

The **DLTH** command specifies the data length to be used by the specified task for all reads and writes. Normally the sequencer varies the data length. Tasks that use data lengths as multiples of a constant use dlth as the constant.

The data length is given in hexadecimal, specifying the number of bytes, calculated by:

data length = (int(length/c) + 1)*c

DPAT

Format: dpat *taskname pattern*

The **DPAT** command specifies the data pattern used by the specified task.

The data pattern is a four byte pattern in hexadecimal notation. If the pattern entered is less than four bytes, the pattern is filled with hex nulls (hex 00). All data written by the task then consists of repetitions of this 32-bit pattern.

HEAD

Format: head *taskname lowerbound* {*upperbound* {A}}

The **HEAD** command specifies the read/write head or heads to be used in the Winchester or floppy drive task specified. The head numbers are given in decimal notation.

If only the lowerbound is specified, only that head is used. If both the upper and lower bounds are specified without the A, then the inclusive range of heads is used. If the A option is used, then the task alternates between the two heads.

The lower and upper bounds are not checked for being in the legal range.

INIT

Format: init

The **INIT** command reinitializes the SIT news update counts.

NEWS

Format: news *seconds*

The **NEWS** command specifies the amount of time between news update status messages in seconds. The time is specified in hexadecimal notation.

OPTION

Format: option {-}{stop | no-print | bell | lp | slave | pause | none | show}

The **OPTION** command conditions SIT execution with the following options:

- Stop - SIT execution stops on occurrence of an error.
- No_print - News output to the printer is stopped.
- Bell - The bell is sounded when an error occurs.
- Lp - News output is sent to the parallel printer.
- Slave - News output is sent to the terminal slave printer.
- Pause - SIT execution pauses on occurrence of an error.
- None - All options are cleared.

Preceding an option with a dash, "-", deselects that option.

PRIOR

Format: prior *taskname level*

The **PRIOR** command assigns the priority level of the specified task. The larger the number assigned as a task's level, the higher the probability that the task will be started before any other.

RANDOM

Format: random {*seed*}

The **RANDOM** command instructs the sequencer to send random parameters to the tasks. The seed is the value used to initialize the random number generator.

SAGE

Format: sage

The **SAGE** command invokes the SIT system debugger. SAGE requires detailed knowledge of the internal structure and operation of SIT, and so is of little values for system diagnostic purposes. To return to SIT, enter **GO**.

SECTOR

Format: sector *taskname lowerbound* {*upperbound* {A}}

The **SECTOR** command specifies the sector or sectors to be tested by the specified floppy or fixed disk task. The upper and lower bounds are specified in decimal notation.

If only the upper bound is specified, only that sector is tested. If the upper and lower bounds are specified without the A option, then the inclusive range of sectors from the lower to the upper bound are tested. If the A option is specified, the task will alternate between the two sectors.

SECTION 6
LOGIC TESTS

6.1 OVERVIEW

This section describes the procedures for running the Logic Tests and interpreting their results.

Logic tests provide the ability to exercise individual subsystems, such as controller and drive pairs or memory, checking for faults at that level. Logic tests provide diagnostic capabilities one step deeper than does the System Interaction Test, to the board or subsystem level. Further isolation to the component (chip) level is provided by the Function Select Tests (Section 6).

Logic test are typically hierarchical, beginning with low level tests, such as reading and writing registers, and building to tests of the entire subsystem. The object is to determine if the unit can operate in an operating system environment. Faults are isolated by interpreting the results as to which tests pass and which fail.

The logic tests can be booted from the winchester disk, if they have been installed on the diagnostics partition, from MCS or MTS tape or from floppy diskette (floppies are supported on the MAI 2000 only). The MCS, MTS and floppy based diagnostics are on media labeled "DIA".

6.2 GENERAL RUN PROCEDURES

The logic tests are all run under the control of the diagnostic executive program. Depending on the system configuration, the diagnostic executive is booted from fixed disk, MCS, MTS or floppy diskette. Regardless of the medium used, the diagnostics are accessed by performing an alternate system load. The specific alternate load procedure varies depending on the type of system.

6.2.1 Load Diagnostic Executive

All of the Logic tests run under the control of the diagnostic Executive. The executive is started by booting to the diagnostic medium. Refer to Section 4 for instructions on booting the diagnostics and using the Executive.

6.2.2 Start Printer Output

Diagnostics messages can be output to a parallel printer or to a serial printer with an industry standard or Basic Four interface. The printer must be configured as the system printer in NVRAM (refer to section 8.5.4). The messages are printed in addition to being displayed on the terminal screen.

To print messages to the serial printer, at the <exec> prompt, enter:

```
option sp
```

For the parallel printer, enter:

```
option lp
```

Make sure the printer is on-line. Now proceed with the tests.

6.2.3 Running the Logic Tests

Once you have loaded the diagnostic executive, you are ready to run the Logic Tests. The logic tests can be either individually loaded and run, or run automatically by the automatic test mode. For each logic test, you can also select individual subtests. All test selection and execution is performed by command.

6.2.3.1 RUN LOGIC TESTS IN AUTO MODE

To run the logic tests in automatic mode, enter at the diagnostics executive prompt:

```
auto
```

The program displays the following warning and asks you to verify that you want to continue:

```
WARNING: ALL NON-WRITE-PROTECTED MEDIA IS SUBJECT TO CORRUPTION (WINCHESTER DISK EXEMPT)...Continue (y/n)?
```

The tests perform read/write testing on MCS, MTS and floppy diskette. If you booted from MCS, MTS or floppy, make sure diagnostic medium is write-protected. Make sure that any other tape or floppy is scratch. The write tests do destroy data on these media.

Type **Y** and press **ENTER** to begin running the Logic Tests. No further intervention is required until the tests are complete. Note that the memory test lasts a long time, varying with the amount of memory, with no apparent activity.

At each step of its execution, the program displays a message indicating what it is doing. First it loads a particular logic test, then runs it. Each logic test is made up of several sub-tests. As each sub-test is run, its number and a brief description is displayed. When a logic test has been completed, a message indicating the number of errors that occurred is displayed. The next test is then loaded and run, and so on until all tests are finished.

The last test to run is the fixed disk logic test. When the logic tests are finished, the prompt for this test is displayed:

<disk>

If you started printing of the test results (Section 6.2.3), the tests do need to be monitored; otherwise you should watch and note any logic test and sub-test that produces an error.

Some apparent errors only indicate that the MCS cartridge or floppy diskette is write-protected, and these errors can be ignored. Similarly for the fixed disk test, some sub-tests are not run.

6.2.3.2 RUNNING INDIVIDUAL LOGIC TESTS

Individual logic tests may be loaded and run by test name. The test is loaded by specifying the test name following the "load" command at the diagnostic executive prompt or at the prompt for any other logic test. The test names are listed in Table 6-1.

Table 6-1. Logic Test Names

Name	Description
DISK	Winchester Disk (6.3.1)
MCS	Magnetic Cartridge Streamer (6.3.2)
MTS	Magnetic Tape Streamer (Reel) (6.3.3)
PIT	Parallel Interface/Timer (PI/T) (6.3.4)
SCC	CMB Serial Communications Controller (6.3.5)
FWAY	4-Way (6.3.6)
EWAY	8-Way (6.3.7)
LAN	Local Area Network (6.3.8)
MEMORY	System Memory (6.3.9)
PORTS	Serial Ports (6.3.10)
FLOPPY	MAI 2000 Floppy Diskette (6.3.11)
IOTST	MAI 2000 I/O Bus (6.3.12)
RTC	MAI 3000 Real Time Clock (6.3.13)
MMU	MAI 3000 Memory Management Unit (6.3.14)
CMB	MAI 3000/4000 CMB (6.3.15)
CACHE	MAI 3000/4000 Cache (6.3.16)
EIAB	MAI 4000 Expansion Interface (6.3.17)

For example, to begin the Winchester Disk logic test, enter:

```
load disk
```

The prompt for this logic test is then displayed, and it is ready to run. To run the test, use the "run" command:

```
run
```

6.2.3.3 SELECTING AND RUNNING SUBTESTS

Each logic test consists of a set of subtests, which are identified by number. When a logic test is first loaded (as described in 6.2.4.2), all subtests are initially selected, and would be run if the **RUN** command is executed. The set of tests can be changed using the **TEST** command which selects and deselects tests

If the command is executed without parameters, the currently selected tests are listed.

The command takes parameters consisting of the numbers of tests to be selected or deselected. A sequence of test numbers may be specified by using a hyphen between two numbers. A hyphen preceding a single number or a range specification deselects the specified tests. A zero (0) deselects all tests. Commas separate test and range specifications. For example:

tests 0,5	Deselects all tests, then selects test 5
tests -4-7	Deselects tests 4 through 7
tests 1-6,-5	Selects tests 1 through 6, then deselects test 5

6.2.3.4 SHUTDOWN FROM DIAGNOSTICS EXECUTIVE

To return the system to normal operation, you must shut down the system and then perform a normal boot.

To shut down the system from the diagnostic executive prompt or the prompt for any logic test, enter:

shutdown

The usual reboot prompt or menu is then displayed.

6.2.3.5 EXECUTIVE COMMANDS FOR LOGIC TESTS

There are several commands available through the diagnostic executive. The following commands are relevant to running the Logic Tests.

ALL

This command runs all logic tests in a predefined order. User intervention tests are run only if the OPTIONS MANUAL command has been executed.

AUTO

This command runs all logic tests, except those requiring user intervention (manual tests).

INIT

This command initializes the current test program. It also initializes the total error count, the pass number, and enables all tests to be run.

ITERATION *n*

This command specifies the number of times a subtest is looped through during a test run, overriding any default number of loops.

LIST_COMMANDS

This command lists the commands available in the currently loaded logic test.

LIST_TESTS

This command lists the subtests by number and name of the currently loaded logic test.

LOAD *fff*

This command loads the logic test *fff*.

OPTIONS {*list*}

This command selects/deselects options or displays the current options. If no parameters are listed, the currently selected options are displayed. Preceding an option with the minus sign (-) deselects the option. The options are:

B14, B15	Meanings differ for each logic test. Refer to the test descriptions in the following subsections.
BELL	Sounds a bell or beep when an error occurs.
CLOCK	Initializes and enables the system clock, and resets the elapsed time counter. When enabled, the elapsed time is displayed upon completion of each test pass.
ECC	Enables logging of corrected single-bit memory errors (MAI 2500/3000/4000 only).
HELP	Lists options on the screen.
LOOP	The test loops back to the beginning when an error occurs.
LP	All output to the terminal is also echoed to the parallel printer.
MANUAL	Enables the running of user intervention by the **ALL** and **RUN** commands.
NO_PRINT	Suppresses the display of individual error messages, but the error total is still updated.
NONE	Deselects all options.
NUMBER	Displays the test number and title when it is run.
PARITY	Enables the detection and reporting of memory parity errors. The reporting message displays the parity error register to help identify the failing memory.
PAUSE	Pauses the test after each error, giving the user the option to continue the test (press SPACE), turn off pause and continue (press "."), or ending the test (press ESCAPE).
PERCENT	Prints the percentage of errors versus passes through a test loop when each loop is completed. Note that if the LOOP option is set and an error occurs, the loop does not complete.

REMOTE	Permits the diagnostics to be controlled via a remote connection. Remote Diagnostics are not supported for field use on BOSS/IX systems.
SP	All output to the terminal is also echoed to the serial printer. The option is disabled if an error is detected in the operation of the printer.
STOP	Stops the test if an error occurs, returning control to the diagnostic executive.
TEST_PCT	Displays the percentage of errors versus passes when the test pass (all loops) is complete.

RESUME

This command continues execution of a test after an error if it has been stopped by the error (**STOP** option) or by **ESCAPE**.

RUN *x*

This command runs the currently selected subtests through the number of passes specified by *x*. If the parameter is not specified, one pass is executed. If the parameter is zero, the tests will run continuously until interrupted by **ESCAPE**. The test order is determined by the selection order of the tests.

SHUTDOWN

This command shuts down the system, returning to the reboot prompt or menu.

TESTS {*list*}

This command selects and deselects subtests, and displays a list of the currently selected subtests. The selected tests are run by succeeding **RUN** commands. If no parameter is supplied, the currently selected tests are displayed. Parameters are the test numbers to be selected or deselected. A minus sign ("-") preceding a number deselects that test. A dash between two numbers specifies a range of tests to select/deselect. A zero (0) deselects all tests. For example:

tests 0,5	Deselects all tests, then selects test 5
tests -4-7	Deselects tests 4 through 7
tests 1-6,-5	Selects tests 1 through 6, then deselects test 5

6.2.3.6 MANUAL INTERVENTION TESTS

Some tests used by the Logic tests require manual intervention. The tests are not run unless the **OPTIONS MANUAL** command has been issued. Once manual intervention tests are enabled, they are included in the tests run by **ALL**, and may be selected and run by **RUN**.

Before a manual intervention test is run, it prompts the user for verification. This is usually required only when the test is destructive of data (e.g., disk writes) or requires special equipment (e.g., external loopback plugs). The test descriptions indicate whether the test requires manual intervention.

6.3 LOGIC TEST DESCRIPTIONS

6.3.1 Winchester Disk Logic Test

The Winchester Disk Logic Test tests the fixed disk subsystem, including the controllers and disk drives. The test is intended to isolate faults to the replaceable unit.

It is beyond the capability of the hardware to test the entire subsystem. Fault isolation can be enhanced by using a few spare items, such as cables, disk drive and controller.

The test requires at least one disk drive for a complete test. With only one disk drive, the other port is not tested. It is best to have two disk drives, and for both drives to be the same type. The disk drives must be formatted in the usual BOSS/IX format. (Refer to the BOSS/IX Reference manual or the system service manual for formatting instructions.)

The disk subsystem is "autosized" by Test 14, which determines the number of disks and reads the Superblock for configuration data. If the autosize fails, subsequent tests requiring that information will also fail.

The diagnostic has a write protection feature, allowing it to be used at a customer site where data loss is not acceptable. The diagnostic will not write on the disk unless the **WRITEOK** command is executed. Tests that write to the disk are not run unless **WRITEOK** has been executed.

The full diagnostic is extensive and runs over an extended period of time. For a briefer test of the disk subsystem, the **QUICK** command can be used. The quick test is only intended for verifying that the system works, such as after a faulty component has been swapped out. It is not intended for fault isolation.

The tests are grouped as follows:

1-10	Test controller registers.
11-15	Test basic controller board functions to verify that the adapter to controller interface is working.
16-25	Test basic operations for higher level operations. Require that a disk drive is attached.
26-31	Seek tests, verifying that the subsystem can correctly locate cylinders on the drives
32-40	Verify correct operation of data transfer controller commands.

6.3.1.1 LOADING AND RUNNING THE WINCHESTER LOGIC TEST

The Winchester Logic Test is loaded and run in the usual manner for logic tests. At either the diagnostic executive prompt, or at the prompt for another logic test, enter:

```
load disk
```

To run the logic tests in normal mode, enter:

```
run
```

Refer to the command descriptions in Section 6.3.1.3 for additional test run options.

6.3.1.2 WINCHESTER SUBSYSTEM TEST DESCRIPTIONS

6.3.1.2.1 Test 001 - Addressability

This test checks the basic addressing of the adapter registers. It reads read-only registers, and writes and reads the others. The test loops 128 times by default.

Registers CC000C through CC0000 are accessed in descending order.

6.3.1.2.2 Test 002 - Write-Read Control Reg

This test writes data to the control register, then reads it back and compares it with what was originally written. Only the 4 least significant bits are used. The test is performed for 70 data patterns, and normally loops 128 times.

Interpretation: This test may fail if test 1 fails.

6.3.1.2.3 Test 003 - Data Reg Loop Thru

This test identifies failures in the data bus using all 8 bits. 70 data patterns are written, read back and compared. The loop is normally repeated 128 times.

Interpretation: This test may fail if tests 1 or 2 fail.

6.3.1.2.4 Test 004 - Vector Reg Loop Thru

This test checks the vector register and the 8-bit bus. 70 data patterns are written, read back and compared. The loop is normally repeated 128 times.

Interpretation: This test may fail if tests 1-3 fail.

6.3.1.2.5 Test 005 - Null Status

This test checks that the status register is equal to 00 after the reset line is activated, and that the reset line status bit functions. The test loop is normally performed 25 times.

Interpretation: The test may fail if tests 1-4 fail. Failure of this test alone indicates that the reset line or the status bit have failed.

6.3.1.2.6 Test 006 - Test Mode DMA

This test checks the DMA logic without involving the controller or disk drive.

This test will abort if errors occurred in tests 1-5.

Interpretation: The test usually fails if an error occurred in tests 1-3 or 5. Passing tests 1-5 and failing 6 indicates DMA logic failure.

6.3.1.2.7 Test 007 - Bus Error

This test forces a DMA bus error, to ensure that the status is correctly indicated. It also tests that the bus error clear command resets the bus error status bit.

The test loop is normally performed 128 times.

This test will abort if errors occurred in tests 1-5.

Interpretation: This test depends on test 6 to pass in order for the bus error status bit to set. Passing tests 1-6 and failing 7 indicates a failure in the bus error status detection logic, or of the DMA register itself.

6.3.1.2.8 Test 008 - DMA Register

This test uses all available memory to test the ability of DMA logic to use all DMA register bits. Two loops are performed.

This test aborts if test 6 fails.

Interpretation: If tests 1-3 and 6 pass, and this test fails, the fault is the DMA register. This test will fail if tests 1-3 fail.

6.3.1.2.9 Test 009 - Conditional Interrupt

This test forces a bus error, first with interrupts disabled to verify that the disable function works, then with interrupts enabled to verify that the interrupt is generated. It also verifies that no interrupt is generated simply by enabling interrupts. A vector value of 40 hex is used. Fifty loops are normally performed.

Interpretation: If tests 6 and 7 pass and this test fails, the error is in the vectored interrupt generation logic. The test will fail if 6 or 7 fail.

6.3.1.2.10 Test 010 - Bus Error Interrupt

This test is similar to test 9, except that all vector values are used, from 40 to FF hex. The test normally loops 20 times.

This test aborts is tests 1-9 fail.

Interpretation: If test 9 fails, 10 will fail. If test 9 passes and 10 fails, the fault is at a vectored interrupt other than 40 hex. If test 4 passed, the error is not due to the vector register itself, but due to invoking the interrupt.

6.3.1.2.11 Test 011 - Select Status

This test checks the adapter indicates the correct status from a SASI controller select and a reset.

This test aborts is any of tests 1-9 fail.

Interpretation: If tests 1-9 pass and 11 fails, the failure may be in the SASI interface logic, the cables or the controller. The returned status may indicate whether a single interface has failed, or all have failed.

6.3.1.2.12 Test 012 - Handshake

Test 12 is an extension of test 11. This test checks that the adapter indicates the various phases, that the interface signals are dynamic (not stuck on or off), and that the complete handshake routine is working. The test loop is normally performed 50 times.

The test aborts if any of tests 1-5 fail.

Interpretation: If test 11 passes and 12 fails, the failure is probably in the controller. Passing test 12 indicates that the SASI interface logic on the adapter is good (only DMA and interrupt data transfers remain to be tested on the adapter).

6.3.1.2.13 Test 013 - Sense Command

This test checks the use of the SENSE command by the diagnostic driver. The test issues the SENSE command, and tests for the standard handshake operation. The tests normally loops 128 times.

The test is performed on one unit only, using the lowest number unit (0) unless changed by the UNIT command.

This test aborts if any of tests 1-5 fail.

Interpretation: If test 12 passes and 13 fails, the remaining tests will probably fail. The failure is probably due to tight timing in the handshake, or an intermittent failure detected only by this test.

6.3.1.2.14 Test 014 - Unit Status (Drive Auto Configure)

This test checks the controller's ability to execute the Test Unit Ready command without errors, and to determine the number of disk drives attached. The command is issued to unit 0 and then to unit one, keeping track of how many units respond without errors. The test normally loops 128 times.

This test is used for autosizing in preparation for later tests. Failure of this test may cause later tests to abort.

This test will abort if tests 1-5 fail.

Interpretation: Failure of test 14 alone, with a disk drive attached, indicates a faulty disk drive, or a disk drive interface, cable or controller failure. Additional testing is required to isolate the fault. Test 14 passing indicates that the controller can pass ending status correctly, and that basic handshake is working for units.

6.3.1.2.15 Test 015 - Abnormal Termination

This test checks the controller's ability to detect an invalid command argument. This is done using the Test Unit Ready command with reserved bytes equal to FF hex and a unit number of 7, and a Read command with an illegal logical block of 1FFFFF hex. The test normally loops 128 times.

This test aborts if any of tests 1-5 failed.

Interpretation: This test will fail if either tests 12 or 13 failed. If 15 fails alone, the fault is probably in the controller. The message indicates the exact command that failed. Passing test 15 indicates that the controller can detect a bad DCB and that basic adapter-controller communication is working.

6.3.1.2.16 Test 016 - Recalibrate

This test checks the ability of the Recalibrate command to recalibrate the drive heads. The command is issued, and the controller is checked for correct phases, and that the controller returns command completion status within the allotted time. The test is performed on all drives optioned to run.

This test requires the entire winchester subsystem.

The test aborts if tests 1-5 or 14 failed, or if no drives are attached.

Interpretation: Failure of this test normally indicates a drive failure, if previous tests passed. Drive ST506 interface failures may trigger controller detected errors. Use the controller error code to interpret the error.

6.3.1.2.17 Test 017 - Write WDC RAM

This test checks that the Write Buffer Command works. 1024 bytes are written to the controller sector buffer without using DMA.

The entire winchester subsystem is required for this test.

The test aborts if errors occurred in tests 1-5 or 11-15.

Interpretation: If this test passes, only command completion interrupts and read data transfer in DMA mode need testing.

6.3.1.2.18 Test 018 - Write-Read WDC RAM

This test checks the ability of the write sector buffer to transfer and store data in the buffer RAM. It does this by reading the data back with the Read Sector Buffer command and comparing the data. The test normally runs 128 loops.

The test aborts if errors occurred in tests 1-5 or 11-14.

Interpretation: If test 17 passes and 18 fails, the fault is in either the write data or read transfer, as indicated by the command code returned. If the data doesn't compare, check for apparently dropped bits. If tests 1-5 pass, the error may be due to the data buffers from the SASI interface, or gating control logic. If the commands are successful except for a controller detected error, the error is in the controller.

6.3.1.2.19 Test 019 - Translate

This test checks the Translate command for successful completion, in preparation for tests 22.

This test requires the entire winchester subsystem.

The test aborts if there were errors in tests 11-15.

Interpretation: If test 18 passes and 19 fails, either there is a problem with the translate command, or possibly the drive is incorrectly formatted. The test will probably fail with a position error if test 18 failed.

6.3.1.2.20 Test 020 - Read Disk

This test reads a track worth of information without involving DMA logic or interrupts. It is also a pretest for tests 21 and 37.

This test requires the entire winchester subsystem.

Interpretation: Compare the results of test 20 with those of tests 17 and 18 to determine if the failure is associated with the read command alone, or with the data transfer from the controller. If tests 17 and 18 pass and only 20 fails, the fault is probably caused by the disk drive.

6.3.1.2.21 Test 021 - Configuration Read

This test checks that the controller can access the superblock, and that the superblock is intact and can be read to determine the disk type. Unlike test 20, this test interprets the data to see if it is valid.

The test aborts if errors occurred in tests 11-15 or 20.

Interpretation: If the superblock is bad, display the data from sector 4 using the RDISPLAY command (refer to section 6.3.1.3 for command arguments). The first long word should be ".SB." in ASCII. If this is not correct, the disk must be reformatted. (Refer to the BOSS/IX Technical Reference Manual (M6227), or the system service manual, for instructions on formatting the disk.)

6.3.1.2.22 Test 022 - Head Select

This test checks the ability of the controller to select all available heads for the disk drive(s) attached. It also checks that the Translate command correctly indicates the correct head. The test normally performs 25 loops on all disk drives.

This test requires the entire winchester subsystem.

The test aborts if there are not drives to test, or if previous errors occurred in tests 11-15 or 19.

Interpretation: If test 22 alone fails, the problem is probably the controller. If test 19 fails, 22 will also fail. Use tests 17 and 18 to check the data path to the adapter. Investigate a fault in the drive cables (head select) or the ST506 drive interface.

6.3.1.2.23 Test 023 - Mode Sense

This test checks the Mode Sense command, and compares the results with the diagnostic defaults (set by the **DEFAULTS** or **MICROP** commands) or the superblock data read by test 21. The test compares this information: bytes per sector, physical number of cylinders and heads, precomp and reduced write current cylinders.

The entire winchester subsystem is required for this test.

The test aborts if there are no drives, or errors occurred previously in tests 1-5, 11-14, 20 or 21.

Interpretation: If the mode sense data does not match, the disk format may be incorrect. If the error occurs during a normal run, and the **DEFAULTS** or the drive type selection commands have not been used, the error is probably caused by the controller. The test will fail if the Superblock is incorrect or has parameters different than is expected. The causes of errors other than mode sense are identified by previous errors.

6.3.1.2.24 Test 024 - ECC Disable/Enable

This test checks that the controller accepts the command to enable and disable the error correction and retry logic. The test primarily checks controller commands.

This command requires the entire system.

The test aborts if there are no drives or if there were previous errors in tests 1-5 and 11-14.

Interpretation: If test 13 passes and 24 fails, the controller is probably bad.

6.3.1.2.25 Test 025 - Stop/Start Unit

This test checks that the controller accepts the start/stop unit command to position the heads in the landing zone. The test primarily checks controller commands.

The test requires the entire winchester subsystem.

The test aborts if there are not drives or if there are previous errors in tests 1-5 or 11-14.

Interpretation: If test 13 passes and 25 fails, the controller is probably bad.

6.3.1.2.26 Test 026 - Seek Cyl. 1

This test verifies the winchester subsystem's ability to position the drive heads correctly to cylinder 1. Head 1 and sector 0 is selected using the **SELECT** command, and then the **SEEK** command is issued. Positioning is then checked using the **TRANSLATE** command. This is done for all drives attached.

This test requires the entire winchester subsystem.

The test will abort if no drives are attached or if there were previous errors in tests 1-5.

Interpretation: If the translate command failed in tests 19 and 22, it will fail in 26 also. If tests 19 and 22 passed and a position error occurs in 26, then either the controller may be failing to select the correct head, or the drive direction or the stepping pulse lines may be failing.

6.3.1.2.27 Test 027 - Seek to 0

This test verifies the winchester subsystem's ability to position the drive heads correctly to cylinder 0. Head 1 and sector 0 is selected using the **SELECT** command, and then the **SEEK** command is issued. Positioning is then checked using the **TRANSLATE** command. This is done for all drives attached.

This test requires the entire winchester subsystem.

The test will abort if no drives are attached or if there were previous errors in tests 1-5.

Interpretation: If the translate command failed in tests 19 and 22, it will fail in 27 also. If tests 19 and 22 passed and a position error occurs in 27, then either the controller may be failing to select the correct head, or the drive direction or the stepping pulse lines may be failing. If test 26 passes and 27 fails, the error is probably in the controller or cables, though the drive may be failing to position correctly.

6.3.1.2.28 Test 028 - Seek to 255

This test verifies the winchester subsystem's ability to position the drive heads correctly to cylinder 255. Head 1 and sector 0 is selected using the **SELECT** command, and then the **SEEK** command is issued. Positioning is then checked using the **TRANSLATE** command. This is done for all drives attached.

This test requires the entire winchester subsystem.

The test will abort if no drives are attached or if there were previous errors in tests 1-5.

Interpretation: If the translate command failed in tests 19 and 22, it will fail in 27 also. If tests 19 and 22 passed and a position error occurs in 27, then either the controller may be failing to select the correct head, or the drive direction or the stepping pulse lines may be failing. If test 26 passes and 27 fails, the error is probably in the controller or cables, or the controller may be failing since 255 stepping pulses are issued, rather than just one as in tests 26 and 27.

6.3.1.2.29 Test 029 - Sequential-Alternate Seek

This test checks the ability of the disk drive to seek to cylinders in opposite directions and in increasing seek lengths. A seek is first issued to cylinder 0. Additional seeks are issued to higher cylinders, returning to 0 after each seek. The test normally loops 50 times.

This test requires the entire winchester subsystem.

The test will abort if there are no drives attached or if there were previous errors in tests 26-28.

Interpretation: If tests 26-28 pass and 29 fails, the disk drive is probably failing. Tests 26-28 passing indicate that the direction and stepping lines work properly. Intermittent errors in this test also indicate a failing disk drive.

6.3.1.2.30 Test 030 - Random Seeks

This test checks the ability of the disk drive to seek to cylinders in random direction with random seek lengths. 100 seeks are issued to random cylinders on all drives. The reliability of the SEEK command is verified by the controller itself, without use of the **TRANSLATE** command.

This test requires the entire winchester subsystem.

The test will abort if there are no drives attached or if there were previous errors in tests 26-28.

Interpretation: If tests 26-28 pass and 30 fails, the disk drive is probably failing. If 29 fails and 30 passes, the fault is probably with the **TRANSLATE** command, not with **SEEK**.

6.3.1.2.31 Test 031 - Max Seeks

This test checks the ability of the disk drive to seek to cylinders in opposite directions with maximum seek lengths. 20 seeks are issued to cylinders at opposite ends of each drive. Due to the length of the seek, if the drives are formatted to allow buffered seeking and there are two or more drives attached, this test also exercises the ability of the subsystem to do overlapped seeks. After each seek, the head position is checked using the **TRANSLATE** command.

This test requires the entire winchester subsystem.

The test will abort if there are no drives attached or if there were previous errors in tests 26-28.

Interpretation: This test may fail if there are excessive disk defects that have been formatted out, which causes position errors since the **TRANSLATE** data does not match the expected data. If tests 26-28 pass and 31 fails, the disk drive is probably failing. If 31 fails and other seek tests pass, the controller may be failing to correctly process overlapped seeks.

6.3.1.2.32 Test 032 - Write Test Area

This test checks the **WRITE** command, and the ability to seek to the innermost track. The test loops 128 times writing a different pattern for each loop. Four sectors are written on the test cylinder, and then read back and compared.

This test requires the entire winchester subsystem. The test is performed with interrupts ON and DMA logic enabled.

The test aborts if there are not drives or the drives are write protected, or if errors occurred in tests 11-14, 21, 22, or 26-30.

Interpretation: If test 32 passes, the disk and controller are working together properly. If an error occurs, the controller operation code and the error are displayed to indicate the error. Test 32 will fail if more primitive tests fail. Controller detected errors probably indicate a drive failure.

6.3.1.2.33 Test 033 - Long Read

This test primarily checks the 10-byte **LONG READ** command, and also the ability to transfer 136 sectors from the disk to host memory. The sectors are transferred using the head and cylinder selected in earlier tests, then transferred to the diagnostic read buffer for comparison.

This test requires the entire winchester subsystem. It is performed with interrupts ON and DMA logic enabled.

The test aborts if there are not drives or the drives are write protected, or if errors occurred in tests 11-14, 21, 22, or 26-30.

Interpretation: Passing test 33 indicates that the majority of the subsystem works well, including DMA and interrupt logic. The test will fail if more primitive tests fail, such as seek and read tests. Controller detected errors indicate a drive failure. Test 33 will fail if the DMA tests 6 or 8 fail, or if the command completion interrupt fails, as tested in test 29.

6.3.1.2.34 Test 034 - Long Write

This test primarily checks the 10-byte **LONG WRITE** command, and also the ability to transfer 136 sectors to the disk from host memory. The sectors are written.

This test requires the entire winchester subsystem. It is performed with interrupts ON and DMA logic enabled.

The test aborts if there are not drives or the drives are write protected, or if errors occurred in tests 11-14, 21, 22, or 26-30.

Interpretation: Passing test 33 indicates that the disk drive and controller work together, but the success of the write must be verified by successive tests. The test will fail if more primitive tests fail, such as seek and read tests. Controller detected errors indicate a drive failure. Test 33 will fail if the DMA tests 6 or 8 fail, or if the command completion interrupt fails, as tested in test 29.

6.3.1.2.35 Test 035 - DMA Parity or ECC

This test verifies that the CMB correctly detects and reports a main memory parity error while the controller is performing DMA access with the controller as bus master. If the system does not have parity logic, the test verifies correct detection of an ECC error during DMA transfer. The test is an extension of test 6.

This test only requires the controller.

The test aborts if errors occurred in test 6.

Interpretation: If test 6 fails, so will 35. If 6 passes and 35 fails with "DMA parity error not received," the failure is not in the controller/adapter, but indicates that he CMB cannot detect a parity error or ECC error caused by a DMA cycle while the controller is bus master.

6.3.1.2.36 Test 036 - Verify

This test verifies the **VERIFY** controller command, and the no controller detected errors occur.

This test requires the entire winchester subsystem.

The test aborts if no drives are attached, or if errors occurred in tests 1-6, 11-14, 20-22 or 26-31.

Interpretation: If test 37 fails, test 36 may fail. If 37 passes and 36 fails, the error is probably in the controller, but may be located in the drive since it checks the ECC from a previous write operation. Failures in DMA or interrupt logic also cause this test to fail.

6.3.1.2.37 Test 037 - Write and Verify

This test verifies the WRITE-AND-VERIFY controller command, and the no controller detected errors occur.

This test requires the entire winchester subsystem.

The test aborts if no drives are attached or they are write protected, or if errors occurred in tests 1-6, 11-14, 20-22 or 26-31.

Interpretation: If test 34 or 35 fails, test 37 will fail. If 34 passes and 37 fails, the error is probably in the controller, but may be located in the drive since it checks the ECC after the write operation completes. Failures in DMA or interrupt logic also cause this test to fail.

6.3.1.2.38 Test 038 - Random Cylinder Read

This test checks the ability to do an implied seek to a random cylinder, that an entire cylinder can be read without error. The controller is also checked for head switching and auto-seek to obtain one additional sector beyond the end of each track. The test normally performs 25 loops.

This test requires the entire winchester subsystem.

The test aborts if there are no drives or if previous errors occurred in tests 1-6, 11-14, 20-22, 26-31.

Interpretation: If tests 20-22 or 30 fail, 38 will fail. Passing this test gives good assurance that the entire subsystem works.

6.3.1.2.39 Test 039 - Random Write/Read

This test verifies the ability of the controller to do a successful write to a random cylinder and head. The data is read back and compared to the data written. The test normally loops 25 times.

This test requires the entire subsystem.

The test will abort if there are no drives or they are write protected, or if previous errors occurred in tests 1-6, 11-14, 20-22 or 26-32.

Interpretation: If test 37 passes and 38 fails, use previous tests (e.g., 30, 32, 34) to isolate the failure to the write capability. If tests 32 and 34-36 pass and 38 fails, the error is probably caused by an unsuccessful read. Analyze the comparison data to determine if the error is caused by a particular bit or byte, or if all data read is bad.

6.3.1.2.40 Test 040 - Worst Case Write/Read

This test checks the general reliability of the subsystem, reading and writing random disk locations. It should be performed as the final test when all other tests pass, if a problem is still suspected. The test normally loops 25 times.

This test requires the entire winchester subsystem.

The test aborts if there are no drives or they are write protected, or if there were previous errors in tests 1-6, 11-14, 20-22 or 26-32.

Interpretation: Failure of test 40 when all other tests pass indicates a data reliability problem.

6.3.1.3 WINCHESTER TEST COMMANDS

The following commands can be entered directly at the disk test prompt to perform operations or set test parameters. These commands are in addition to the general diagnostic executive commands.

Use the commands to investigate faults more fully, such as to check the data in buffers.

ADDRESS {*n*}

Select or display the disk controller address.

BERRC

This command clears the bus error bit in the adapter status byte.

CCONTROL

This command clears the SASI controller, checking status before and after.

CLRLOG

This command clears the logout list and previous error flags.

DEFAULTS

This command initializes the controller by first clearing the controller, loading the vector register and DMA register, restoring all default parameters for the diagnostic for a Rodime drive, selects unit 0, and sets write protection.

ERRORS

This command displays controller error codes.

FAST {*n*}

This command enables all of the disk logic tests, but at a faster rate than the normal run. **FAST 1** (or simply FAST) sets fast mode; **FAST 0** returns to normal run speed. FAST should not be used for fault isolation since it makes intermittent errors difficult to identify. After issuing a FAST command, **RUN** begins the testing.

INIT

This command initializes the diagnostic and restores all tests to the test execution list.

INQUIRE

Sizes and displays the system configuration, and tests the controller registers. The test results in either "OK" or "ERROR". If "ERROR" is displayed, the controller should be replaced.

KEYS

This command downloads codes to an EVDT terminal to program function keys F1 through F14.

Key#	Command	Description
1	BOOT	Boot operating system
2	LO DISK	Load disk test
3	RDA	Read data from input reg
4	SENSE	Issue the sense command
5	STAT	Issue the status command
6	CC	Clear controller command
7	WRITEOK	Allow disk write tests
8	LOG	Display the logout
9	TE	Display the current test list
10	FAST	Enable fast run of tests
11	QUICK	Begin a quick run of tests
12	RUNIN	Issues RUNIN command
13	INIT	Issues INIT command
14	RUN	Issues RUN command
SHIFT + Function Keys 1-14		
15	BOOT	Boot operating system
16	LO UTIL	Load the UTIL program
17	LO DISK	Load disk function select tests
18	SELECT	Select SASI
19	not used	
20	RDIS	Display read buffer
21	WDIS	Display write buffer
22	CLRLOG	Clear the logout data
23	LIST	List the diagnostic commands
24	not used	
25	not used	
26	RESTA	Restart the diagnostic
27	DEFAU	Reset the defaults
28	REV	Display the diagnostic revision

LOGOUT

This command displays the current list of errors detected by the diagnostic, as well as the count of operations done.

LUNIT {*n*}

Selects the logical unit (controller and drive) to test, where *n* is 0-7. If UNIT n,n+1 is issued, both drives are tested using the appropriate controller.

MAXTOR {*n*}

This command defines the drive capacity for Maxtor drives. If the argument is not specified, the drive is identified as a 140 MB drive. Values for *n* are 85, 105 or 190.

MICROPOL {*n*}

This command defines the drive capacity for Micropolis drives. If the argument is not specified, the drive is identified as a 50 MB drive. Values for *n* are 50 and 85.

MODESENS

This command issues the Mode Sense command, and displays the data received.

OVERLAP {*n*}

This command enables/disables the test function of waiting for two interrupts from a SEEK command. This is done on single board controllers only. **OVERLAP 1** enables the dual interrupt seek checking; **OVERLAP 2** disables the feature.

QUICK

Execute a subset of the disk logic tests to. The command should be used for a quick determination of the hardware status, such as after swapping a component. The **QUICK** test option should not be used for fault isolation.

RDATA

This command reads data from the input register and displays it on the VDT screen.

RDISPLAY {*n*}

Display the contents of the read buffer. The argument specifies the number of words to display. Without an argument, a page of the read buffer is displayed.

READONLY

This command turns ON write protection, prohibiting tests from writing on the disk. (See WRITEOK)

RESTART

This command initializes the diagnostic by issuing the RECALIBRATE command, resetting necessary flags to restore operation to INIT state, selecting unit 0, and setting write protection.

REVISION

Displays the program title and revision.

RODIME {*n*}

This command defines the drive capacity for Rodime drives. If the argument is not specified, the drive is identified as a 20 MB drive. Values for *n* are 20, 40 and 53.

RUNIN {*n*}

This command executes long duration testing on the disk for data reliability checking. Each loop displays a logout of the operations performed and the number of errors detected. Using an argument causes a different type of testing, as follows:

Argument	Operation
1	Do controller burn-in tests with no logout.
2	Do controller burn-in tests with logout.
3	Do controller and disk burn-in.

RUNIN runs continuously until the user presses **CTRL + C** or **ESCAPE**.

SEEK *n*

This command issues a **SEEK** command to cylinder n of the last selected unit.

SELECT

Select the SASI controller, and test for correct status.

SENSE

This command issues the Mode Sense command. If an error is indicated by the check bit in the sense byte, it is displayed. If the valid bit is set, the cylinder, head and sector where the last error occurred is displayed.

START *n*

This command issues the Start Unit controller command and returns the heads to the drive's default position. The argument specifies the unit number, 0 or 1.

STATUS *n*

This command reads the adapter status. If there is no argument, the status is displayed. If there is an argument, the command waits for the status to match the argument. The command will wait indefinitely, unless **CTRL + C** is pressed.

STOP *n*

This command issues the Stop Unit controller command to the specified device to position the heads in the landing zone for shipping. The argument specifies the unit number, 0 or 1.

SUMMARY

This command displays a table of the results of the test pass. The table indicates those tests that PASS, FAIL, ABORT or SKIP.

TSTDMA {*n*}

This command executes a DMA test sequence. Data is accessed from memory in DMA and transferred to the input register a byte at a time for comparison to data contained in memory. The optional argument specifies the data type to be created in the write buffer. If no argument is specified, the next type of data is used. The data types are:

Data Type	Description	Data Block Created
0	All zeros	0000 0000 0000 0000 0000
1	All ones	FFFF FFFF FFFF FFFF FFFF
2	Floating 0	FFFE FFFD FFFB FFF7 FFEF
3	Floating 1	0001 0002 0003 0008 0010
4	Random	F946 EA89 12DD 32A0 00FF
5	All 5's	5555 5555 5555 5555 5555
6	Di-bit	CCCC CCCC CCCC CCCC CCCC
7	Tri-bit	E38E 38E3 8E38 E38E 38E3
8	Quad-bit	F0F0 F0F0 F0F0 F0F0 F0F0
9	Zero di-bit	6DB6 DB6D B6DB 6DB6 DB6D
A	Incrementing	0001 0002 0003 0004 0005
B	Zero tri-bit	7777 7777 7777 7777 7777
C	Zero quad-bit	7BDE F7BD EF7B DEF7 BDEF
D	One di-zero	9249 2492 4924 9249 2492
E	One tri-zero	8888 8888 8888 8888 8888
F	One quad-zero	8241 0824 1082 4108 2410
10	Increase freq.	FF00 FE03 F03E 0F0E 32AA
11	Decrease freq.	24AA 6663 8E38 7878 783F
12	Jitter	9659 6596 5965 9659 6596
13	Worst case 1	D936 4DB6 4ED9 364D B64E
14	Worst case 2	6DB6 BD0F 6DB6 BD0F 6DB6
15	Worst case 3	DADA CA58 C2FE DADA CA58

TSTINT

This command verifies that the adapter generates an interrupt from a forced bus error condition. This command can detect that the bus error is not generated as expected and that the controller does not generate an interrupt at all.

UNIT {*a*,*b*}

Selects drive unit(s) to test, overriding the default selection. If only one drive is specified, only that drive is tested.

USTATUS {*n*}

This command issues the Test Unit Ready command and checks for errors. If an argument is used, the unit specified is selected for future testing. If no argument is used, the last unit selected remains in effect.

WDATA *n*

This command writes the byte n to the output register. This can be used to send a controller command.

WDC {*n*}

Selects the winchester controller to test. If no argument is used, the currently selected controller and type are displayed. When the diagnostics are first loaded, testing is set to be performed on all controllers found. Use this command to restrict testing to a single controller.

WDISPLAY {*n*}

Display the contents of the write buffer. The argument specifies the number of words to display. Without an argument, a page of the write buffer is displayed.

WRITEOK

This command turns OFF the automatic write protection feature of the disk diagnostics, allowing tests to write to the disk. This can destroy data and programs on the disk. Certain tests are skipped (aborted) unless **WRITEOK** is executed, as described in the test descriptions. (See **READONLY**)

6.3.1.4 COMMAND MACROS

TSTREAD

This macro selects those tests which do not write on the disk, and enable running tests 1-31, 33, 35, 36 and 38. It also executes these commands:

WRITEOK, OPTION PAUSE, RUN

After the test, **READONLY** is executed.

TSTSEEK

This macro executes all the seek tests once, by executing the commands:

TEST 0,26-31; RUN

TSTWRITE

This macro executes these commands:

FAST, OPTION PAUSE, WRITEOK, RUN

6.3.1.5 ERROR DESCRIPTIONS

The following paragraphs give more detailed descriptions of the causes of the error messages listed for each test.

6.3.1.5.1 Bad Status Before Command Issue

- Adapter not ready prior to issuing command

  Either the controller was busy prior to issuing the command, or the command caused the controller to stay busy after the command was issued.

- Unit write protected

  The write protect flag in memory was set when a write operation was attempted.

- Bad status after SASI selected

  The adapter status should indicate selected and busy condition and output register free. After the SELECT command is issued the adapter status of C2 is expected, and a timeout occurred.

- DCB byte transfer timeout

  The bytes which make up the DCB are sent to the controller in simple output mode. Correct status is checked prior to sending each byte. A timeout occurred waiting for status C2 before transferring the next DCB byte.

- Data transfer timeout

  If the operation which transfers disk data in the non-DMA mode, or if the command itself requires sending data associated with the command is simple I/O, status must be correct before the data is transferred. If the correct status is not obtained in the allotted time, this error occurs.

- Bad status found before command completion

  After the command is issued, the program will wait for an interrupt and check status, or will wait for correct ending status from the adapter.

6.3.1.5.2 Bad Status After Command Issued

- Expected bus error status not found

  The program attempts to get an interrupt from the bus error status. After trying to force the bus error, the status bit did not set.

- Unexpected adapter detected address bus error

  The adapter detected a bus error while doing a DMA transfer.

- Adapter detected bus error didn't generate interrupt

  The adapter has the bus error bit set in status, but it did not generate an interrupt as expected by the program.

- Can't obtain the ending status

  The ending status byte was not read from the SASI controller for ending status error checking because the command completion status was not obtained.

- Message phase timeout

  After the ending status byte is read from the controller, it should indicate message phase with a status of E8 hex.

- Message byte not zero

  The actual message byte read from the controller while in message phase should be 0.

- Error processing: SENSE command failure

  The previous operation had the check bit set in the ending status byte indicating that a SENSE command was necessary. The SENSE command is automatically issued in this state. This error is displayed when the SENSE command also failed to complete correctly.

- Position error

  After the diagnostic has selected seek verification, the translate command issued by the program indicated that there is a difference between the physical position requested and the actual position that the SASI controller indicates.

- With interrupts enabled, no interrupt occurred

  After the command is issued, the controller did not generate an interrupt as expected in the time allotted.

- With interrupts disabled, an interrupt occurred

  The program detected an interrupt when the controller was not expected to generate one.

- Non-zero status after reset

  The controller status should be equal to 0 when the clear controller command is issued.

- Ending status non-zero

  The command completion status byte read from the controller after the command completes is not 0. The ending status obtained from the controller is displayed along with the command code. The command code is the DCB opcode byte.

- Command aborted before completion

  The program found operation status bytes in error while trying to check the results of the operation. This normally occurs when the operator hits ESCAPE during command processing.

- Unexpected sequence or program error

  Normally caused by a program fault, where the expected control over the result of the operation does not exist. The program should be reloaded in this case, since it is not operating as expected.

- Bad unit x status

  The controller detected bad unit status after the command. The unit, error code from the SENSE operation. logical sector address, and the cylinder, head, and sector where the error occurred if applicable, is displayed.

- Cylinder exceeds maximum

  The cylinder requested is too large, or larger than the cylinder specified by default values. This prevents the user from trying to seek to an invalid cylinder. The cylinder requested is displayed, along with the program maximum.

- Buffers don't compare

  The write buffer data is not the same as the read buffer data. The expected and received words are displayed, as well as the buffer addresses. If the data in the read buffer is the hex word "BAD1", then the read operation did not occur.

- Adapter address bus error

  The program received a bus error from a legal adapter address.

- Controller detected error

  At the completion of a controller command, the ending status byte indicated an error. The results of the sense command are displayed as the error code followed by the command causing the error.

- Bus error status won't reset

  The command to reset the bus error bit in status did not cause the status to clear.

- Reset-line status not set

  After issuing the SASI reset command, the status disk not indicate SASI reset activity.

- DMA Test data compare error

  The DMA test found that the data in memory is not the same as the data accessed via the DMA test cycle. This normally means that the DMA register is not correctly accessing the memory area expected.

- Vector Reg Data Loop-through error

  Data written to the vector register is not the same as the data read back. Expected and received data are displayed.

- Control Reg Data Loop-through error

  Data written to the control register (least significant four bits) is not the same as the data read back. Expected and received data are displayed.

- Data Registers Loop-through error

  Data written through the output data register in test mode is not the same as the data read from the input register. Expected and received data are displayed.

- Device busy timeout after multiple retries

  If the controller indicates that it is busy by the ending status, the operation is retried automatically. This error is displayed when the controller continually indicates busy status.

- WARNING: BAD SUPERBLOCK!

  The superblock does not contain valid data, which is needed to do an auto-size.

- Check bit not set in ending status

  The ending status byte read during command completion did not have the check bit on as expected. An error is forced to set the check bit.

- Input reg. full status won't clear

  The status register bit 3 stays on after timeout and a byte is read from the input register.

- Input reg. full status won't set

  The status register bit 3 won't dome on after the command is issued, or after a previous register read.

- Output reg. empty status won't clear

  The status register bit 1 won't clear after a byte is written to the output register.

- Output reg. empty status won't set

  The status register bit 1 won't set after the command is issued, or after a byte is sent.

- Timeout waiting for null status after msg. phase

  After the message byte is read from the controller, the status did not return to 00.

- With interrupts disabled, bus-error int. occurred

  The control register was written to disable interrupts, and an interrupt occurred after forcing an adapter detected bus error.

- No interrupt after interrupt enable transition

  With a bus error interrupt status active, an interrupt is not generated when the interrupt enable bit is turned off and back on.

- Interrupt generated after bus error cleared

  After clearing the bus error by command, an interrupt was generated after the interrupt enable bit was turned off and back on.

- Modesense data doesn't match auto-size configuration

  The diagnostic did not find a comparison between the data returned by the modesense command and the data found in the SUPERBLOCK.

6.3.1.5.3 Interpretive Errors

Interpretive errors are displayed as warnings or to further qualify the error which has occurred.

- SASI reset active

  During error processing, the reset line status bit in status was found on.

- Lost BUSY status

  During the processing of the command after the SASI controller selected, the busy status bit was found off.

- Command complete timeout

  The status of CC hex was not read by the program in the time allotted.

6.3.1.5.4 Disk Warning Messages

These messages are not standard error messages. They may occur but not indicate that an error has occurred. The condition or warning should be noted, however.

- Interrupt service routine hung

  This message is displayed if the interrupt service routine is called 128 times before the diagnostic can reset the counter.

- Invalid emulation instruction

  This message is displayed before a trap 15 if the emulation code is invalid. Normally this indicates that the program is corrupted.

- PROGRAM CHECKSUM ERROR!!

  A checksum is generated when the program is loaded. If the user reinitializes the diagnostic, the checksum is again generated and compared with the original. If they do not match, this message is displayed, indicating that the program has been changed in some way. If this message is displayed, the diagnostic should be reloaded.

6.3.2 MCS Logic Test

The Magnetic Cartridge Streamer (MCS) Logic Test checks the hardware and firmware of the MAI MCS subsystem. It is intended to test the ability of the MCS subsystem to operate with the CMB in an operating system environment. The subsystem consists of the MCS controller and the MCS drive itself.

The test begins with simple tests and, depending on the success of failure of these tests, proceeds to more complex operations. The MCS Logic Test consists of 25 subtests, which are described in section 6.3.2.2.

6.3.2.1 Loading and Running the MCS Logic Test

The MCS Logic Test is loaded and run in the usual manner for logic tests. At either the diagnostic executive prompt, or at the prompt for another logic test, enter:

```
load mcs
```

To run the logic tests in normal mode, enter:

```
run
```

Refer to the command descriptions in section 6.3.2.3 for additional test run options.

6.3.2.2 MCS Test Descriptions

6.3.2.2.1 Test 1 - Status Register Addressing

This test attempts to address the controller's status register. If this register is addressable using a normal bus read cycle, the test exits. If a bus error occurs, a message is displayed indicating that a serious problem exists in the controller's board-decode and/or DTACK generation logic.

6.3.2.2.2 Test 2 - Go Register Addressing

This test is similar to test 1, except that it attempts to access the controller's go register.

6.3.2.2.3 Test 3 - Reset

This test issues a reset command. An error is reported if the controller does not set the CTS and CR bits within the time allotted.

6.3.2.2.4 Test 4 - Read Status

This test issues a read status command. An error is reported if the controller does not set the CTS and CR bits within the time allotted.

6.3.2.2.5 Test 5 - IOPB Processing

This test verifies the controller's ability to do a DMA read cycle from the I/O Parameter Block (IOPB) in host memory, parse a valid IOPB, executed the operation requested, and do a DMA write cycle to host memory. The test checks that the operation is performed within the time allotted, and whether as error status is returned.

6.3.2.2.6 Test 6 - Rewind

This test issues the rewind command. The beginning of tape (BOT) must be reached within 70 seconds.

6.3.2.2.7 Test 7 - Rewind Interrupt

This test is the same as test 6, except that interrupts are enabled and one interrupt is transmitted along with the rewind command. The system checks that exactly one interrupt is received, and that good status is returned.

6.3.2.2.8 Test 8 - DMA Integrity

This test verifies the controller's ability to perform DMA to the intended area. This is done by filling the data area of memory with a known pattern. The data is then written to tape, the tape is rewound, the data read back and compared with the data written. The data pattern in memory is also checked, to verify that the DMA did not change it. The data pattern is then inverted, and the routine is repeated.

This test is not included in the bootable disk resident version of the MCS Logic Test.

6.3.2.2.9 Test 9 - Write and Read One Record

This tests attempts to write and read one 256 word (512 byte) record. A buffer is filled with data, written, read back into another buffer and compared. The test reports any miscompares, as well as status errors during the write, rewind or read.

6.3.2.2.10 Test 10 - Write and Read One Record Interrupt

This test is similar to test 9, except that an routine is sent and interrupts are enabled. Then the write, rewind and read operations are performed. Exactly one interrupt is expected after each IOPB. Errors are reported if an interrupt is missed or if bad status is returned.

6.3.2.2.11 Test 11 - Write/Read Shoeshine

This test writes 50 blocks (512 bytes) of data, with a slight delay between blocks. The must drive reposition before each block, causing a "shoeshine" type motion. Once all 50 blocks are written, the system is reset and the blocks are read and compared to the data written. Any miscompares are reported.

6.3.2.2.12 Test 12 - On-The-Fly Write/Read Shoeshine

This test is the same as test 11, except that it uses on-the-fly reads and writes.

6.3.2.2.13 Test 13 - Writes and Read Multiple Records Ramp-Up

This test checks for word count related problems by performing multiple record reads and writes. The test begins writing 4 blocks, using pseudo-random data, incrementing the block count by four for each loop until a maximum of 52 blocks is reached. The test is run once with interrupts disabled, then with interrupts enabled. In both cases, data miscompare errors are reported, and in the later case missed interrupts are also reported.

6.3.2.2.14 Test 14 - File Addressing

This test checks the ability to write and read filemarks, and to skip over and keep track of them. This is done by writing 16 one block files, and then checking the ability to position at the beginning of each of them.

6.3.2.2.15 Test 15 - Append

This test checks the ability to append a block to the end of tape data. This is done by first writing between 1 and 31 blocks, the appending another block, all with unique data patterns. The data of all blocks written, including the appended block, is compared with the written data.

6.3.2.2.16 Test 16 - Streaming Write/Read

This test checks the subsystem's ability to read and write the entire tape surface without any repositioning. Two blocks are created and alternately written until the EOT is reached. The data is then read and compared with what was written. Any data mismatches are reported.

6.3.2.2.17 Test 17 - Interrupt-Driven Streaming Write/Read

This test is similar to test 12, except that interrupts are used. Exactly one interrupt is expected after each of the write, rewind and read operations. An error is reported if an interrupt is missed or if bad status is returned.

6.3.2.2.18 Test 18 - On-The-Fly Streaming Write/Read

This test is similar to test 16, except that the extended on-the-fly commands are used for I/O. Any data miscompares or status errors are reported.

6.3.2.2.19 Test 19 - BOZO Parameter

This test tests non-functionality by forcing error conditions. First, a write is attempted with a non-modulo 256 word count. Next, an undefined opcode is sent. If either of these fail to cause an error, that condition is reported.

6.3.2.2.20 Test 20 - Bus Master Parity Error

This test checks the CMB's ability to detect parity errors when the MCS controller is bus master. Bad parity is injected into memory, and the controller then performs DMA access to that memory. The test reports an error if the CMB does not detect the parity error.

6.3.2.2.21 Test 21 - Non-Existent Memory Data Transfer

This test requires operator intervention. This test checks the controller's ability to detect a bus error while it is bus master. A write operation is set with a non-existent memory address as the buffer location. Upon completion, the controller's status register is examined for the bus error bit. If it is not set, the test reports an error.

6.3.2.2.22 Test 22 - Write-Protected Cartridge

This test requires operator intervention. The test instructs the operator to remove the cartridge, turn the write-protect plug to "SAFE" and reinsert it. The test then attempts a write operation, and checks for write-protect status. If the status is not detected, the test reports the error. The test then instructs the operator to return the tape to "NOT SAFE" condition.

6.3.2.2.23 Test 23 - Missing Cartridge

This test requires operator intervention. The test instructs the operator to remove the cartridge. A read operation is then attempted, and the test checks for a cartridge missing status. If the status is not detected, the test reports an error. The test then instructs the operator to return the cartridge.

6.3.2.2.24 Test 24 - Streaming Write Only

This test requires that the OPTION MANUAL command be issued before it is run. No other manual intervention is required.

This test performs a write operation only. It can be used with test 25 (Read Only) to isolate a data problem to the write operation. If tests 24 and 25 are looped several times, and the data miscompare occurs on the same data pattern every time during the read cycle, the problem is almost certainly with the write operation.

6.3.2.2.25 Test 25 - Streaming Read Only

This test requires that the OPTION MANUAL command be issued before it is run. No other manual intervention is required.

This test performs a read operation only. It can be used with test 24 (Read Only) to isolate a data problem to the read operation.

6.3.2.3 MCS LOGIC TEST COMMANDS

The following commands are in addition to the general diagnostic executive commands. They can be entered directly at the MCS prompt.

AUTOTENS

This command sets a flag to cause a tape retension at the start of each pass through the test sequence.

CLRLOG

This command initializes the statistics log.

DICTNARY

This command displays certain status and other definitions that are too long to display in the usual error messages. Printer output should be started using diagnostic executive commands prior to executing this command.

DMP_CORE *x,y*

This command displays the specified bytes of core memory. The first argument, *x*, specifies the number of bytes to report, and *y* specifies the beginning address. For example, to dump 500 bytes (hex) beginning at address 2000 (hex), use this command:

```
DMP_CORE 500,2000
```

DMP_IOPB *n*

This command displays the contests of the I/O Parameter Block. The argument is a number in hex from 0 to E representing the relative location of the block in the IOPB chain.

ERASE

This command erases the entire tape.

LOGOUT

This command displays the most recent statistics log generated by RUNIN.

OPTIONS {*list*}

This is the Executive **OPTIONS** command (refer to Section 4.3.2). The B15 option causes the test to display the statistics log after every pass of the diagnostic.

QUICK

This command runs only the shortest MCS tests to provide a fast verification of the operating condition of the subsystem. This command should not be used for fault isolation, but only to verify general system operation.

RETENSION

This command causes the cartridge to be retensioned by running the tape form BOT to EOT, then back to BOT.

RUNIN

This command runs all selected tests in a continuous loop until stopped by **CTRL + C**. After each pass, the test displays an error statistics log.

6.3.2.4 MCS Logic Test Error Messages

- Status Errors

  Status errors are caused by any of the following:

  - Self-test errors detected in the controller status register
  - Bus error bit in controller GO register is set
  - Anything but 8000 hex status in an IOPB primary status word

- Data Miscompare Errors

  This error occurs when the data read does not match the data written on a bit-by-bit basis. The message shows the expected data, the actual data, and the buffer addresses in host memory where both data exist.

- Software Timeout Error

  Many operations have a set time limit within which they must complete, returning an appropriate status. Any failure to occur in the allotted time results in a timeout error.

- Unexpected Interrupt Error

  This error occurs whenever an interrupt is detected for which a request was not issued. This type of error is fatal, and is likely to cause the system to crash. If this occurs, the diagnostic displays a message and exits to the debugger (MAI 2000) or MDS (MAI 2500/3000/4000).

6.3.3 MTS Logic Test

The Magnetic Tape Streamer (MTS) Logic Test checks the hardware and firmware of the MAI MTS subsystem. It is intended to test the ability of the MTS subsystem to operate with the CMB in an operating system environment. The subsystem consists of the MTS controller, the SCSI controller adapter, and the MTS drive itself.

The test begins with simple tests and, depending on the success of failure of these tests, proceeds to more complex operations. The MTS Logic Test consists of 35 subtests, which are described in Section 6.3.3.2.

6.3.3.1 Loading and Running the MTS Logic Test

The MTS Logic Test is loaded and run in the usual manner for logic tests. At either the diagnostic executive prompt, or at the prompt for another logic test, enter:

```
load mts
```

To run the logic tests in normal mode, enter:

```
run
```

Refer to the command descriptions in Section 6.3.3.3 for additional test run options.

6.3.3.2 MTS Test Descriptions

The following paragraphs describe the individual MTS subtests.

6.3.3.2.1 Test 1 - Addressability

This test checks the addressability of the adapter's registers. Only the adapter is required for this test.

The test writes and reads (as appropriate) each register on the adapter.

6.3.3.2.2 Test 2 - Write/Read Control Register

This test checks the control register on the adapter board. Only the adapter is required for this test.

The test alternately writes various data patterns to the register, and reads the data back, comparing the results. The adapter is reset and the control register is cleared at the end of the test.

6.3.3.2.3 Test 3 - Data Registers Loop Through

This test checks the data register logic for the adapter board. Only the adapter is required for this test.

The test writes various data patterns to the output data register on the adapter, and compares these patterns with what is read from the input data register.

6.3.3.2.4 Test 4 - Vector Register Loop Through

This test checks the adapter vector register logic. Only the adapter is required for this test.

The test writes each valid user interrupt vector number to the vector register and read them back for comparison. Any vector register errors are detected and reported.

6.3.3.2.5 Test 5 - Null Status

This test checks the adapter reset logic. Only the adapter is required for this test.

The test selects the adapter and resets it. The status register is then checked, and must be zero.

6.3.3.2.6 Test 6 - Test Mode DMA

This test checks the adapter DMA transfer and logic. Only the adapter is required for this test.

The test cycles a complete block (512 bytes) through the adapter output and input data registers, on word at a time, by DMA. The full block is cycled for each loop. The data is compared, and any comparison errors are reported.

6.3.3.2.7 Test 7 - Bus Error Status

This test checks the adapter DMA logic for ability to detect and set bus errors properly. Only the adapter is required for this test.

The test attempts a DMA cycle from non-existent memory and verifies that the adapter detects and set the bus error status. A bus error clear is the performed and verified.

6.3.3.2.8 Test 8 - DMA Register

This test verifies that all adapter memory addresses are accessible by DMA. Only the adapter is required for this test.

The test sequentially accesses 512 bytes of memory and cycles each block through the DMA test. Errors indicate that the DMA logic cannot access all memory locations.

6.3.3.2.9 Test 9 - Conditional Interrupt

This test verifies that the bus error interrupt logic functions properly. Only the adapter is required for this test.

The test disables interrupts, then forces bus errors and verifies that no interrupts are received. Interrupts are then enabled, bus errors are forced and it is verified that interrupts are received. Errors indicate a problem with interrupt logic.

6.3.3.2.10 Test 10 - Vectored Interrupt

This test checks the adapter vectored interrupt logic. Only the adapter is required for this test.

The test forces a bus error for each vectored interrupt address. Errors indicate a problem with the vectored interrupt logic.

6.3.3.2.11 Test 11 - Parity Generation/Detection

This test verifies the adapter parity generation and detection logic is functioning. Only the adapter is required for this test.

The test writes even and odd numbers of bits through the output data register, and verifies that the auxiliary status parity error bit is set/reset properly.

6.3.3.2.12 Test 12 - Select Status

This test verifies the adapter's and controller's response to a Select command. Both the controller and adapter are required for this test.

The test selects the adapter, and verifies that the adapter enters the command in phase. The adapter is then reset and its status is checked for a free state. Errors indicate that the adapter is not responding to the Select, or that the controller is not responding.

6.3.3.2.13 Test 13 - Handshake

This test checks the system handshake routine. Both the controller and adapter are required for this test.

The test processes a complete handshake, with interrupts enabled and then with interrupt disabled. Successful handshake is verified with a Test Unit Ready command. Failure of this test indicates a controller error.

6.3.3.2.14 Test 14 - Test Unit Ready

This test checks for Unit Ready status on the test unit. Both the controller and adapter are required for this test.

6.3.3.2.15 Test 15 - Unit Status

This test checks for a ready status on the controller. Both the controller and adapter are required for this test.

The test issues a Test Unit Ready command to the controller if the unit is ready and the proper sense status is returned following a reset command.

6.3.3.2.16 Test 16 - Request Sense

This test verifies that the adapter and controller respond to the Request Sense command with interrupts enabled and disabled. Both the controller and adapter are required for this test.

The test processes a Request Sense command through the normal driver sequence to verify that the subsystem responds to the full command sequence. Errors probably indicate a failing controller.

6.3.3.2.17 Test 17 - Abnormal Termination

This test checks the controller for proper checking of command formats and parameters. Both the controller and adapter are required for this test.

The test sends commands to the controller containing invalid logical unit numbers, reserved bit set and invalid commands, to verify that all cases result in the return of an invalid request status. Errors indicate that the controller is not properly checking the command formats and parameters.

6.3.3.2.18 Test 18 - Read Status

This test sends a valid Mode Sense command to verify that the current modes of the drive may be sensed. The entire subsystem is required for this test.

6.3.3.2.19 Test 19 - Rewind

This test issues a rewind command, both with interrupts enabled and with them disabled. Errors indicate controller or drive problems. The entire subsystem is required for this test.

6.3.3.2.20 Test 20 - Mode Select/Sense

This test sends various mode select parameters to the drive, followed by a mode sense to check that the parameters are set. Failure of this test would prohibit the use of modes other than the power-on default. The entire subsystem is required for this test.

6.3.3.2.21 Test 21 - Data Integrity

This test writes a large block of data from memory to the tape, followed by a rewind and read. The data is verified by performing a checksum on the data written and on the data read.

6.3.3.2.22 Test 22 - Write/Read One Record

This test issues write commands with various data patterns. Each write is followed by a rewind, read and compare of the data buffers to test for data integrity on the media. The entire subsystem is required for this test.

6.3.3.2.23 Test 23 - Write/Read Shoeshine

This test writes 50 blocks of data, one at a time, with a delay between each write to cause the drive to underrun, requiring a reposition (shoeshine). After writing all blocks, the tape is rewound and read, comparing the data for errors.

6.3.3.2.24 Test 24 - On-the-fly Write/Read Shoeshine

This test writes 50 files of 20 blocks each with a filemark after each file. The tape is then rewound after each file, read and checked for data comparison and filemark detection.

6.3.3.2.25 Test 25 - Write/Read Multiple Records Ramp-up

This test writes multiple files of increasing block length, each with a different data pattern. The files are then read back to verify data and length.

6.3.3.2.26 Test 26 - File Addressing

This test writes multiple files of increasing block sizes and different data patterns. The files are then read back to verify data length and contents.

6.3.3.2.27 Test 27 - Append

This test writes multiple blocks with unique data patterns. It then rewinds the tape and appends another unique block to the data. It rewinds the tape again, and reads back and verifies the data.

6.3.3.2.28 Test 28 - Streaming Write/Read

This test verifies the subsystem's ability to write the entire tape medium without repositioning. When EOT is detected, the tape is rewound. The data is then read back and verified.

6.3.3.2.29 Test 29 - Read Reverse

This test writes multiple patterns to tape, then reads each pattern in reverse to verify the data. If this test fails, the Read Reverse command does not work.

6.3.3.2.30 Test 30 - Odd-byte Fixed Block

This test does repeated writes and reads with an odd number of bytes as the block length. The data is compared to verify that an odd byte data transfer works properly.

6.3.3.2.31 Test 31 - Variable Block

This test verifies the functioning of variable block mode. The test performs repeated writes and reads, and compares the data and data lengths.

6.3.3.2.32 Test 32 - Bus-master Parity Error

This test forces a parity error to occur during the DMA phase with the adapter as bus master. Failure of this test indicates that the CMB is unable to detect and report a memory parity error during DMA with the adapter as bus master.

6.3.3.2.33 Test 33 - Missing Tape

This test requires operator intervention. It instructs the operator to remove the tape. It then attempts to read the tape, and checks for a unit off-line error. The operator is then instructed to reload the tape.

6.3.3.2.34 Test 34 - Write-Protected Tape

This test requires operator intervention. It displays a prompt instructing the operator to remove the write ring from the tape reel, and reload the tape. A write operation is then attempted, and the return status is checked for write-protect status. The operator is then instructed to return the write ring and reload the tape. Again a write operation is attempted, this time verifying that the tape can be written.

6.3.3.2.35 Test 35 - Streaming Write Only

This test requires operator intervention. It is to be used in conjunction with test 35 to verify media interchangability between drives and subsystems.

The test writes to the entire medium until EOT, and is then rewound. The tape may then be removed.

6.3.3.2.36 Test 36 - Streaming Read

This test requires operator intervention. It is to be used in conjunction with test 34 to verify media interchangability between drives and subsystems.

The test reads the data and compares it to the known data pattern of the streaming tape.

6.3.3.3 MTS LOGIC TEST COMMANDS

The following commands can be entered directly at the MTS test prompt to perform operations or set test parameters. These commands are in addition to the general diagnostic executive commands.

CLEARLOG

This command clears all current error counters for the logic test.

DICTIONARY (R,S,C,E,P,K,B)

This command displays a dictionary of definitions to help the operator interpret error messages and statistics. The options are:

R	Register definitions
S	Adapter Status register bit definitions
C	Control register bit definitions
E	Returned Error codes
P	Phase bit definitions
K	Sense Key meanings
B	Status Byte bit definitions

DUMP (P,R,W,S,C,M)

This command displays the specified buffer(s). The buffers are:

P	Parameter buffer
R	Read buffer
W	Write buffer
S	Status block
C	Command block
M	Memory locations

ENABLE (I,D,B)

This command selects modes for the test execution. Options are set by specifying the parameter, and reset by preceding the parameter with a minus sign (-). The options are:

I	Interrupt enable
D	DMA data transfer enable
B	Buffered I/O enable

LOGOUT

This command displays the current error counters for the MTS logic test.

QUICK

This command selects a subset of the logic tests in order to test the functionality of the subsystem quickly. The tests include all of the adapter tests, plus several data transfer tests. This command should not be used for fault isolation, but only to verify operation of the subsystem.

RUNIN

This command causes the logic test to begin a continuous loop of the currently selected tests. A logout report is displayed following each loop. The test continues until interrupted by a CTRL + C.

SPEED {*n*}

This command specifies the drive speed during the tests. A mode select is issued to affect the speed change. The options are:

0	High speed (default)
1	Low speed
2	High speed
3	Automatic speed select

6.3.3.4 ERROR MESSAGES

Most errors returned are self-explanatory.

Errors detected during a command handshake with the tape controller return detailed register information. The information bytes are:

Return Status - internal code indicating the nature of the problem detected.

Bus Phases - status bits indicating the bus phases that have been completed.

Status Byte - the status byte as returned by the tape controller during the status phase.

Message Byte - the message byte as returned by the tape controller during the message phase.

Status Register - the value of the adapter status register at the time the error was detected.

Error Code - the request status sense key detailing the drive condition.

Use the **DICTIONARY** command for more information on each information byte.

6.3.4 Parallel Interface/Timer Logic Test

The PI/T test verifies the basic functionality of the Motorola MC68230 parallel interface/timer I.C., the associated parallel printer that the chip drives, and the drivers/receivers that link these two devices.

The PI/T Logic Test consists of 10 subtests. These tests begin by checking the functionality of registers in the MC68230, and then build to testing the operation of drivers and the parallel printer.

Tests 7 through 10 require operator intervention. Tests 7 and 8 require a parallel loopback plug (MBF part number 907588-001). Tests 9 and 10 require a parallel printer with Centronics interface to be connected to the parallel port. A prompt is displayed prior to each test reminding the operator to connect the proper device.

6.3.4.1 Loading and Running the PI/T Logic Test

The PI/T logic test is loaded and run in the usual manner for logic tests. At either the diagnostic executive prompt, or at the prompt for another logic test, enter:

```
load pit
```

When the test is loaded, it runs an initialization routine, as described in 6.3.4.2.

To run the logic tests in normal mode, enter:

```
run
```

Refer to the command descriptions in Section 6.3.4.4 for additional test run options.

6.3.4.2 PI/T Test Initialization

Because of the nature of the test, if the PI/T test has been loaded with either the CLOCK or LP options selected, the program will deselect these options for the duration of the test.

The test also sets the default line count for printers to 80.

6.3.4.3 PI/T Test Descriptions

6.3.4.3.1 Test 1 - Register Addressing

This test verifies that all 23 registers of the PI/T chip can be addressed without causing a bus-timeout error. This is done by executing a read cycle for each register.

If a timeout error occurs, an error message is displayed indicating the register that failed.

6.3.4.3.2 Test 2 - Register Read/Write

This test verifies that each bit-wise writable and readable PI/T register can be written with a data pattern and then read back. The test is run only on registers that can be both written and read.

The test initially writes a 55 hex to each register, then reads back each register and verifies the data. If the pass is successful, the test inverts the data and runs another pass.

Any data miscompares are reported in error messages

6.3.4.3.3 Test 3 - Unique Register Address

This test verifies that each writable and readable PI/T register can be uniquely addressed with respect to the other registers. Each register is written using its address as data. The data is then read back to verify that its data is the same as its address.

Any data miscompares are reported in error messages

6.3.4.3.4 Test 4 - Reset Condition Status

This test verifies that a bus reset clears all PI/T registers that should be cleared. Each register that the reset should clear is initially written with AA hex. The reset is then executed, and the registers are read back to verify that they have cleared.

Any registers that fail to clear are reported in error messages.

NOTE

The RESET used in tests 4 and 5 is a system reset, and affects components other than the PI/T chip. In particular, this affects the MCS drive with a cartridge inserted. If the test is to be repeated several times, the cartridge should be removed, or the drive powered off, to avoid damage to the MCS.

6.3.4.3.5 Test 5 - Counter Zero Detect Status

This test checks the timer portion of the PI/T chip. It checks the ability of the chip to detect a countdown to zero condition.

The timer pre-load registers are loaded with 0000FF hex. The timer is then started, and the timer registers are decremented. A software time is then started, using a time delay much longer than the delay in the PI/T timer. The software timer waits for either a bit set in the timer status register, signifying a zero countdown detected status, or will time out. If the software timeout occurs, an error message is displayed reporting the condition.

If the bit is set in the timer status register, a bus reset is issued to verify that the bit can be cleared.

6.3.4.3.6 Test 6 - Timer Countdown Interrupt

This test is identical to the first part of test 5, except that prior to beginning timer countdown, the CPU interrupt priority is lowered to level zero. Upon countdown, the occurrence of an interrupt is verified.

If no interrupt is detected, an error message reports the condition.

6.3.4.3.7 Test 7 - External Wraparound

This test requires operator intervention. An external loopback plug must be attached for the test to run properly. Before the test runs, a message is displayed instructing the operator to install the plug.

This test verifies the PI/T chip's ability to send and receive parallel data. The PI/T is programmed with port A as output and port B as input. Using the loopback plug, a byte of data is sent out port A and read on port B. The data is verified, and any miscompares are reported in an error message.

6.3.4.3.8 Test 8 - External Wraparound Interrupt

This test requires operator intervention. An external loopback plug must be attached for the test to run properly. Before the test runs, a message is displayed instructing the operator to install the plug.

This test is identical to test 7, except that interrupts are enabled and the CPU priority is lowered to zero prior to data transfer. Upon completion, the occurrence of an interrupt is verified. If an interrupt is not detected, an error message reports the failure.

6.3.4.3.9 Test 9 - Printer

This test requires operator intervention. In order for this test to run properly, a parallel printer must be connected to the parallel I/O port.

This test verifies that the PI/T can function as a parallel printer controller. The chip is programmed for mode 0, pulsed submode. Next, all printer status bits are checked to verify that the printer is on-line, selected, loaded with paper and not busy. If all of these conditions are met, printing begins. All printable ASCII characters are printed, with each line filled with a single character.

The number of characters per line may be set using the CHAR_CNT command (refer to section 6.3.4.3).

6.3.4.3.10 Test 10 - Printer

This test requires operator intervention. In order for this test to run properly, a parallel printer must be connected to the parallel I/O port.

This test is similar to test 9, with the following exceptions. PI/T interrupts are enabled and CPU priority is lowered to level zero. All output is then performed using the interrupt service routine. Output to the printer is in the form of an ASCII "ripple pattern."

This test attempts to emulate the device-driver environment, as used by the operating system.

6.3.4.4 PI/T Logic Test Commands

The following commands can be entered directly at the PI/T test prompt to set test parameters. These commands are in addition to the general diagnostic executive commands.

CHAR_CNT {*x*}

This command sets and displays the number of characters per line for tests 9 and 10. If a parameter is not specified, the current number of characters per line is displayed. If a parameter is specified, the number of characters per line is set to this number. The value should be between 1 and 132. The default value is 80, set when the test is initialized.

INT_FACE {*x*}

This command sets and displays the interface to be used in printer tests 10 through 12. If a parameter is not specified, the current interface is displayed, followed by a menu of interface options. Press **ENTER** to leave the current interface unchanged. The interface options for *x* are:

1 = Centronics Compatible Interface (CCI)
2 = Data Products Compatible Interface (DPCI)
3 = General Purpose Parallel Interface (GPPI)

The initially selected interface is the Centronics Compatible Interface.

QUICK {*x*}

This command begins the quick-run version of the PI/T test, for fast verification of the functioning of the PI/T chip. The optional parameter *x*, where *x* = 1, causes tests 7 and 8 to be included in the run.

6.3.4.5 Error Messages

Refer to the individual test descriptions in Section 6.3.4.3 for the types of errors that are displayed by these tests.

6.3.5 SCC Logic Test

This test verifies the functionality of the Serial Communications Controller (SCC) on the CMB of the MAI 2000/3000 and related systems. The tests also allow the operator to isolate faults to the SCC, SCC clock sources, external receiver/drivers, modem cable, switched (or leased) lines or remote modem.

The SCC Logic Test consists of 11 subtests. These tests build in complexity and provide the user the ability to set up different loopback paths. In this way, failure at any point, in context of the success or failure of previous tests, is indicative of the specific fault.

6.3.5.1 Special Hardware Requirements

The SCC Logic Test has the following hardware requirements in addition to the general requirements for logic tests:

- To check both SCC channels, a 4-way or 8-way board must be installed, and the console moved to a 4/8-way port.
- Either a synchronous modem with cables or an external loopback plug (MBF part number 907641-001) must be used for the Bisync Modem Loop Test (test 11) and Bisync External Loop Test (test 10).

In addition, the following jumper configuration is required for external loopback and modem testing:

SCC Pin	RS232 Pin
Receive data	pin 3
Transmit data	pin 2
DTR	pin 6
Receive clock	pin 17
Transmit clock	pin 15

6.3.5.2 Special Software Requirements

All tests default to SCC channel B (corresponding to SCC port 1) only. If channel A is to be tested, the diagnostic must be configured to use an 8-way or 4-way port as the system console, using the diagnostic executive **CONSOLE** command.

The test may also be run on SCC port 0 if the system console is moved to SCC port 1.

6.3.5.3 Loading and Running the SCC Logic Test

The SCC logic test is loaded and run in the usual manner for logic tests. At either the diagnostic executive prompt, or at the prompt for another logic test, enter:

```
load scc
```

To run the logic tests in normal mode, enter:

```
run
```

Refer to the command descriptions in Section 6.3.5.5 for additional test run options.

6.3.5.4 SCC TEST DESCRIPTIONS

6.3.5.4.1 Test 1 - Read Register Test

This test attempts to read the control and data registers of the channel(s) being tested. If a read is not completed in the time allowed, a bus timeout occurs. The bus error, along with the register being read, are reported in an error message.

6.3.5.4.2 Test 2 - Write Register Test

This test attempts to write a 00 hex to the control and data registers of the channel(s) being tested. If the write is not completed in the time allowed, a bus timeout occurs. The bus error, along with the register being read, are reported in an error message.

6.3.5.4.3 Test 3 - Write/Read Register Test

This test writes to the interrupt vector register, the lower byte time constant register and the upper byte time constant register of the SCC. The registers are then read back and compared with the data written. The test performs one pass writing 55 hex, then repeats the test writing AA hex.

6.3.5.4.4 Test 4 - Async PCLK Loop

This test writes a 256 byte message using internal loopback on the channel(s) being tested. The message consists of the byte pattern 00 to FF hex. The test is performed at 9600, 4800 and 2400 baud. The message transmitted is compared with the message received, and any errors are reported.

When the test completes, it is repeated with the message cut in half. This continues until the message cannot be cut in half.

6.3.5.4.5 Test 5 - Async 3.6 MHZ Loop

This test is identical to test 4, except that the 3.6864 MHZ clock is used, and the message is transmitted at 19.2K, 9600, 4800 and 2400 baud.

6.3.5.4.6 Test 6 - Async X16 Clock

This test is identical to test 4, except that the x16 clock divide logic is used, and the message is transmitted at 19.2K, 9600, 4800, 2400, 1200 and 600 baud.

6.3.5.4.7 Test 7 - Async 3.6 MHZ Clock

This test loops the 256 byte message (see Test 4) using internal loopback at 9600 baud, and using the 3.6864 MHZ clock. The duration of this transmission is timed using the PI/T. In this way, the 3.6 MHZ clock and PI/T are used to check each others accuracy.

6.3.5.4.8 Test 8 - Split Baud

This test uses internal loopback to loopback 35 characters with the transmitter sending at 1200 baud and the receiver receiving at 2400 baud. The receiver uses the PCLK as its baud rate generator while the transmitter uses the x16 clock.

6.3.5.4.9 Test 9 - Bisync with CRC

This test loops back 256 bytes using the internal loopback at 9600 baud. The transmission is performed using bisync protocol with CRC enabled. The CRC bit is checked for proper updating.

6.3.5.4.10 Test 10 - Bisync External Loop

This test requires operator intervention, and must have the loopback cable installed.

This test verifies the correct operation of the external receiver/drivers. The DTR line is complemented every time the PI/T interrupts, thus supplying the external clock since DTR loops back to RCLK and TCLK.

6.3.5.4.11 Test 11 - Bisync Modem Loop

This test requires operator intervention, and must have a synchronous modem. The modem must be set in either analog loopback mode or have a connection with a remote modem that is in digital loopback mode (refer to the documentation for the modem involved).

Analog loopback testing allows fault isolation to the modem/controller. Digital loopback allows fault isolation to the switched or leased line/remote modem.

6.3.5.5 SCC Logic Test Commands

The following commands can be entered directly at the SCC test prompt to perform operations or set test parameters. These commands are in addition to the general diagnostic executive commands.

CHAN *x,y*

This command specifies which SCC ports are to be tested. The channels are 0 and 1, and either one or both may be specified. Whenever channel 0 is tested, the system console must be configured to use a 4-way or 8-way port. If CHAN is not used, testing defaults to port B (channel 1).

QUICK *x*

This command begins the quick run of the diagnostic. If an argument is not specified, only tests 1-9 are run. If a non-zero argument is used, test 10 is included.

MODEM

This command tests for the presence of the DTR signal active from a possibly attached modem. The DTR signal is indicated in the DMB status. An error is displayed if DTR is inactive.

PLUGTEST

This command tests for the presence of the loopback connector. The presence is determined by reading the CMB status, and testing for the ability to control the DTR signal.

6.3.5.6 Error Messages

Errors detected by the logic tests are reported by the following error messages. Additional information may also be included, giving further details of the error.

- TIMED OUT WAITING FOR RX TO COMPLETE

  The received never received the message. The problem may be in the transmitter or in the loopback path. If in the transmitter, then either the SCC is defective or it is not receiving a clock. If the problem is in the loopback path, it may be the remote or local modem, cables, telephone lines, external plug or internal loopback. Use other tests to isolate the fault.

- SOFTWARE ERROR

  This message is displayed when an illegal parameter is passed to a routine. Either a software bug exists or memory has been corrupted.

- CMB STATUS ERROR WITH DTR OFF

  Verify that the external plug is inserted correctly and that the jumpers are set correctly.

- CMB STATUS ERROR WITH DTR ON

  Verify that the external plug is inserted correctly and that the jumpers are set correctly.

- BUFFER MISCOMPARE

  The receive buffer does not match the expected data.

- UNEXPECTED INTERRUPT

  If this message scrolls the screen, a line is probably stuck.

- TOO MANY TX INTERRUPTS

  The transmit interrupt could not be reset. If write and read SCC register tests have passed, the SCC is probably defective.

- PARITY ERROR

  Parity error detected by SCC. If internal loopback is being used, then the SCC is probably defective; otherwise, the loopback path is suspect.

- RX OVERRUN

  The SCC detected a receiver overrun. Usually this is caused by inaccurate clocks. If the clock tests passed, the SCC is probably faulty.

- FRAMING ERROR

  The SCC detected a framing error. If most other loopback tests pass, then the loopback path for the test running is suspect.

- TOO MANY RX INTERRUPTS

  The receiver interrupt failed to clear after receiving the entire message. If read/write SCC register tests have passed, then the SCC internal logic is suspect.

- CRC ERROR

  If the clock tests have passed, then the SCC or loopback path is suspect.

- BUS ERROR WHILE READING DATA REGISTER

  DTACK was not returned.

- BUS ERROR WHILE READING CONTROL REGISTER

  DTACK was not returned.

- BUS ERROR WHILE WRITING DATA REGISTER

  DTACK was not returned.

- BUS ERROR WHILE WRITING CONTROL REGISTER

  DTACK was not returned.

- REGISTER READ-COMPARE ERROR

  Bad SCC or interface logic between SCC and CMB.

- CLOCK EXCEEDED 10% ERROR

  The clock is too inaccurate for safe communication.

- RX CHARACTER NOT AVAILABLE FOR RX INTERRUPT

  Unexpected receiver interrupt or defective SCC.

6.3.6 Four-Way Logic Test

The Four-way Logic Test checks the hardware and firmware of the 4-way board. It is designed to run with a minimum of user intervention. If a 4-way port is configured as the system console port, this board is not tested.

The test begins by initializing buffers and verifying its own ability to communicate with the 4-way boards on the system. From there the test builds in complexity, in order to verify the ability of the boards to operate in an operating system environment.

The four-way logic test consists of 21 subtests, which are described in Section 6.3.6.3.

Tests 5, 6 and 18 require a special loopback cable. This cable is available from MBF as part number 907530.

6.3.6.1 Loading and Running the 4-Way Logic Test

The 4-way logic test is loaded and run in the usual manner for logic tests. At either the diagnostic executive prompt, or at the prompt for another logic test, enter:

```
load fway
```

When the test is loaded, it runs an initialization routine. First it determines if a 4-way port is the system console port. If so, then it reports that this board will not be tested.

The routine then determines how many 4-way boards are on the system available for testing. If there are none, it reports:

```
REQUIRED DEVICE MISSING!
```

If boards are found, the routine reads 4-way status to verify that self-test has been completed, and to determine if errors occurred during the self-test. Any errors are reported on the screen.

To run the logic tests in normal mode, enter:

```
run
```

Refer to the command descriptions in Section 6.3.6.4 for additional test run options.

6.3.6.2 Test Initialization

When the test is loaded, it runs an initialization routine. First it determines if a 4-way port is the system console port. If so, then it reports that this board will not be tested. The routine then determines how many 4-way boards are on the system available for testing.

If boards are found, the routine reads 4-way status to verify that self-test has been completed, and to determine if errors occurred during the self-test. Any errors are reported on the screen.

4-way boards expect base addresses for the command block and interrupt vectors. The initialization routine writes these addresses to the appropriate ports, and tests for a passing response by the board status register and the transmit status register.

Finally, a data buffer area of 512 bytes is reserved for use during transmission tests.

6.3.6.3 4-WAY TEST DESCRIPTIONS

6.3.6.3.1 Test 1 - Primitive Instructions

This test verifies communication with the 4-way registers.

The test executes one operation of the following:

1. Read from the transmit status register
2. Write 55 hex to the instruction register
3. Read from the data out register
4. Write 55 hex to the data in register

6.3.6.3.2 Test 2 - Software Reset

This test issues a software reset command to each board. After the reset is issued, the base command block address and base interrupt vector are passed to the boards.

6.3.6.3.3 Test 3 - Command Status

This test passes an illegal command code, and checks that the 4-way sets the appropriate bad command status byte in the command block.

6.3.6.3.4 Test 4 - Configuration and Read Status

This test uses the 4-way Configure command to change SCC write registers. A 55 hex is written to write registers 12 and 13. The test then executes the 4-way Status command. Read registers 12 and 13 are then examined for 55 hex to verify both the configure and status commands. The default SCC configuration is then restored.

6.3.6.3.5 Test 5 - Remote Loopback Test 1

This test requires loop back connectors installed from port 0 to port 1, and from port 2 to port 3, on each board. If the connectors are already installed, the test begins immediately.

If connectors are not installed, the program prompts you to do so. Press **ENTER** when the connectors are installed, or **A** to abort the test.

The test transmits alternately from one port to the other, using predetermined data patterns. Each port is activated one at a time, one board at a time. Received data is compared against transmitted data.

6.3.6.3.6 Test 6 - Remote Loopback Test 2

This test operates in the same way as does test 5, except that it executes an external loopback transmission on all ports simultaneously.

6.3.6.3.7 Test 7 - Local Loopback Test 1

This test configures the SCC ports for local loopback, and then transmits predetermined data patterns through each port, one port at a time, one board at a time. The received data is compared with the transmitted data.

6.3.6.3.8 Test 8 - Local Loopback Test 2

This test configures the SCC ports for local loopback, and then transmits predetermined data patterns through all ports simultaneously. The received data is compared with the transmitted data.

6.3.6.3.9 Test 9 - 7 Bit/8 Bit Mask

The 4-way usually strips bit 8 from all bytes received. This test disables this feature, and then checks for a non-zero eighth bit.

6.3.6.3.10 Test 10 - Increment Byte Count

This test prepares a data packet with each byte equal to the lower byte of its data address. One byte of the data packet is then transmitted on each port being tested, in local loopback. The byte count is then increased by one and another pass is run. This continues for byte counts 1-F. The byte count is then increased by 10 for counts 10 to F0, and additional passes are run. Finally, passes are run with byte counts of 100 hex and 200 hex. All ports are transmitting simultaneously for this test.

6.3.6.3.11 Test 11 - Single Byte Transfer

This test transmits an 80 byte buffer in local loopback mode, using the single byte transfer command. It checks for data comparison errors. The test loops through for 24 passes.

6.3.6.3.12 Test 12 - Address Exercise

This test checks each bit in the 4-way address drivers.

This test creates a 512 byte data packet at the low end of memory, with each byte equal to the lower byte of its address. The data is transmitted in internal loopback, and is checked for comparison errors. The base address is then shifted left by 1 and the test runs again. The cycle continues until the top of memory is reached.

6.3.6.3.13 Test 13 - Bus Error

This test checks the 4-way's ability to set the bus error flag for each port. It does this by forcing the 4-way to access a data packet in non-existent memory, causing a bus error while the 4-way is bus master.

This test aborts if 2 MB or more memory is present, since it cannot handle this much memory.

6.3.6.3.14 Test 14 - Memory Parity Error

This test checks that the CMB is able to return a parity error status.

Bad parity is injected at a data packet address. The command to transmit is given to port 0, and the 4-way start to access the data. An error is reported if the CMB does not report a parity error.

6.3.6.3.15 Test 15 - 4-Way Interrupt Inhibit

This test checks the 4-way interrupt inhibit command.

Interrupt handling is disabled, and then 100 bytes are transmitted in local loopback. The 4-way should buffer these bytes. If an interrupt occurs, the test flags this as an error. Interrupts are then enabled, and the test waits for all characters to transmit. Finally, the received and transmitted data are compared and any errors reported.

6.3.6.3.16 Test 16 - Port Reset of XOFF Condition

This test checks the functioning of the Port Reset command.

The command resets the X-off condition, if it exists; otherwise it clears the 4-way's receive and transmit buffers. The test configures the port at 19200 baud, then sends an X-off character to set the X-off condition. A Port Rest is then issued. All characters are then compared, and the baud is checked to make sure the reset did not reconfigure the port.

6.3.6.3.17 Test 17 - XOFF Enable/Disable With Port Reset

This test checks the 4-way X-off enable/disable feature.

The test disables the X-on/X-off feature, then transmits a data packet containing the X-off character and verifies that transmission is not stopped. It then enables X-on/-off, transmits the data packet again, and verifies that transmission is stopped. Two resets are then issued to clear the X-off and to clear the receive and transmit buffers.

6.3.6.3.18 Test 18 - DTR Hardflow Control

This test requires loopback connectors to be installed, as described for test 5. The test verifies the function of the 4-way when DTR hardflow control is enabled (the default mode).

The test transmits enough characters to fill the 4-way receive buffer to the 3/4 full point, and checks for an external status interrupt due to a change in CTS. Hardflow control is then disabled and the test repeated, expecting no CTS transition.

6.3.6.3.19 Test 19 - 127 Byte Buffer Overflow

This test checks the ability of the 4-way to detect data overflow.

With interrupts disabled, the test transmits 129 bytes in local mode to each port. Interrupts are then enabled, and the test checks that exactly 127 bytes are received.

6.3.6.3.20 Test 20 - Special Receive Conditions

This test checks for the 4-way's ability to detect parity and framing errors.

The test modifies all receivers for 5-bit data. Data is then transmitted from each port, forcing framing and parity errors. An error is reported if the 4-way does not report framing and parity errors.

6.3.6.3.21 Test 21 - Interrupt Buffering

This test checks the 4-way's ability to buffer interrupts as well as data.

The test disables interrupts. A packet is then transmitted in local mode, followed by a packet that forces a special receive condition. Another good packet is then transmitted. Interrupts are then enabled, and the test checks that receive interrupts and the special receive condition interrupts are received in the correct order.

6.3.6.4 FOUR-WAY LOGIC TEST COMMANDS

The following commands can be entered directly at the 4-way test prompt to perform operations or set test parameters. These commands are in addition to the general diagnostic executive commands.

Use the commands to investigate faults more fully, such as to check the data in buffers.

CLRLOG

This command clears the error log counters.

FWAY {*list*}

This command specifies the boards to test, overriding the default to test all boards. The argument is a list of boards to test, separated by spaces. For example, to test only boards one and three, enter FWAY 1 3.

LOGOUT

This command displays the latest error log.

QUICK {*x*}

This command executes a quick version of the 4-way Logic Test. If no argument is entered, tests 1, 3, 4, 7, 8,11, 12, 13 and 16 are run. If **QUICK 1** is entered, tests 1, 3, 5-8, 11-13 and 16-18 are run.

RESET *x* {,*y*}

This command sends a reset command. The x argument specifies the board. If the y argument is not specified, the board is reset. If the y argument is specified, it indicates the channel to be reset (0-3).

RUNNIN {*list*}

This command runs a random, asynchronous test of the 4-way intended as a data reliability test. The test runs continuously, until interrupted by **ESCAPE**. After each pass, the test prints an error log. Each pass consists of running once through each option specified. The tests are run in local loopback mode.

If the user specifies no options, all of the options are run. Specify options by number, separated by spaces. The options are:

1. Random Character Transmission. Random characters are transmitted on all ports, one 4-way board at a time.

2. Concurrent Transmission. All ports on all boards are activated simultaneously.

3. Random Board, Random Port Transmission. A board is selected at random, and ports are activated on a binary increment pattern. Bits 0-3 represent ports 0-3. For each pass, a byte value is incremented by one, from 01 to 0F. For each pass, if a bit value is 1, that port is active for that pass. This tests all combinations of ports on the board.

4. Concurrent Transmission with Asynchronous Start Times. All ports on all boards are activate simultaneously, but the task for each runs independently of others. When a port's task is complete, and all characters received, the data is compared and the task is restarted.

STATUS *x*

This command runs the status routine in 4-way firmware. It returns the contents of the SCC read registers, transmit status byte, and the board status byte, where *x* is the board number.

6.3.6.4 ERROR MESSAGES

The following messages may be reported by the 4-way Logic Test.

- Z80 busy - time-out

  This message indicates that the busy bit in the transmit status register did not clear in the allotted time.

- Parity Error

  A special receive interrupt occurred indicating a parity error.

- Receiver Overrun Error

  A special receive interrupt occurred indicating the receive buffer has overflowed.

- Framing Error

  A special receive interrupt occurred indicating a framing error.

- Unexpected Interrupt

  An interrupt vector number was given for a port that is not programmed to be active.

- Undetermined Receive Error

  A special receive interrupt occurred, but software could not interpret the status byte.

- Command Complete Interrupt Not Received - time out

  The 4-way failed to complete its task or send the interrupt.

- External Status Interrupt

  An external status interrupt was received.

- Read Register not equal to Write Register

  The values written to the SCC in test 2 is not the same as the values read back into the read registers.

- Error in command execution flagged by 4-way

  The test attempted to send a command to the 4-way board, but the 4-way rejected it as an invalid command.

- Time-out on Self-test

  The self-test failed to complete in the time allotted by the initialization routine.

- Fatal Error flagged by Self-test!

  The self-test determined that the specified board is completely inoperable.

- ROM Failure

  This is a fatal error. The self-test detected an error in the 4-way ROM performance.

- DMA Bus Failure

  This is a fatal error. The self-test detected a failure in the 4-way DMA logic.

- RAM Failure

  The self-test detected a data compare error in a portion of the 4-way board's RAM.

- SCC Failure

  The self-test determined that the SCC specified has become inoperable.

- No characters received on any port

  The test timed out waiting to receive data through a 4-way port.

- Time-out on receive character

  At least one character was received, but the test timed out waiting for the next expected character.

- Data Compare Error

  The data in the transmit and receive buffers are not the same.

- No parity errors reported by 4-way

  The controller failed to report forced parity errors.

- No framing errors reported by 4-way

  The controller failed to report forced framing errors.

- Bus Error on DMA write to CMB

  The 4-way detected a bus error while it was bus master.

- Bus Error on read from status register

  A bus error occurred while reading the 4-way status register.

- Bus Error on write to instruction register

  A bus error occurred while writing the 4-way instruction register.

- Bus Error on read from data register

  A bus error occurred while reading the 4-way data register.

- Bus Error on write to data register

  A bus error occurred while writing the 4-way data register.

- Command Status incorrect

  The logic test issued a command known to be good or bad, but the 4-way did not respond with the correct status.

- Bus Error Not Flagged by 4-way

  The test forced a bus error condition while the 4-way was bus master, but the 4-way did not flag the error.

- XOFF Not Recognized by 4-way

  An X-off character was sent to a port, but the port did not stop transmitting.

- XOFF Not Disabled by 4-way

  The X-off disable command did not execute properly.

- Interrupts Not Inhibited by 4-way

  The interrupts disable command did not execute properly.

- 8th Bit Masked Off While in 8-Bit Mode

  The 4-way continued to mask out the 8th bit on all received characters after the mask feature was disabled.

- 8th Bit Not Masked Off While in 7-Bit Mode

  The 8-bit mask feature was enabled but not working.

- CTS Drop Detected While Hardflow Control Disabled

  With DTR flow control disabled, a CTS signal drop occurred.

- No CTS Drop Detected While Hardflow Control Enabled

  With DTR flow control enabled, no CTS signal drop occurred.

- Port Configuration Changed After Port Reset

  The SCC's were configured to 19200 baud, but this changed after a port reset

- Buffered Interrupts Not Received in Correct Order

  While interrupts were disabled, interrupts were not buffered properly. When interrupts were enabled again, the order of the data and interrupts were changed.

- Memory Parity Error Not Flagged by CMB

  A memory parity error was forced, but not detected by the CMB.

6.3.6.5 ERROR MESSAGE HEADERS

The following headers may be displayed with the error messages, giving additional information about the error.

- Transmit status - The contents of the transmit status register.
- Rx Port - The contents of the receive buffer.
- Tx Port - The contents of the transmit buffer.
- Events - The number of times an error occurred.
- Bytes - The number of bytes in the data packet that an attempt was made to transmit.
- Board - The number of the board on which the error occurred.
- Received, Sent - The data that was received and the data that was sent.
- Cmnd - The code of the 4-way command that was executed. The codes are as follow:

Command	Description
1	Configure SCC Registers
2	Transmit Data Packet
3	DMA SCC Read Registers
4	Return 4-way Registers to Default Values
5	Change X-on Character
6	Change X-off Character
7	Transmit Single Byte
8	X-on Enable/Disable
9	Hard Flow Control (DTR/CTS) Enable/Disable
A	7-bit/8-bit Mode

- Err Code - The type of error that occurred. Error Codes are:

Code	Description
1	4-way RAM Failure
2	Command Unrecognized
3	Bus Error during 4-way bus mastership

6.3.7 Eight-Way Logic Test

The Eight-way Logic Test checks the hardware and firmware of the 8-way board. It is designed to run with a minimum of user intervention. If an 8-way port is configured as the system console port, this board is not tested.

The test begins by initializing buffers and verifying its own ability to communicate with the 8-way boards on the system. From there the test builds in complexity, in order to verify the ability of the boards to operate in an operating system environment.

The eight-way logic test consists of 22 subtests, which are described in section 6.3.7.2.

Tests 18 through 22 require a special loopback cable. This cable is available from MBF as part number 907530.

6.3.7.1 Loading and Running the 8-Way Logic Test

The 8-way logic test is loaded and run in the usual manner for logic tests. At either the diagnostic executive prompt, or at the prompt for another logic test, enter:

```
load eway
```

When the test is loaded, it runs an initialization routine. First it determines if an 8-way port is the system console port. If so, then it reports that this board will not be tested.

The routine then determines how many 8-way boards are on the system available for testing. If there are none, it reports:

```
REQUIERD DEVICE MISSING
```

If boards are found, the numbers of those boards is reported.

To run the logic tests in normal mode, enter:

```
run
```

Refer to the command descriptions in Section 6.3.7.3 for additional test run options.

6.3.7.2 8-Way Test Descriptions

6.3.7.2.1 Test 1 - Primitive Instructions

This test verifies communication with the 8-way registers. Where possible, read data is verified.

6.3.7.2.2 Test 2 - Software Reset

This test issues a software reset command to each board. After the reset is issued, the base command block address and base interrupt vector are loaded.

6.3.7.2.3 Test 3 - Command Status

This test passes an undefined indirect and direct command to the board, and checks that the 8-way sets the appropriate bad command status byte in the command block.

6.3.7.2.4 Test 4 - Configuration and Read Status

This test uses the 8-way Configure command to change SCC write registers. The test then executes the STATUS command for each port on the board. The registers are then examined to verify that the SCC is addressable. The default SCC configuration is then restored.

6.3.7.2.5 Test 5 - Local Loopback: All Ports Simultaneously

The test configures all ports on the board for local loopback. Test data patterns are transmitted through all ports at once. After all ports are finished, the received data is compared to the original data.

6.3.7.2.6 Test 6 - Local Loopback: Each Port Sequentially

The test configures all ports on the board for local loopback. Test data patterns are transmitted through each port individually. After all ports are finished, the received data is compared to the original data.

6.3.7.2.7 Test 7 - Local Loopback: Each Port Individually

The test configures all ports on the board for local loopback. Test data patterns are transmitted through each port individually. After each port is finished, the received data is compared to the original data.

6.3.7.2.8 Test 8 - 7 Bit/8 Bit Mask

The 8-way usually strips bit 8 from all bytes received. This test disables this feature, and then checks for a non-zero eighth bit.

6.3.7.2.9 Test 9 - Increment Word Count

This test transmits, in local loopback, data packs of incrementing word counts are transmitted to all ports of the board. The received data is compared to the expected data. The word count increments from 1 to F hex by 1, and from 10 to C0 hex by 10.

6.3.7.2.10 Test 10 - Single Byte Transfer

This test transmits individual bytes in local loopback mode and verifies the data.

6.3.7.2.11 Test 11 - Address Exercise

This test creates the transmit buffer at various locations in memory from the end of the Diagnostic Executive to the top of memory. For each transmit buffer, the data is sent in local loopback, and the received data is compared to the source buffer.

6.3.7.2.12 Test 12 - Bus Error

This test causes the 8-way to access a transmit buffer that is beyond memory range, forcing a DMA mode bus error. The command status is then checked for validity.

6.3.7.2.13 Test 13 - Memory Parity Error

This test forces a memory parity error to occur during the source data DMA cycles to verify that errors are reported correctly while the 8-way is bus master.

6.3.7.2.14 Test 14 - 8-Way Interrupt Inhibit

This test verifies that 8-way interrupts are not received for a port that has had interrupts inhibited. All transmitted characters should be buffered by the 8-way until interrupts are enabled.

6.3.7.2.15 Test 15 - Port Reset of XOFF Condition

This test verifies the Port Reset command's ability to reset an existing X-off condition, or to clear the transmit and receive buffers for the port. The test transmits the X-off character within the data packet in local loopback mode. After verifying that the X-off did stop data transmission, a port reset is issued and all data is received.

6.3.7.2.16 Test 16 - XOFF Enable/Disable With Port Reset

This test checks the 8-way X-off enable/disable feature. The test disables the X-on/X-off flow control, then transmits a data packet containing the X-off character in local loopback and verifies that transmission is not stopped. It then enables X-on/X-off, transmits the data packet again, and verifies that transmission is stopped. Two resets are then issued to clear the X-off and to clear the receive and transmit buffers.

6.3.7.2.17 Test 17 - Special Receive Conditions

This test modifies all ports to transmit 7 bit data and receive 5 bit data. Transmitted data in local loopback mode should cause framing and parity errors for each port.

6.3.7.2.18 Test 18 - DTR Hardflow Control

This is a manual intervention test, and requires the loopback connectors to be installed. The test verifies the function of the 8-way ports when DTR hardflow control is enabled. CTS is checked on the port to verify that DTR has dropped when the buffers are full. DTR is then disabled and the test is repeated to verify that hardware flow control is not operating.

6.3.7.2.19 Test 19 - Interrupt Buffering

This is a manual intervention test, and requires the loopback connectors to be installed. The test checks the 8-way's ability to buffer interrupts as well as data. The test inhibits interrupts and sends small data packets to force special receive conditions in sequence with good data transmission. The order of interrupts is checked to verify that each occurs at the correct time.

6.3.7.2.20 Test 20 - Remote Loopback:- All Ports Simultaneously

This is a manual intervention test, and requires the loopback connectors to be installed. The test transmits data patterns through all ports at once. After all ports are finished, the received data is compared to the original data.

6.3.7.2.21 Test 21 - Remote Loopback: Each Port Sequentially

This is a manual intervention test, and requires the loopback connectors to be installed. The test transmits data patterns through each port individually. After all ports are finished, the received data is compared to the original data.

6.3.7.2.22 Test 22 - Remote Loopback: Each Port Individually

This is a manual intervention test, and requires the loopback connectors to be installed. The test transmits data patterns through each port individually. After each port is finished, the received data is compared to the original data.

6.3.7.3 EIGHT-WAY LOGIC TEST COMMANDS

The following commands can be entered directly at the 8-way test prompt to perform operations or set test parameters. These commands are in addition to the general diagnostic executive commands.

Use the commands to investigate faults more fully, such as to check the data in buffers.

EWAY {*list*}

This command specifies the boards to test, overriding the default to test all boards. The argument is a list of boards to test, separated by spaces. For example, to test only boards one and three, enter EWAY 1 3.

IOCB *x,y*

This command displays the current values of the command block for the specified board *x* and port *y*.

QUICK

This command executes a quick version of the 8-way Logic Test as a confidence check. A quick run should not be used for fault isolation.

RBUFFER *x,y*

This command displays the receive buffer contents for the specified board *x* and port *y*.

RESET {*x*}

This command sends a reset command. The argument specifies the board to be reset (0-5). The command also reloads the command block base address and the base interrupt vector.

RUNNIN {*list*}

This command runs a random, asynchronous test of the 8-way intended as a data reliability test. The test runs continuously, until interrupted by **ESCAPE**. After each pass, the test prints an error log. Each pass consists of running once through each option specified. The tests are run in local loopback mode.

Up to 19 tests may be specified. If the user specifies no options, all of the options are run. Specify options by number, separated by spaces. The options are:

0 Sequence through all tests. For each board, all the remaining options (1-5) are called.

1 Transmit all ports simultaneously. A data packet of random characters is transmitted to all ports of the board simultaneously.

2 Transmit all ports sequentially. A data packet of random characters is transmitted to each port of the board one at a time.

3 Transmit all ports individually. A data packet of random characters is transmitted to each port individually, waiting for all receive characters from the port.

4 Random board, random port transmit. A board is randomly selected and a random port is selected for a data packet of random characters to be transmitted.

5 Transmit all boards, all ports simultaneously. All ports on all boards are activated simultaneously. The test will wait for all receive characters on all ports of all boards.

STATUS *x*

This command returns the contents of the SCC read registers for each port on the board specified by the argument (0-5). The command uses the 8-way Read Status command.

XBUFFER

This command displays the transmit buffer contents.

6.3.8 LAN Logic Test

The LAN Logic Test verifies the functionality of the LAN subsystem, and aids in isolating faults to the host, network, LAN host interface logic, Corvus chip set and the LAN network interface. Fault isolation to the chip level is not available.

Fault detection concentrates on the host/controller interface and onboard logic, with minimal exercise of the network. To fault check all of the logic on a given LAN board, two boards are needed. These boards may be in the same or different hosts, but must be connected in either case.

Tests that use two or more controllers distinguish the master and slave controllers. Refer to the SLAVE command in 6.3.8.3 for information on selecting the slave controllers.

Refer to the system service manual for LAN controller board switch settings.

6.3.8.1 Loading and Running the LAN Logic Test

The LAN Logic Test is loaded and run in the usual manner for logic tests. At either the diagnostic executive prompt, or at the prompt for another logic test, enter:

```
load lan
```

To run the logic tests in normal mode, enter:

```
run
```

Refer to the command descriptions in Section 6.3.8.3 for additional test run options.

6.3.8.2 LAN Test Descriptions

6.3.8.2.1 Test 1 - Read Register Test

This test verifies the ability to read the read registers. Each read register is read ten times, and then the next register is read.

The usual fault during this test is a bus error causes by a bus timeout if the board failed to return DTACK.

6.3.8.2.2 Test 2 - Write Register Test

This test verifies the ability to write to the write registers. Each write register is written ten times, and then the next register is written.

The usual fault during this test is a bus error causes by a bus timeout if the board failed to return DTACK.

6.3.8.2.3 Test 3 - Write/Read Register Test

This test checks the control and vector registers. Each register is written and read back twice, comparing the data. The first pass uses data AA hex, and the second pass uses 55 hex.

6.3.8.2.4 Test 4 - DMA Polling Test

This test uses the WHO command to test DMA in both directions while running without interrupts in a polling environment.

After the CAR is written with the address of the command vector, the Corvus Chip set should DMA (DMA read by LAN) in the command vector and then DMA back (DMA write by LAN) the node address of the board under test.

6.3.8.2.5 Test 5 - DMA Interrupt Test

This test is the same as test 4, except that interrupts are used.

6.3.8.2.6 Test 6 - DMA Data Test

This test verifies that no data lines are stuck during DMA.

The test uses the Poke command to write a byte value to a RAM address internal to the Corvus Chip Set that is not overwritten by the Corvus firmware. It then uses the Peek command to recover the byte value. This procedure is repeated for every combination of bytes using interrupts.

6.3.8.2.7 Test 7 - Bus Error Test

This command forces a bus error while the LAN controller is bus master. This is done by executing a WHO command and writing the return code to nonexistent memory.

NOTE

The LAN board cannot address more that 2 MB. For the bus error test, the system must have less than 2MB of memory.

6.3.8.2.8 Test 8 - Ready Interrupt Test

This test checks the LAN controller's ability to process interrupts. The ready bits in the LAN are first checked for being set. Then the interrupt on ready bit of the control register is set with interrupts enabled. An interrupt is then executed and processed.

6.3.8.2.9 Test 9 - Interrupt Test

This test verifies that all interrupt vector addresses are functioning properly. This is done by causing interrupts, and verifying that the correct vector is used.

6.3.8.2.10 Test 10 - Address Test

This test uses the Who command repeatedly, with each iteration writing the one byte return code to a different memory location. This continues until all LAN controller address lines needed to access host memory have been tested in the on/off state.

6.3.8.2.11 Test 11 - Reset Test

This test is primarily a check of the Corvus Chip Set. The test overwrites the retry count stored in RAM internal to the chip set. A reset is then issued via the LAN control register and the location is checked for the correct retry count.

6.3.8.2.12 Test 12 - Checksum Test

This test adds every location in the Corvus Chip Set ROM to form an accumulated sum. The result should be zero, since Corvus stores a checksum in ROM.

6.3.8.2.13 Test 13 - Clock Test

This test checks the LAN controller's ability to transmit at 1 MHZ without the need for a second controller. The controller is programmed to broadcast a message of known length, while timing the duration of the broadcast. An error greater than +/- 5% is reported by an error message.

6.3.8.2.14 Test 14 - Parity Test

This test checks the controller's parity circuitry. Errors are injected into transmitted data to force parity errors. Correct data is also transmitted, and the parity is checked.

6.3.8.2.15 Test 15 - Send Test

This test sends a packet from the master controller to the slave that commands the slave to initialize socket 80 hex to receive a message of a specified length and pattern. The message is then transmitted. This is repeated for various lengths and data patterns.

6.3.8.2.16 Test 16 - Receive Test

This test initializes the master controller's socket 80 hex to receive a message of a specified length and pattern. A packet is then sent to the slave instructing it to send to the master a message of the specified length and pattern. This is repeated for the same lengths and patterns used in the Send Test.

6.3.8.2.17 Test 17 - CRC Test

This test checks the CRC logic by causing a CRC error during transmission from slave to master and checking for an automatic retransmission. The slave controller to transmits a known message to the master. At about the middle of the transmission the master disables the receiver briefly, causing a loss of data and then re-enabled to complete transmission. If the loss is detected at the end of transmission, a retransmission is begun automatically.

6.3.8.2.18 Test 18 - Overrun Test

This test checks the overrun logic on the LAN controller. During transmission, DMA is disabled for one byte time. If the overrun logic is working properly, the acknowledge should be repressed resulting in an automatic retransmission.

6.3.8.2.19 Test 19 - DMA Polling Scope Loop Test

This test requires operator intervention, and involves the use of an oscilloscope. Loop parameters are controlled by the switches on the LAN itself. One pass of the test involves the following:

1. If bit 2 of SW1 is ON, then reset the LAN controller.
2. Build command block for WHO command.
3. Write address of command block to LAN controller.
4. DMA in 4 bytes of WHO command.
5. Read node address register switches.
6. Write node address to CMB memory and turn on CINT bit.
7. If bit 3 of SW1 is ON and an error occurred, then report to CRT. If bit 1 is on, then exit test, else go back to step 1.

6.3.8.2.20 Test 20 - ECHO Scope Loop

This test requires operator intervention. The use of the test switches of SW1 are explained on the screen. (The switch number is not the same as the bit number.)

While the test is executing, the node address register is read to see if a reset should occur before each DMA, if message reporting is requested, and if the test should end.

The ECHO command is issued and the return code is checked on LAN 0. Looping occurs on the ECHO command being sent to the transmitting logic, received from the receiving logic, and comparing the return code. If an error occurs, it is reported.

To end the test, set SW1 position 1 to ON.

6.3.8.3 LAN TEST COMMANDS

The following commands can be entered directly at the LAN test prompt to perform operations or set test parameters. These commands are in addition to the general diagnostic executive commands.

CLRLOG

This command clears all error log entries.

CONFIG

This command causes the master to echo to every possible node, then to send a special packet to every node that responded to determine which are slaves. When this information has been gathered, the configuration of the network is output to the CRT.

EQUE

This command displays the last approximately 80 errors that occurred during the LAN test.

INIT

This command initializes the LAN and the test parameters.

LAN *x*

This command switches mastership when two LANs are in the same host. The parameter specifies which LAN (0 or 1) is the master.

LOGOUT

This command displays the current error log entries.

MONITOR *x*

This command controls the monitor display. Only one parameter may be specified, selected from the following:

O	Turns the monitor OFF (this is the default)
H	Causes a one-line header to be displayed
R	Causes the entire message to be displayed in hex

NAR

This command reads and displays the Node Address Register, giving the address of the LAN controller currently selected (see LAN command).

QUICK

This command executes a short test. It is used only to verify that a controller is working. It should not be used for fault isolation.

RUNIN

This command begins continuous execution of the selected tests.

SLAVE

This command activates slave mode, and selects a slave LAN controller if both controllers are not in the same host. This is necessary before any of the slave tests can be run. If both controllers are in the same host, the slave is assumed (but see the LAN command).

6.3.8.4 ERROR MESSAGES

The following errors may be reported by the LAN Logic Test.

- %Error for 1 Meg. Clock Exceeded

  The error in transmission rate is greater than +/-5%. The clock is not performing acceptably.

- Status Error

  The expected LAN status is different than reported in the status register. The observed and expected status are reported.

- Node Address Miscompare

  The node address in the return code does not match the contents of the node address register.

- Bus Error Interrupt From LAN Didn't Occur

  A bus error was forced with LAN as bus master, and interrupts were enabled, but the interrupt was not detected.

- Multiple Interrupts

  More interrupts were received than expected. If this message scrolls across the screen, either an interrupt line is stuck or the clear CINT/BINT bit(s) operation is not working.

- Ready Bit is Not ON

  The ready bit should be ON. The expected and observed status is displayed.

- Ready Interrupt Didn't Occur

  Interrupt on ready was enabled and the ready bit was ON, but the interrupt did not occur.

- PEEK Data Miscompare

  The data read by a PEEK does not match the data written by a POKE. This is performed on the RAM internal to the Corvus chip set.

- Incorrect Retry Count After Issuing Reset

  The Corvus chip set did not reset or did not reset correctly.

- Timed Out Waiting for Correct Status to Write CAR

  The ready and rdy bits of the status register were never ON at the same time, which is the criterion for writing the CAR.

- LAN Interrupt Used Wrong Vector

  The interrupt vector chosen by the LAN does not correspond to the last data byte used to initialize the VCR.

- Receive Buffer Miscompare

  The message just received does not match the pattern expected.

- ELANC Addressing Error

  The LAN controller wrote the return code to the wrong address during the address test.

- Time Out Waiting for RX Cmd

  The process timed out waiting for the initialize receive socket command to complete after writing CAR.

- Time Out Waiting for SEND Cmd

  The process timed out waiting for the initialize send command to complete after writing CAR.

- Time Out Waiting for WHO Cmd

  The process timed out waiting for the initialize who command to complete after writing CAR.

- Time Out Waiting for ECHO Cmd

  The process timed out waiting for the initialize echo command to complete after writing CAR.

- Time Out Waiting for INIT Cmd

  The process timed out waiting for the initialize init command to complete after writing CAR.

- Time Out Waiting for PEEK Cmd

  The process timed out waiting for the initialize peek command to complete after writing CAR.

- Time Out Waiting for POKE Cmd

  The process timed out waiting for the initialize poke command to complete after writing CAR.

- Time Out Waiting for END REC. Cmd

  The process timed out waiting for the initialize end receive command to complete after writing CAR.

- Programming Error

  Either there is a bug in the program or memory corruption occurred.

- Checksum Error

  The checksum was incorrect in the Checksum test.

- Invalid Socket Number in Command Vector

  An error is returned to the host by the Corvus chip set.

- Receive Socket in Use

  An attempt was made to initialize a socket which is already in use. The error is returned by the Corvus chip set to the host.

- Undefined Return Code

  The return code is not recognized.

- Message Not Acknowledged (Retry Count Exceeded)

  The set number of retries was exceeded without receiving an acceptable acknowledge from the target (slave).

- Message Data Portion Too Long for Rx Buffer

  The message just sent is too long for the target (slave) node receive buffer.

- Message was Sent to Uninitialized Socket

  The target (slave) node is present, but the receive socket is not initialized.

- Invalid Destination Node Number in Command Vector

  The node number is not in the range 0 to 63.

- Message was Sent Successfully After N Retries

  A valid acknowledge was received before the maximum number of retries was exceeded.

- Too Many Interrupts from LAN

  If this message scrolls across the screen, either the interrupt line is stuck or the interrupt cannot be cleared by a write to the CLRCINT/CLRBINT registers.

- Unexpected Interrupt

  An interrupt from the LAN controller occurred without a corresponding return code for a command that was pending.

- LAN Buserr Bit is ON

  A bus error occurred while the LAN controller was bus master.

- CPU Buserr While Reading Control Register

  A bus time out error occurred, probably because DTACK was not returned.

- CPU Buserr While Reading VCR Register

  A bus time out error occurred, probably because DTACK was not returned.

- CPU Buserr While Reading Status Register

  A bus time out error occurred, probably because DTACK was not returned.

- CPU Buserr While Reading Node Address Register

  A bus time out error occurred, probably because DTACK was not returned.

- CPU Buserr While Writing CTRL

  A bus time out error occurred, probably because DTACK was not returned.

- CPU Buserr While Writing VCR

  A bus time out error occurred, probably because DTACK was not returned.

- CPU Buserr While Writing CLRBINT

  A bus time out error occurred, probably because DTACK was not returned.

- CPU Buserr While Writing CLRCINT

  A bus time out error occurred, probably because DTACK was not returned.

- CPU Buserr While Writing CAR

  A bus time out error occurred, probably because DTACK was not returned.

- CPU Buserr While Writing CARH

  A bus time out error occurred, probably because DTACK was not returned.

- CPU Buserr While Writing CARM

  A bus time out error occurred, probably because DTACK was not returned.

- CPU Buserr While Writing CARL

  A bus time out error occurred, probably because DTACK was not returned.

- Timed Out Waiting for Packet

  The slave did not send message before the time out occurred.

- Probably CRC Checker Error

  An error occurred in the CRC test.

- PIT Failed to Reset

  A PI/T interrupt occurred after the PI/T was reset. See the Clock test.

- Garbage Packet Received by Slave

  The slave received an undefined packet.

- LAN Failed to Detect Overrun

  See the description of the Overrun Test.

6.3.9 Memory Logic Test

The Memory Logic Test checks the functionality of system memory on the MAI 2500/3000/4000 systems (the MAI 2000 systems do not use this test).

In case that a fault is suspected in only a portion of memory, the user may specify the area to test. All memory can be tested except in the range 0 to FFF hex, which is used by the diagnostic executive and the debugger.

Unlike most logic tests, this test also provides the feature of building command files. Refer to Section 6.3.9.5.

6.3.9.1 Loading and Running the Memory Logic Test

The Memory Logic Test is loaded and run in the usual manner for logic tests. At either the diagnostic executive prompt, or at the prompt for another logic test, enter:

```
load memory
```

To run the logic tests in normal mode, enter:

```
run
```

Refer to the command descriptions in Section 6.3.9.4 for additional test run options.

6.3.9.2 Memory Test Initialization

The Memory Test begins by autosizing the system memory. The memory found is shown is a display, for example (the actual display will differ):

```
0000000 - 00FFFFF=> FF
0100000 - 01FFFFF=> FF
0200000 - 02FFFFF=> FF
0300000 - 03FFFFF=> FF
0400000 - 04FFFFF=> 00
0500000 - 05FFFFF=> 00
0600000 - 06FFFFF=> 00
0700000 - 07FFFFF=> 00
0800000 - 08FFFFF=> 00
0900000 - 09FFFFF=> 00
0A00000 - 0AFFFFF=> 00
0B00000 - 0BFFFFF=> 00
0C00000 - 0CFFFFF=> 00
```

In the display, an "FF" indicates that the bank address range was found, and "00" indicates that it was not found.

The test pattern is set to 5555ABAB hex.

6.3.9.3 MEMORY TEST DESCRIPTIONS

6.3.9.3.1 Test 1 - Non-Existent Memory Access

This test attempts to access a memory location known to be outside the system's address bounds. This should cause a bus timeout error and trap. If the trap does not occur, the test reports an error.

6.3.9.3.2 Test 2 - ECC Memory Array Test

This test verifies the memory used by the ECC is valid. The ECC is first disabled, allowing direct access. Several patterns are then written to the ECC memory and verified. The test reports any errors that occur.

Following all test passes, ECC is enabled and the entire test range is written with a long word to establish valid checkwords for the memory range.

6.3.9.3.3 Test 3 - EDC Dynamic Checkword Test

This test verifies that every longword in the selected range can generate a single bit error. The test writes an inverted longword address to each longword in the range. As the longword is written, a single bit is inverted and written back to memory with the EDC OFF. This creates a single bit error in memory. The EDC is turned back ON, and the entire range is read 1 longword at a time. Each read should cause an interrupt.

6.3.9.3.4 Test 4 - March Test

This test verifies that there are no multiple addressing errors in the memory module.

The test consists of writing FF hex in each byte being checked, reading and verifying each for FF, and then writing 00 hex. After all bytes have been written with this pattern, each is read and verified to be 00 hex, and then rewritten with FF hex.

6.3.9.3.5 Test 5 - Sliding Ones Test

This test verifies that the bit data lines are not shorted at any location. It can also reveal grounding and layout problems.

The test writes 0001 hex in each word being checked. Each word is then read and verified. This cycle is then repeated for the following patterns:

0002	0004	0008	0010	0020	0040
0080	0100	0200	0400	0800	1000
2000	4000	8000			

6.3.9.3.6 Test 6 - Sliding Zeroes Test

This test is identical to test 5, except that these patterns are used:

FFFE	FFFD	FFFB	FFF7	FFEF	FFDF
FFBF	FF7F	FEFF	FFDF	FFBF	FF7F
FWFF	FDFF	FBFF	F7FF	EFFF	DFFF
BFFF	7FFF				

6.3.9.3.7 Test 7 - Ping-Pong Test

This test verifies that there are no random access problems

The test writes 00 hex in all odd addresses and FF hex in all even addresses. The program then reads and verifies the low address byte, the high address byte, the low address byte +1, the high address byte -1, and so on until all memory locations are verified.

6.3.9.3.8 Test 8 - Complement Ping-Pong Test

This test is identical to test 7, except that it writes FF hex in all odd address and 00 hex in all even addresses.

6.3.9.3.9 Test 9 - Worst Case Data Movement

This test verifies the MOVEM (move multiple registers) instruction.

The test writes to all eight data registers the following data patterns (all in hex):

0F0F0F0F	FFFFFFFF	FFAAFFAA	AAFFAAFF	00000000
99999999	55555555	AAAAAAAA		

Starting at high memory, these data patterns are written to contiguous memory locations. Then the locations are read back and compared to verify data integrity.

6.3.9.3.10 Test 10 - Address Test

This test is functionally the same as the address test performed during system self-test. If the self-test reports a "mema" error, this test should be used to pinpoint the error location. All data mismatch errors are logged to the console.

The test writes the address of each word within the selected range to the words being tested (only the least significant word of the address can be written to the word). The words written are then read and verified. Next, the complement of the address is written to each word and verified.

6.3.9.3.11 Test 11 - Galloping LS Word

This test verifies the memory can run all possible address transitions. This test is run on each 1K memory segment.

The test writes a background pattern (the most significant word of the test pattern) to all words being checked. A testword (the least significant word of the test pattern) is then written into the first word. This sequence is then performed:

Read location two, read and verify location one. Read location three, read and verify location one. This is repeated until every pair of transitions is checked.

The testword is then moved to the second location and the sequence repeated checking all locations with the second location. This is repeated, incrementing the testword location, until all locations have been the testword.

6.3.9.3.12 Test 12 - Galloping MS Word

This test is identical to test 11 except that the background pattern is the least significant word of the test pattern and the testword is the most significant word of the test pattern.

6.3.9.3.13 Test 13 - Galloping LS Word Write Recovery

This test verifies that a memory location can be read with no errors as all other locations are written with the most significant and least significant word of the test pattern. The following sequence is run on each 1K memory segment:

A background pattern (the most significant word of the test pattern) is written to all words being checked. The test word (least significant word of the test pattern) is then written to location two and location one is then checked. Location two is then written with the background word and location one is checked. Location three is written with the test word and location one is checked, then location three is written with the background word and location one is checked. This repeats until all locations have been written with both the background and test words and location one checked. The sequence is then repeated, checking locations two, three, four, and so on until all locations have been checked.

6.3.9.3.14 Test 14 - Galloping MS Word Write Recovery

This test is identical to test 13, except that the least significant word of the test pattern is the test word and the most significant word is the background pattern.

6.3.9.4 MEMORY TEST COMMANDS

The following commands can be entered directly at the Memory test prompt to perform operations or set test parameters. They can also be included in a command file (refer to section 6.3.9.5). These commands are in addition to the general diagnostic executive commands.

DELAY *x*

This command defines how many micro seconds a test or command should wait before reading the data just written to memory. Each unit of delay is 2.625 micro seconds.

EDC *x*

This command (Error Detection/Correction) sets or resets a flag for the RSYN and WCBR commands. This affects the **RSYN** and **WCBR** commands only. With no parameters, the current state of the EDC flag is displayed. The parameters are:

0	Turns the EDC flag OFF
1	Turns the EDC flag ON

INIT

This command reinitializes the test, including autosizing memory, setting the test pattern to 5555ABAB hex, and selecting the test range as 1000 to the end of memory.

MODULES *x,y*

This command selects the memory modules (megabyte boards) to be tested. If only one argument is entered, then only that board will be tested. If both arguments are given, then the first is the lowest board to be tested and the second is the highest; the range between and including these boards becomes the test range.

PATTERN *hhhhhhhh*

This command specifies the pattern to be used in write/read tests. It is specified as four hexadecimal bytes. If the pattern is too long, only the last four bytes are used. If the pattern is too short, it is zero filled on the right.

RANGE *x,y*

This command selects the address range to be tested. If no argument is given, then the test range previously specified (by MODULES, BANKS, INIT or RANGE) is displayed. The test range must be valid, not including any of the range 0 to FFF hex or above the end of memory.

QUICK

This command runs tests 1-4 and 7-10 to test the range quickly.

RB (read byte)

This command reads each byte of memory in the current range. Memory is read one byte at a time, and is not verified.

RW (read word)

This command reads each word of memory in the current range, beginning at word boundaries. Memory is read one word (2 bytes) at a time, and is not verified.

RL (read long word)

This command reads each long word of memory in the current range, beginning at word boundaries. Memory is read one long word (4 bytes) at a time, and is not verified.

RSYN {*a*,*b*}

This command (Read Syndrome Register) displays the long word contents of memory from start address, *a*, to end address, *b*. If EDC is turned OFF, the syndrome byte for each long word of data is displayed with the memory data. With no parameters, **RSYN** displays the entire test range.

WB (write byte)

This command fills the address range with the current pattern, writing one byte at a time.

WW (write word)

This command fills the address range with the current pattern, starting at a word boundary, writing one word (2 bytes) at a time. If the range is not large enough to begin at a word boundary, an error message is displayed and the command aborts.

WL (write long word)

This command fills the address range with the current pattern, starting at a word boundary, writing the entire pattern (4 byte long word) with each write. If the range is not large enough to begin at a word boundary, an error message is displayed and the command aborts.

WRB (write/read byte)

This command alternately writes, reads each byte in the selected range. The most significant byte of the pattern is written to the first byte location, the least significant byte to the fourth byte location, then the most significant to the fifth, and so on. Data mismatch errors are logged.

WRW (write/read word)

This command is the same as **WRB**, except that it writes and reads one word (2 bytes) of the pattern at a time, starting at word boundaries.

WRL (write/read long word)

This command is the same as WRW, except that it writes and reads a long word (4 bytes) at a time.

WCBR *a,b,c,d*

This command (write to checkbit register) writes a specific checkword to ECC memory. It can also write a long word to memory. EDC must be OFF to write the checkword; otherwise the checkword is based on the data being written. The parameters are:

a	start address (long word)
b	end address (long word)
c	check word (byte)
d	data pattern (long word)

If no parameters are used, 0 is written to the entire memory test range. The combinations of parameters have these effects:

WCBR *a*	Writes 1 long word of 0 with checkword of 0 at address *a*.
WCBR *a,b*	Writes all longwords of 0 with checkword of 0 from address *a* to *b*.
WCBR *a,b,c*	Writes all longwords of 0 with checkword of *c* from address *a* to *b*
WCBR *a,b,c,d*	Writes all longword of *d* with checkword of c from address *a* to *b*.

6.3.9.5 BUILDING COMMAND FILES

The commands described in 6.3.9.4 can be used to build command files. A command file is begun with the CMD_FILE command. This changes the prompt, at which any of the commands may be entered.

The command file is ended by pressing ENTER alone at the prompt. It is then executed by the XCMD command.

An example of a command file is as follows (the comments enclosed in braces are not allowed in the actual command file):

```
<memory>CMD_FILE                    {start to build the command file}
<cmd>RANGE 1000,1000                {select byte at address 1000}
<cmd>PATTERN FFFFFFFF               {select pattern}
<cmd>WRB                            {write/read/verify FF at address 1000}
<cmd>RANGE 2FFF,2FFF                {select a higher byte}
<cmd>PATTERN 0                      {select a new pattern}
<cmd>XCMD                           {repeat above endlessly}
<cmd><CR>
<memory>
```

The command file is only temporary, and cannot be saved.

6.3.9.6 SCOPE LOOP COMMANDS

The following memory test commands are used to test memory with a scope. The commands read or write a byte, word or long word to the same location continuously until the reset button is pressed.

All of the following scope loop commands provide a way to sync on the read or write by setting the CMB Test Point 8 (TP8) signal high before the read or write and resetting the signal after the write or read is complete.

The scope loop commands perform tight loops with few instructions to perform the loop. These instructions are used to perform the following 4 steps:

1. Set Test Point 1 signal high.
2. Perform the write or read to the selected memory location.
3. Set Test Point 1 signal low.
4. Jump to step one.

WB_LOOP (write byte scope loop)

This command repetitively writes the least significant byte of the selected pattern to the first byte of the range selected by the RANGE or MODULES command.

WW_LOOP (write word scope loop)

This command repetitively writes the least significant word of the selected pattern to the first word of the selected range that falls on a word boundary.

WL_LOOP (write long word scope loop)

This command repetitively writes the entire 4 byte (long word) pattern to the first four bytes of the selected range that falls on a word boundary.

RB_LOOP (read byte scope loop)

This command repetitively reads the first byte of the selected range. The CMB TP8 signal goes high before the byte is read and goes low after the byte is read, providing a scope sync point. The data is not verified.

RW_LOOP (read word scope loop)

This command repetitively reads the first word of the selected range. The CMB TP8 signal goes high before the byte is read and goes low after the byte is read, providing a scope sync point. The data is not verified.

RL_LOOP (read long word scope loop)

This command repetitively reads the first long word of the selected range. The CMB TP8 signal goes high before the byte is read and goes low after the byte is read, providing a scope sync point. The data is not verified.

6.3.9.7 ERROR MESSAGES

Error messages displayed by the Memory Test are self-explanatory. Refer to the test description of any test that fails.

6.3.10 Serial Ports Test

This logic test checks the ability of serial ports to support various printers. The test provides print patterns allowing visual inspection for errors.

6.3.10.1 Loading and Running the Serial Ports Logic Test

The Serial Ports Logic Test is loaded and run in the usual manner for logic tests. At either the diagnostic executive prompt, or at the prompt for another logic test, enter:

```
load ports
```

To run the logic tests in normal mode, enter:

```
run
```

Refer to the command descriptions in Section 6.3.10.3 for additional test run options.

6.3.10.2 Serial Ports Test Description

The Serial Ports test proceeds based on the testing parameters specified by the commands. If no commands are used, the test uses the following default values:

- 80 character column
- Ripple pattern
- CMB port SCC 1
- Basic Four communications protocol

When **RUN** is executed, the test prints the specified pattern according to the test parameters. The operator must visually inspect the pattern to verify that it is correct.

6.3.10.3 Serial Ports Commands

The following commands can be entered directly at the Serial Ports test prompt to set test parameters. These commands are in addition to the general diagnostic executive commands.

BAUD *p* {*x*}

This command sets the baud rate for the specified serial port. The *p* parameter specifies the port to set, and may take the values SC*n* and FW*bn*, where *b* is the controller board number and *n* is the port number. The optional parameter *x* specifies the baud rate, in the range 50 to 19200. If the baud rate is not specified, it defaults to 9600.

GETBAUD *p*

This command displays the current baud rate of the specified port. The *p* parameter specifies the port, and may take the values SC*n* and FW*bn*, where *b* is the controller board number and *n* is the port number.

PATTERN {*pat*}

This command specifies the pattern that the test prints. If no parameter is specified, the ripple pattern is selected. Other options are **E** and **H**, which print lines of "E" and "H", respectively.

PORTS {*list*}

This command specifies the ports to be used in the test. Each port specified in *list* is separated by a space, and takes the value SC*n* or FW*bn*, where *b* is the controller board number (0-3) and *n* is the port number (0-3).

WIDTH {*x*}

This command specifies the width of the character pattern, where *x* is the number of characters. If the parameter is not specified, the width default to 80 characters.

6.3.10.4 ERROR MESSAGES

The ability of a port to support the attached printer is determined by visually inspecting the pattern. The following messages may be displayed, indicating a failure of the test itself or an input error.

- Inalid Input

  This message indicates that the parameter value specified is invalid.

- SCC Error

  This message indicates that an error occurred while trying to communicate with a CMB SCC port. Use the SCC Logic Test to further diagnose the problem.

- 4-Way Error

  This message indicates that an error occurred while trying to communicate with a 4-way port. Use the Four-Way Logic Test to further diagnose the problem.

- Cannot Print on Console Port

  The console port has been selected for testing, which is not allowed. Select an alternate port.

6.3.11 MAI 2000 Floppy Logic Test

This logic test checks the MAI 2000 5 1/4 inch Floppy Disk Controller and Drive(s).

6.3.11.1 Loading and Running the Floppy Logic Test

The Floppy Logic Test is loaded and run in the usual manner for logic tests. At either the diagnostic executive prompt, or at the prompt for another logic test, enter:

```
load floppy
```

To run the logic tests in normal mode, enter:

```
run
```

Refer to the command descriptions in Section 6.3.11.3 for additional test run options.

6.3.11.2 Floppy Logic Test Descriptions

The following paragraphs describe the individual Floppy subtests.

6.3.11.2.1 Test 1 - FCD Addressing

This test verifies that addressing each Floppy Disk Controller (FDC) register does not cause a bus error trap. It checks that a DTACK is generated in response to a valid FDC register address.

6.3.11.2.2 Test 2 - FCD Register Write/Read

This test verifies the integrity of the FDC chip. It writes and then reads various data patterns to each write/read register of the FDC chip, and verifies that the data pattern is stored correctly.

6.3.11.2.3 Test 3 - FCD RAM Buffer Write/Read

This test writes, reads and verifies various data patterns to each address of the FDC RAM buffer on the MAI 2000 CMB.

6.3.11.2.4 Test 4 - FCD RAM Buffer Addressing

This test writes an incrementing data pattern to the entire FDC RAM buffer, and then reads it back. The test verifies unique addressing of the FDC RAM buffer.

6.3.11.2.5 Test 5 - FCD Restore Command

This test issues a restoer command and verifies the correct FDC initial and terminating status. During a second pas, interrupts are enabled and the test verifies the autovector interrupt occurs correctly.

6.3.11.2.6 Test 6 - FCD Write Sector

This test writes sector 01 or track 00 with various data patterns. During the second pass of the test, FDC interrupts are enabled and verified for the write sector command.

6.3.11.2.7 Test 7 - FCD Write/Read Sector

This test writes sector 01 or track 00, then reads back and verifies the data, for various data patterns. During the second pass of the test, FDC interrupts are enabled and verified for the read sector command.

6.3.11.2.8 Test 8 - FCD Sector Addressing

This test writes to each sector (01 to 08) of track 00 with a pattern equal to is sector number, then reads back and verifies the data. The test verifies unique sector addressing.

6.3.11.2.9 Test 9 - FCD Side Addressing

This test writes to each side, 0 and 1, of sector 01, track 00 with a data pattern equal to the side. Both sides are then read back and verified. This verifies unique side addressing.

6.3.11.2.10 Test 10 - FCD Track Addressing

This test writes sector 01 of each track, 00 to 79, with a data pattern equal to its track number. Each track (sector 01) is then read back and verified. This verifies unique track addressing and seek head positioning.

6.3.11.2.11 Test 11 - FCD Worst Case Inner Track

This test writes, then reads and verifies, three worst case data patterns to the last 4 tracks on the diskette.

6.3.11.2.12 Test 12 - FCD Disk Addressing

This is a manual intervention test. The tests writes to all sectors, both sides, of all tracks a data pattern indicating the sector, side and track. It then reads back and verifies the data. This test verifies unique addressing for the entire diskette.

6.3.11.2.13 Test 13 - FCD Write Protect

This is a manual intervention test. The test prompts the operator to insert a write protected diskette and press **ENTER**. It then attempts to write to the diskette, and verifies that write protect status is detected by the drive. The operator is then prompted to remove the write protected diskette.

6.3.11.3 FLOPPY TEST COMMANDS

The following commands can be entered directly at the flopy test prompt to perform operations or set test parameters. These commands are in addition to the general diagnostic executive commands.

DCOMMAND

This command displays the command list last output to the FDC controller. For example:

CNTRL	BUFCTL	CMD	TRACK	SECTOR	DATA
XX	XX	XX	XX	XX	XX

Each XX is a hexadecimal value.

STATUS

This command reads and displays the contents of the FDC controller. For example

STATUS	TRACK	SECTOR	DATA	BDSTATUS
XX	XX	XX	XX	XX

Each XX is a hexadecimal value.

CLRLOG

This command clears the accumulated run statistics.

LOGOUT

This command displays the accumulated run statistics.

RUNIN

This command causes execution of a random track, sector, side seek, write/read loop in order to exercise the selected froppy drive(s). Run statistics and errors are accumulated, and displayed approximately every 30 seconds.

DRIVES *list*

This command selects the drives specified in *list* for testing. The legal drive numbers are 0 and 1. If both drives are specified, their numbers must be separated by a comma.

QUICK

This command executes a short test sequence, consisting of tests 1, 2, 3, 4, 5 and 11.

ALIGN

This command executes the floppy disk alignment routine. The operator is prompted to select either the drive alignment test, which requires the Dysan Alignment Diskette (Model 506-44, Part Number 802010), or analog alignment routines.

6.3.11.4 ERROR MESSAGES

When an error occurs, the Floppy Logic Test displays an error message followed by additional information indicating the failing funciton, program listing address and error number. On the following lines, further information relevant to the failure is displayed. For example:

Timed out waiting for FDC busy after cmd: Restore 003D4A 21

status	track	sector
XX	XX	XX

In this example, the failing function is Restore, 003D4A is the program address and 21 is the error number. The status, track and sector values are given in hexadecimal.

The specific error messages that may be displayed are:

- Accessing an FDC address has caused a bus error trap
- FDC chip write/read compare error
- FDC Ram Buffer data compare error
- FDC RAM Unique Address error
- FDC busy prior to issuing command
- Timed out waiting for FDC busy after issuing cmd
- FDC status not correct following cmd
- With FDC interrupts enabled, no FDC interrupt occurred
- With FDC interrupts disabled, an FDC interrupt occurred
- Timed out waiting for INTP after issuing command
- Sector write/read compare error
- Sector Addressing Test error
- Wrt Protect Status bit not set in FDC status
- Disk Addressing Test error
- Track Addressing Test error
- Drive is not rdy, busy or wrt-protected!

6.3.12 MAI 2000 Bus Logic Test

This logic test checks the functionality of the MAI 2000 I/O Bus.

The tests use the special EIEIO board, which must be installed prior to running the tests. Several tests use the EIEIO board in slave mode. Slave mode allows the CMB to have full control over the EIEIO board, for reading and writing EIEIO registers and RAM. The CMB can read and write both the DMA throttle counter and the interrupt vector latch on the EIEIO board.

The board also works in two DMA modes. In single-word DMA mode, the EIEIO is programmed to DMA a single word (2 bytes) per bus grant. In burst DMA mode, the EIEIO may be programmed to DMA up to 16 words (as programmed), waits for some specified period, and begins the cycle again. This is repeated until the entire data block has been DMA'd.

The tests begin by checking the EIEIO board itself and then, provided it is operational, proceeds to testing the CMB's I/O bus.

6.3.12.1 Loading and Running the I/O Bus Logic Test

The I/O Bus Logic Test is loaded and run in the usual manner for logic tests. At either the diagnostic executive prompt, or at the prompt for another logic test, enter:

```
load iotst
```

To run the logic tests in normal mode, enter:

```
run
```

Refer to the command descriptions in Section 6.3.12.3 for additional test run options.

6.3.12.2 I/O Bus Test Descriptions

The following paragraphs describe the individual I/O Bus subtests.

6.3.12.2.1 Test 1 - EIEIO Addressing

This test writes to each port of the EIEIO board, verifying that the addresses do not cause a bus error trap.

6.3.12.2.2 Test 2 - Write/Read Throttle Register

The DMA throttle register is responsible for creating a time delay after a DMA burst and before the next bus request. This test writes several byte patterns to this register. The data is then read back and compared.

The test checks the CMB bus data lines D08 to D15, the address bus, the Upper Data Strobe (UDS) and the ability to enable data on the CMB bus.

6.3.12.2.3 Test 3 - Write/Read Interrupt Vector Register

The interrupt vector latch holds the interrupt vector number for bus vectored interrupts. This test writes various byte data patterns to this register. The data is then read back and compared.

The test checks the CMB bus data lines D00 to D07, the address bus, Lower Data Strobe (LDS) and the ability to enable data on the CMB bus.

6.3.12.2.4 Test 4 - Slave RAM Write/Read

This test is primarily a check of the functionality of the EIEIO board. The test writes various word data patterns to the EIEIO board RAM, reads it back and compares the data.

6.3.12.2.5 Test 5 - Slave RAM Unique Addressing

This test is primarily a check of the functionality of the EIEIO board. The test writes address 0 of the slave RAM with 0, then writes a RAM address one bit different with all ones. RAM address 0 is then read back, and it is verified that it was not overwritten.

If the address was overwritten, slave RAM addressing is not unique. A RAM address line is probably stuck.

6.3.12.2.6 Test 6 - Slave RAM Addressing

This test checks the EIEIO board's ability to uniquely address each RAM address by means of slave I/O.

Each address of the slave RAM is written with data equal to the address. It is then read back and verified.

6.3.12.2.7 Test 7 - Autovectored Interrupt Test

Autovectored interrupts are set on levels 2 and 4. Interrupts are then requested at these levels, and tested for response.

6.3.12.2.8 Test 8 - Bus Vectored Interrupt

This test writes each valid interrupt vector number to the interrupt vector latch. An interrupt is then requested by the EIEIO board. The test checks that the interrupt was serviced, then increments to the next vector number.

6.3.12.2.9 Test 9 - Single Word DMA Write to CMB

This test writes a single word, 5555 hex, to address 0 of RAM, in EIEIO slave mode. A command is then sent to the EIEIO for a DMA write to CMB memory. The data is compared to verify functioning of DMA.

6.3.12.2.10 Test 10 - Single Word DMA Read from CMB

A CMB memory location is loaded with a single word, AAAA hex. The EIEIO reads this location into EIEIO RAM from the CMB by a DMA. The word is then read back from the EEIO in slave mode and compared to the original value.

6.3.12.2.11 Test 11 - DMA Write to CMB in Burst Mode

This test fills EIEIO RAM with the address data pattern. It then writes this to the CMB by DMA in burst of 16 words per bus mastership. The data is then compared, checking for errors.

6.3.12.2.12 Test 12 - DMA Read from CMB in Burst Mode

This test fills a 256 word CMB buffer with a walking ones data pattern. The EIEIO board then reads this data by DMA into EIEIO RAM . The data is then read back from EIEIO RAM in slave mode and compared with the original data.

6.3.12.3 EIEIO FUNCTION COMMANDS

The following commands can be entered directly at the I/O test prompt to perform operations or set test parameters. These commands are in addition to the general diagnostic executive commands.

READRAM *x*

This command reads the EIEIO RAM location specified by *x* (0 to FF hex), and displays it on the terminal screen.

WRITERAM *x,y*

This command writes to the EIEIO RAM location specified by *x* (0 to FF hex). The data is specified by *y*, and is one word long (2 bytes).

6.3.12.4 ERROR MESSAGES

The following errors may be reported by the I/O test:

- Bus timeout trap while addressing EIEIO

  A bus error occurred while trying to access an EIEIO port.

- Data Compare Error, DMA Throttle Register

  Data written to the DMA throttle register was not the same as the data read.

- Data Compare Error, Interrupt Vector Register

  Data written to the interrupt vector register was not the same as the data read.

- Data Compare Error, RAM slave I/O

  Data read from the EIEIO RAM was not the same as the data written in slave mode.

- RAM Non-unique Addressing Error

  Data was written to two different EEIO RAM addresses, but the data read back from the first address was the data written to the second address.

- RAM Addressing Test Error

  The address data pattern was written to each RAM address (data = RAM address) but the data read back was not the same as the data written.

- Autovector Interrupt Not Serviced

  The EIEIO requested an autovector interrupt, but the CMB did not respond.

- Bus Vectored Interrupt Not Serviced

  The EIEIO requested a bus vectored interrupt, but the CMB did not respond.

- Ready Bit not set, Time Out

  The EIEIO was programmed to DMA, but never set ready after the operation. This indicates an EIEIO hardware problem.

- DMA Data Compare Error

  The data found in the DMA buffer was not the same as the data that was supposedly DMA'ed.

6.3.12.5 ERROR MESSAGE HEADERS

The following may be included with error messages, giving more specific information about the error.

- Address - The EIEIO port that was being accessed when the error occurred.
- PC @ Err - The program counter value at the time of the error.
- SR - The contents of the Status Register at the time of the error.
- IR - The contents of the Instruction Register at the time of the error.
- Acc Addr - Address being accessed by an aborted bus cycle.
- Wr Data, Rd Data - The data written and the data read are displayed for comparison.
- Overwrite Address - The second EIEIO RAM address where data was written during an overwrite error.
- Level - I/O error level.
- Vector - Interrupt vector placed on the bus during a bus vectored interrupt.
- Invalid Input - Bad data was entered by the operator on an EIEIO function command.

6.3.13 Real Time Clock Logic Test

The Real Time Clock Logic Test checks the functionality of the Motorola MC146818 real time clock chip (RTC) used on the MAI 2500/3000/4000 systems (the MAI 2000 does not use this chip). The test verifies the functioning of the chip's 50 bytes of RAM, RTC status, alarm status and reading and displaying the time.

6.3.13.1 Loading and Running the RTC Logic Test

The RTC Logic Test is loaded and run in the usual manner for logic tests. At either the diagnostic executive prompt, or at the prompt for another logic test, enter:

```
load rtc
```

To run the logic tests in normal mode, enter:

```
run
```

Refer to the command descriptions in Section 6.3.13.4 for additional test run options.

6.3.13.2 RTC Test Initialization

When the test is loaded, it checks the Valid Time/Ram (VTR) bit in the RTC chip. This bit shows if a power failure to the chip has occurred. If a power failure has occurred, the user is warned to set the time before running the diagnostic. Use the **SET_TIME** command (Section 6.3.13.4) to set the time.

6.3.13.3 RTC Test Descriptions

6.3.13.3.1 Test 1 - RTC RAM Write/Read

This test writes and reads data to the fifty bytes of RAM on the RTC chip, to test the ability of each to store and pass data back to the CPU.

The test writes 55 hex to the first RAM byte used, then reads and compared the data. The remaining bytes are then tested in the same way. A second pass is then run using AA hex. If any miscompares occur, the error is reported with the address, expected value and received value.

6.3.13.3.2 Test 2 - RTC RAM Address

This test verifies the ability to uniquely address each byte of RTC RAM and each register.

The test writes each of the fifty bytes of RAM with the least significant 8 bits of its own address. The data is then read and compared. All registers are also written, read and compared. If any miscompares occur, the error is reported with the address, expected value and received value.

6.3.13.3.3 Test 3 - RTC Status

This test checks the normal running status of the RTC. First, the VTR bit in register D is checked, and should be ON. If it is not ON, an RTC Power Failure is reported.

Next, the UIP bit in register A is checked. If it is ON, it is turned OFF and then back ON; if it is OFF, it is turned ON and then back OFF. If it does not toggle, an RTC Update Failure is reported.

6.3.13.3.4 Test 4 - RTC Display Time

This test displays the current RTC time, day and date. If the UIP bit is not updated, the time, day and date are not displayed, and an RTC Update Failure is reported.

6.3.13.3.5 Test 5 - RTC Alarm Status

This test determines whether the RTC alarm functions properly.

The test saves the current alarm values, and some arbitrary values are inserted. The alarm is checked after the next RTC update. If the UIP bit is not updated, and RTC Update Failure is reported. If the alarm fails, an RTC Alarm Failure is reported.

6.3.13.4 RTC TEST COMMANDS

The following commands can be entered directly at the RTC test prompt to perform operations or set test parameters. These commands are in addition to the general diagnostic executive commands.

SET_TIME

This command starts and sets the RTC chip. The user is prompted for the necessary information.

The first prompt asks for the enable byte. Press **ENTER** alone to set to the default value, 81 hex, which selects decimal, 12-hour mode with daylight savings.

The next prompt asks for the control byte. Press **ENTER** alone for the 20 hex default value, which is the standard value for the clock divider entry.

Prompts for the week, month, time, etc. are then displayed. Enter a value 01 to 12 for AM time, or 81 to 92 for PM time.

6.3.13.5 ERROR MESSAGES

The error messages displayed by the RTC Test are self-explanatory. Refer to the individual test descriptions.

6.3.14 Memory Management Unit Logic Test

The Memory Management Unit (MMU) Logic Test checks the functionality of the Motorola 68851 Paged Memory Management Unit (PMMU) coprocessor or the Motorola M68KVMMB851 memory management board used on the MAI 2500/3000/4000 systems (the MAI 2000 does not use these components).

6.3.14.1 Loading and Running the MMU Logic Test

The MMU Logic Test is loaded and run in the usual manner for logic tests. At either the diagnostic executive prompt, or at the prompt for another logic test, enter:

```
load mmu
```

To run the logic tests in normal mode, enter:

```
run
```

Refer to the command descriptions in Section 6.3.14.4 for additional test run options.

6.3.14.2 MMU Test Initialization

When the test is loaded, it checks to see which memory management component is in place. If the PMMU is installed, the test prompt becomes:

```
<pmmu>
```

If the memory management board is in place, the prompt becomes:

```
<mmb>
```

6.3.14.3 RTC Test Descriptions

6.3.14.3.1 Test 1 - MMU Disable NASA

This test performs a raw memory test with the MMU disabled. The test ensures that physical memory is fully addressable prior to testing the MMU mapping of memory.

An error during this test indicates a failure of the main memory. If this is the case, run the memory diagnostic to isolate the failure.

6.3.14.3.2 Test 2 - MMU Write/Read/Compare TC

This test verifies that the Translation Control Register is addressable and that it retains its assigned value. The test writes a value to the register, then reads it and compares this value with the value written.

A comparison error indicates that the MMU translation control register is not retaining data, or the bus interface to the MMU is faulty.

6.3.14.3.3 Test 3 - MMU Enable Direct Addressing

This test enables the MMU with all memory mapped one-to-one.

If the system hangs during this test, translation does not work at all; the coprocessor chip must be replaced. If the test continues to fail after replacing the MMU, the memory addressing logic on the CMB is failing; replace the CMB.

6.3.14.3.4 Test 4 - MMU Modify and Used Status

This test writes and reads specific memory locations with MMU translation enabled. The appropriate used and modified bits of the translation tables are checked to verify that they have been set properly.

Failure of the used and modified bits prevents page replacement algorithms from functioning; the coprocessor chip should be replaced. If the test continues to fail after replacing the MMU, the memory addressing logic on the CMB is failing; replace the CMB.

6.3.14.3.5 Test 5 - MMU Write Protect Page Descriptor

This test checks the ability of the MMU to detect a write protection violation at the page descriptor level and inform the processor via a bus error.

If this test reports errors, then the MMU coprocessor should be replaced. If the test continues to fail after replacing the MMU coprocessor, then the bus error logic on the CMB is failing, and the CMB should be replaced.

6.3.14.3.6 Test 6 - MMU Write Protect Pointer Descriptor

This test checks the ability of the MMU to detect a write protection violation at the pointer descriptor level, and to inform the process of via a bus error.

If the test reports errors, then the MMU coprocessor should be replaced. If the test continues to fail after replacing the MMU coprocessor, then the bus error logic on the CMB is failing, and the CMB should be replaced.

6.3.14.3.7 Test 7 - MMU Invalid Pointer Descriptor

This test checks the ability of the MMU to detect an invalid pointer descriptor and to inform the processor via a bus error.

Failure of the MMU to detect and report invalid pointer descriptors may result in invalid translations; replace the MMU coprocessor. If the test continues to fail after replacing the MMU coprocessor, then the bus error logic on the CMB is failing, and the CMB should be replaced.

6.3.14.3.8 Test 8 - MMU Invalid Page Descriptor

This test checks the ability of the MMU to detect an invalid page descriptor and to inform the processor via a bus error.

Failure of the MMU to detect and report invalid page descriptors may result in invalid translations; replace the MMU coprocessor. If the test continues to fail after replacing the MMU coprocessor, then the bus error logic on the CMB is failing, and the CMB should be replaced.

6.3.14.3.9 Test 9 - MMU Upper Limit

This test checks the ability of the MMU to detect an upper limit violation when scanning the page table descriptors and inform the processor via a bus error.

Failure of the MMU to detect and report the error may result in invalid translations; replace the MMU coprocessor. If the test continues to fail after replacing the MMU coprocessor, then the bus error logic on the CMB is failing, and the CMB should be replaced.

6.3.14.3.10 Test 10 - MMU Lower Limit

This test checks the ability of the MMU to detect an lower limit violation when scanning the page table descriptors and inform the processor via a bus error.

Failure of the MMU to detect and report the error may result in invalid translations; replace the MMU coprocessor. If the test continues to fail after replacing the MMU coprocessor, then the bus error logic on the CMB is failing, and the CMB should be replaced.

6.3.14.3.11 Test 11 - MMU Indirect Page Descriptor

This test checks the ability of the MMU translation to follow in indirect page table descriptor while traversing the page tables.

Failure of the MMU to follow the indirect descriptor may result in invalid translations; replace the MMU coprocessor.

6.3.14.3.12 Test 12 - MMU Translation

This test verifies the ability of the MMU to map logical addresses to different physical addresses.

Failure of the MMU to change logical addresses to physical addresses will result in incorrect memory accesses; replace the MMU coprocessor. The the problem persists, the address logic on the CMB may be failing; replace the CMB.

6.3.14.3.13 Test 13 - MMU Supervisor/User Transitions

This test calls a module to run in user mode to verify that the MMU translates properly for both processor states.

If the system hangs during this test, the MMU is not translating properly in user mode; replace the MMU coprocessor.

6.3.14.3.14 Test 14 - MMU User Read Access Level

This test checks the detection and reporting of read access level violations in user mode.

If the MMU does not report access level violations via a bus error, system security is compromised; replace the MMU coprocessor. If the problem persists, the bus error logic on the CMB may be failing; replace the CMB.

6.3.14.3.15 Test 15 - MMU User Write Access Level

This test checks the detection and reporting of write access level violations in user mode.

If the MMU does not report access level violations via a bus error, system security is compromised; replace the MMU coprocessor. If the problem persists, the bus error logic on the CMB may be failing; replace the CMB.

6.3.14.3.16 Test 16 - MMU/FPCP Interference

The test verifies the ability of the MMU and the Floating Point Coprocessor to coexist on the processor bus. This test does not run in the FPCP is not installed.

Failures may be caused by chip select, bus arbitration problems, or a faulty FPCP chip; run the FPCP logic test to verify floating point operation.

6.3.14.3.17 Test 17 - MMU R-M-W Cycle

This test requires operator intervention. It should be run only if the PMMU coprocessor is installed.

6.3.14.3.18 Test 18 - MMU Enabled (No FCTAB) Addressing

This test requires operator intervention, and should be run only if the PMMU coprocessor is installed.

This test checks the ability of the PMMU to translate addresses correctly without an initial function code table.

6.3.14.3.19 Test 19 - Early Termination Test

This test requires operator intervention, and should be run only if the PMMU coprocessor is installed.

Successful completion of this test indicates that the PMMU correctly handles early table termination at the root pointer level.

6.3.14.3.20 Test 20 - MMU Write/Read/Compare Registers

This test requires operator intervention, and should be run only if the PMMU coprocessor is installed.

Successful completion of this test indicates that he PMMU correctly handles having its registers written to and read from, and that those data values compare. If this test fails, the PMMU must be replaced.

6.3.14.4 MMU TEST COMMANDS

The following commands can be entered directly at the MMU test prompt to perform operations or set test parameters. These commands are in addition to the general diagnostic executive commands.

AC	Access Control
CAL	Current Access Level
CRP	CPU Root Pointer
DRP	DMA Root Pointer
PCSR	PMMU Cache Status Register
PSR	PMMU Status Register
SCC	Stack Change Control
SRP	Supervisor Root Pointer
TC	Translation Control
VAL	Validate Access Level

The TC command asks for a translation file, and then proceeds to read the tape. The remaining commands display the current value of the appropriate register of the PMMU. Only the CRP and TC commands are available for the MMB.

BAC *n*
BAD *n*

These commands display or sets the Breakpoint Acknowledge Control register and the Breakpoint Acknowledge Data register, respectively. If an argument is specified (00-07 hex), the register is set to that value.

6.3.14.5 ERROR MESSAGES

The following error messages may be displayed the MMU Logic Test, providing descriptive information about the error that occurred.

- MMU Register Date Miscompare Error - Error #1

  'Data Miscompare'
  'Reg XX Num XX Expect XXXXXXXX Actual XXXXXXXX'

  This message indicates an error writing to and reading from the MMU registers. The message reports the register number (1-7), the value written (expected) and the value read (actual).

- MMU Memory Data Miscompare Error - Error #2

  'Data Miscompare'
  'Address XXXXXXXX Expect XXXXXXXX Actual XXXXXXXX'

  This message indicates an error writing to and reading from memory using the PMMU/MMB. The message reports the address value, the value written (expected) and the value read (actual).

- MMU Unexpected Address Error - Error #3

  'Unexpected Address Error'
  'Address XXXXXXXX'

  This message indicates that an error occurred at an unexpected address. The message reports the address.

- MMU Forced Address Error - Error #4

  'Expected Address Error Not Received'
  'Address XXXXXXXX'

  This message indicates that the test forced an address error, but the PMMU did not detect or report it. The message reports the address.

- MMU Unexpected Bus Error - Error #5

  'Unexpected Bus Error'
  'Address XXXXXXXX'

  This message indicates that an unexpected bus error occurred with the MMU. The message reports the address.

- MMU Expected Bus Error not Received - Error #6

  'Unexpected Bus Error Enabling MMU'
  'Address XXXXXXXX'

  The test forced a bus error, but the PMMU/MMB did not properly detect or report it. The message reports the address.

- Wrong Status Received from Status Byte of Page Descriptor - Error #7

  'Wrong Status Byte in Page Descriptor'
  'Logical XXXXXXXX Descriptor XXXXXXXX'
  'Expect XXXXXXXX Actual XXXXXXXX'

  This message indicates that the value read from the Status Byte of the Page Descriptor is not the value that should have been read. The message reports the logical address, the descriptor, the expected value and the actual value read.

- Translation Register Miscompare Error - Error #8

  'TC register miscompare'
  ' Expect XXXXXXXX Actual XXXXXXXX'

  This message indicates that the value read from the Translation Register is not the value that should have been read. The message reports the expected value and the actual value read.

- Root Pointer Miscompare Error - Error #9

  'RP register miscompare'
  ' Expect XXXXXXXX Actual XXXXXXXX'

  This message indicates that the value read from the Root Pointer is not the value that should have been read. The message reports the expected value and the actual value read.

- MMU Processor Argument Error - Error #10

  'PROC argument error'
  'Value XXXX'

  This message indicates that the argument value for a MMU process service was too long. The invalid argument is displayed.

- Bus Error received in error when MMU was enabled - Error #11

  'Unexpected Bus Error Enabling MMU'

  This message indicates that a bus error occurred when enabling the PMMU/MMB.

- MMU Early Termination Miscompare Error - Error #33

  'Data Miscompare'
  'Address/Expect XXXXXXXX Actual XXXXXXXX'

  This error indicates an early termination occurred. The message reports the expected and actual addresses.

6.3.15 MAI 3000/4000 CMB Logic Test

The CMB Logic test is providing to test and verify the correct operation of several areas of the CMB which are not tested by the other logic tests. It is intended to be used only after all other logic tests have been used without isolating the problem. All of the tests are independent. Most of the tests are manual intervention tests.

6.3.15.1 Loading and Running the CMB Logic Test

The CMB Logic Test is loaded and run in the usual manner for logic tests. At either the diagnostic executive prompt, or at the prompt for another logic test, enter:

```
load cmb
```

Since most of the tests are manual intervention tests, enable these tests:

```
options manual
```

To run the logic tests in normal mode, enter:

```
run
```

Refer to the command descriptions in Section 6.3.15.3 for additional test run options.

6.3.15.2 CMB Test Descriptions

6.3.15.2.1 Test 1 - LED Display Test

This test requires visual inspection of the LED display to verify correct functioning. The LED display on the CMB is first cleared, and then incremented by 1 until FFH is displayed. The display is then cleared again, then incremented by 10H (and 1) until FFH is displayed. The test can be terminated by **CTRL + C**.

6.3.15.2.2 Test 2 - Data Cache Inhibit Bit

This test verifies the proper functioning of the data cache parity bit. First, the test sets the parity bit, then writes data into memory with data parity error checking ON. Parity checking is then turned OFF, and the data is read back. Because the data was originally written with parity error encoding, the data cache inhibit bit should be flagged when it reads the data.

6.3.15.2.3 Test 3 - Switch Setting Verification

This test displays the contents (setting) of the sensor switches, located near the power supply. You can change the switch setting to verify that the switch register is working. To terminate this test, press **CTRL + C**.

NOTE

Exercise caution when changing switch settings. The switches should be returned to their original settings following this test. The test reads the original value, and displays it for reference when the test is complete. DO NOT change the settings of switches 9 and 10.

6.3.15.2.4 Test 4 - SCC Serial Enable Bit

This test checks both CMB serial ports, so the system console must be configured on a 4-way or 8-way. The communications blocks for the CMB serial ports must be configured as DTE or DCE. If port 1 is set for DCE, as for down-loading from a host system, the block must be reversed. Port 1 may be configure as for use as the system console, but may not be the console for this test.

A VDT must be connected to the port being tests. It is preferable to have a VDT connected to both ports, allowing a quick verification that channel A is port 0 and channel B is port 1, and that when one of the ports is addressed, only that port is actually tested. The port to test can be selected from the system console on the 4-way or 8-way port.

The test configures the port under test for local loopback, then writes data to the SCC data register and loops through the data verifying that the data transmitted is the same as the data received. The data is in two parts, as shown below. For the first part, the output drivers (external to the SCC chip) are enabled, and for the second part the drivers are disabled. In this way, all of the data can be verified in the SCC chip, but only the first part should be displayed on the VDT. The data is:

Part 1: "How are we doing?"
Part 2: "You should not be able to see this!"

6.3.15.2.5 Test 5 - Parallel Enable Bit

This test is similar to test 4. No special configuration is required, except that a parallel output device, usually a printer, must be connected to the parallel port. The two part message shown in test 4 is transmitted and verified. Only the first part should actually show on the output device.

6.3.15.2.6 Test 6 - NMI Switch Detect Bit

This test replaces the level seven interrupt vector with an alternative address. The user can then verify level seven interrupts by pressing the NMI contact. A confirmation message is displayed if the NMI is detected. If a timeout occurs before the NMI is detected, an error message is displayed. Upon completion of the test, the original level seven interrupt address is restored.

6.3.15.2.7 Test 7 - Power Fail Detect Bit

This test verifies that, in a power fail situation, the system can execute an interrupt subroutine and set the power-fail bit in NVRAM before the remaining power has expired. When the test starts, it asks the user whether the tests has been started before. The test requires starting the test twice. The first time the user should reply NO to the prompt, and the second time reply YES.

To perform this test:

1. Start the test.
2. Answer NO to the prompt (the test will then prepare to be powered off).
3. Turn off the power.
4. Reload the diagnostics are restart this test.
5. Answer YES to the prompt.

The test completes by verifying whether the test was able to complete its routine before the final power expired.

6.3.15.2.8 Test 8 - UPS Transition Bit

The UPS transition detects a direct signal from the UPS, a level seven interrupt. When you enter this test, you have approximately 30 seconds to turn OFF the UPS and then turn it back ON. If all is functioning properly, a verification message is displayed.

If a UPS is not available, you can simulate a power fail by shorting the two pins on the back side of the power supply labeled UPS. These are the same to pins that connect to the PUS from the system power supply.

6.3.15.2.9 Test 9 - EBUS Interrupt Bit

The EBUS interrupt is tested by pressing the NMI switch on the MAI 4000 expansion unit.

If no expansion unit is attached, the test verifies that no EBUS interrupt occurred or that the NMI register has the correct information regarding the lack of expansion unit. If the CMB thinks there is an expansion unit when there is none, this is noted by an error message.

6.3.15.3 CMB TEST COMMANDS

ERROR_DEF {*x*}

This command displays error descriptions. If the parameter *x* is specified, the description for that error only is displayed. Otherwise, descriptions for all errors are displayed.

GLOSSARY

This command displays brief descriptions of the CMB test subtests.

6.3.15.4 ERROR MESSAGES

The following error messages may be displayed by this test.

- SCC TIMED OUT WAITING FOR RX AVAILABLE
- SCC TX CHAR NOT EQUAL TO RX CHAR
- DATA BYTE<15-8> COMPARE ERROR
- BYTE<15-8> WRITE/READ HAS CAUSED A BUS ERROR TRAP!
- CACHE INHIBIT BIT IS STUCK(?) T A C
- CACHE INHIBIT BIT SHOULD BE CLEARED; BUT NOT
- INTERRUPT NOT CAUGHT?

6.3.16 CMB Cache Logic Test

This test verifies the functionality of the CMB Cache on the MAI 2500/3000/4000 systems, and helps to isolate faults in the functionality of the Cache to the Cache hardware. The CMB Cache checked by this test is different from the cache resident within the MC68020.

The Cache consists of 8K bytes of very high speed memory which are divided into two set of 4K bytes (1K long word) each. Each addressable location of this memory also has its own associated Tag Word for determining whether a given value is correct.

6.3.16.1 Special Hardware Requirements

To completely test the Cache, a Winchester Disk and controller must be installed. This is used for DMA tests.

6.3.16.2 Loading and Running the Cache Logic Test

The Cache Logic Test is loaded and run in the usual manner for logic tests. At either the diagnostic executive prompt, or at the prompt for another logic test, enter:

```
load cache
```

To run the logic tests in normal mode, enter:

```
run
```

Refer to the command descriptions in Section 6.3.16.5 for additional test run options.

6.3.16.3 Cache Test Initialization

When the test is loaded, it performs two sizing functions, determining the size of memory and the presence of DMA devices (Winchester Disk Controllers). If no disk controller is found, this message is displayed:

```
No DMA Device(s) Present. Tests requiring DMA device will not be executed.
```

If you wish to test the DMA functions, power-off the system and install a Winchester subsystem, and then reload the diagnostic. If the message persists, or if a subsystem is already installed, use the Winchester Disk Tests to find the problem.

6.3.16.4 CACHE TEST DESCRIPTIONS

6.3.16.4.1 Test 1 - Cache Data Test

This test checks the ability of the Cache to cache a long word data value in every long word existing in memory. This process is carried out in 1024 long word blocks in set zero of the Cache only. The Diagnostic Executive and Cache Logic Test are moved are required to test all memory locations.

6.3.16.4.2 Test 2 - Cache Instruction Test

This test checks the ability of the Cache to cache instruction operands in every memory location. The test creates routines which, when executed, give different results depending on whether they are run from memory or from the Cache. The routine is then moved to the next memory location and run again. This is repeated until all memory locations have been tested.

6.3.16.4.3 Test 3 - Cache Control Logic Test

This test checks the logic controlling what is cached and when. This includes the logic to switch between two sets and the logic to perform a least recently used replacement scheme between these two sets.

6.3.16.4.4 Test 4 - Cache Parity Interrupt Test

This test checks the ability of the Cache to report and operate when a parity error is detected within the Cache memory. The test proceeds in four parts, testing Cache Tag Word Parity Errors, Cache Data Word Parity Errors, Cache operation when receiving parity errors with interrupts disabled, and Cache operation on retry.

6.3.16.4.5 Test 5 - Cache DMA Test

This test checks the operation of the Cache when memory is being read from and written to by a DMA device (Winchester Disk Controller). The test programs the DMA device to write to and read from the data buffers. It then checks the Cache's ability to perform a DMA write hit, DMA miss replace and DMA match read. These checks verify that data is cached at the right time when dealing with a DMA device.

6.3.16.5 CACHE TEST COMMANDS

The following commands can be entered directly at the Cache test prompt to perform operations or set test parameters. These commands are in addition to the general diagnostic executive commands.

QUICK

This command performs a quick run of the Cache test. Tests 1-5 are executed.

6.3.16.6 ERROR MESSAGES

The following error messages may be displayed by the diagnostic. Errors that occur during the set-up of the test are fatal, and require that the test be exited. Errors that occur during the test run are not fatal, and are reported to the console together with diagnostic information. The non-fatal errors are listed according to the test that may report them.

6.3.16.6.1 Fatal Errors

- No DMA Device(s) Present

  This error simply skips test 5, the DMA test.

- Data Not Being Cached by Cache Properly - Run Cache Data Test to Verify Correct Data Caching.

  This error message is displayed by tests other than the Cache Data Test when the cache is not caching properly and the test requires proper caching. If this error occurs, run the Cache Data Test to verify proper caching.

6.3.16.6.2 Non-fatal Errors

Test 1 - Data Cache Test

- Cache Data Error

  This error message indicates that the value received from a data buffer read is not the value written. If the received value is 1B1B1B1b (the second data pattern), then the data is from memory on a write-hit. The message reports the expected data, the received data, and the address used.

Test 2 - Cache Instruction Test

- Cache Instruction Error

  This error message indicates that the Cache did not cache instructions properly. If the received value is all zeros, then the subroutine was executed from memory and not from Cache. The message reports the expected and returned values for the eight buffers.

Test 3 - Cache Control Logic Test

- Cache Dual Set Error

  This error message indicates that either one or both of the received data values are not the expected values. If the received value is 1B1B1B1B (the second data pattern for Set 0) or 2B2B2B2B (the second data pattern for Set 1), then the data is from memory on a write-hit. The message reports the data received and expected values for both sets 0 and 1, and the addresses.

- Cache Least Recently Used Logic Error

  This error message indicates that an error occurred in replacing the least recently used data in the cache. The message reports the received and expected values for either one or both sets 0 and 1, and the address(es).

- Cache Read/Write Data Error

  This error message indicates that an error occurred in the write/read security, that writes and reads were not properly restricted. The error occurred either when writes were enabled and reads disabled, or when writes were disabled and reads enabled. The message reports the received and expected data, and the address.

Test 4 - Cache Parity Interrupt Test

- No Interrupt Serviced on a Forced Tag Word Parity Error

  This error indicates that a forced parity error on the Tag Word was not detected or reported. The message indicates the data value and address.

- Tag Parity Bits (PEX) Not Cleared on Parity Error

  This error indicates that the proper bits were not set in the Cache Parity Status register signaling the Tag Word in error. The message indicates the Tag Word parity bit in error.

- No Interrupt Serviced on a Forced Data Word Parity Error

  This error indicates that a forced parity error on the Data Word was not detected or reported. The message indicates the data value and address.

- Data Word Parity Bits (DCPEX) Not Set on Parity Error

  This error indicates that the proper bits were not set in the Cache Parity Status register signaling the Data Word in error. The message indicates the Data Word parity bit in error.

- Cache Interrupted with Interrupts Inhibited

  This error indicates that, with interrupts disabled, the CPU still received a level 3 interrupt from the Cache in a parity error (either Tag Word or Data Word). The message reports the data value and address.

- No Retry on Cache Parity Tests

  This message indicates that, when a parity error was forced, the processor did not perform a retry to retrieve the data value from memory. The message reports the data value and address.

Test 5 - Cache DMA Test

- DMA Controller Error

  This error can occur during any of the DMA tests. It indicates that there is an error with the DMA device. Use the disk diagnostics to check the Winchester Disk subsystem.

- Cache DMA Write-Hit Error

  This error indicates that the Cache did not cache the data correctly when the DMA device was loading data into a data buffer. If the data pattern is 2A2A2A2A, then the Cache did not update on a write-hit. If the data pattern is 3A3A3A3A, then the Cache was not properly initialized prior to the DMA loading of the data buffer. The message reports the received data, the expected data, and the address.

- Cache DMA Write-Miss Error

  This message indicates that the Cache cached data during a DMA write cycle, when the device loaded data into the data buffer which had not previously been initialized into the Cache. The message reports the data received, the data expected, and the address.

- Cache DMA Read-Hit Error

  This error indicates that the Cache cached data during a DMA read cycle when the DMA device was being loaded with data. The message reports the received data, the expected data, and the address.

6.3.17 MAI 4000 Expansion Interface Logic Test

The MAI 4000 Expansion Interface Logic Test verifies the correct functioning of the expansion chassis interface. The interface consists of two boards, the EIA board residing in the base chassis, and the EIB board residing in the expansion chassis.

The Expansion Interface Logic Test also tests the two parallel ports on the EIA board. For some of these tests a parallel port loopback connector is required (MBF part number 907588-001).

6.3.17.1 Loading and Running the Expansion Interface Logic Test

The Expansion Interface Logic Test is loaded and run in the usual manner for logic test. At either the diagnostic executive prompt, or at the prompt for another logic test, enter:

```
load eiab
```

To run the logic tests in normal mode, enter:

```
run
```

Refer to the command descriptions in Section 6.3.17.4 for additional tests run options.

6.3.17.2 Test Initialization

When the test is loaded, and initialization routine runs that tests for the presence and proper communication of the expansion interface boards and then level of the CMB. If the proper boards are not installed, a message is displayed and the diagnostic aborts returning to the diagnostic executive. If only the EIB is not found or is not communicating properly, the test continues, though a message indicates the problem.

If the interface appears correct, the test prompt is displayed:

```
<eiab>
```

The test defaults to testing the first parallel port, PI/T 1. The PIT_SEL command, described in Section 6.3.15.4, can be used to change this.

6.3.17.3 EXPANSION INTERFACE TEST DESCRIPTIONS

6.3.17.3.1 Test 1 - EIA Read/Write Registers

The test checks the basic communication of the EIA board by reading and/or writing to all of the registers on the EIA board. These registers include the EIA status/control register, the EIA Interrupt 7 status register, and the EIA Interrupt control register. If any register does not respond to its read or write, an error message reports the error and specifies the register.

6.3.17.3.2 Test 2 - PI/T Register Addressing

This test verifies that all twenty-three registers of the PI/T chips on the EIA can be addressed without causing a bus-timeout error. This is accomplished by executing a bus read cycle to each register. If a bus timeout error does occur, an error message reports the error and specifies the register.

6.3.17.3.3 Test 3 - PI/T Register Read/Write

This test verifies that each bit-wise, writable and readable PI/T register can be written with a data pattern and then read back. This is done by initially writing a 55 hex to each register and then reading back each register, verifying that the data is the same. Upon one successful pass, the data is logically inverted and one more pass is made. Any data miscompares are reported by error messages. Only those registers that can be both written and read are tested.

6.3.17.3.4 Test 4 - PI/T Unique Register Address

This test verifies that each writable and readable PI/T register can be uniquely addressed with respect to the other registers. The test writes to each register using its address as data. After all registers have been written, each is read back and the data is verified. Any data miscompares are reported in error messages.

6.3.17.3.5 Test 5 - PI/T Reset Condition Status

This test verifies that all registers that should be cleared by a bus reset are in fact cleared. Each register being tested is written with an AA hex. A bus reset is then issued, and each register is checked to verify that is has been cleared. If any registers do not clear, they are reported in an error message.

NOTE

The RESET used in tests 5 and 6 is a system reset, and affects components other than the PI/T chip. In particular, this affects the MCS drive with a cartridge inserted. If the test is to be repeated several times, the cartridge should be removed, or the drive powered off, to avoid damage to the MCS.

6.3.17.3.6 Test 6 - PI/T Counter Zero Detect Status

This test checks the timer portion of the PI/T chips on the EIA board. A check is made as to whether or not the timer can detect a countdown to zero condition. The timer pre-load registers are loaded with a 0000FF hex. When the timer is started, the timer registers are decremented using clock path 3 and it assumes that a 2 mhz clock is on pin 40. The timer is started after which a software timer is started. The software timer uses a time delay which is known to be much longer than the delay in the PI/T timer. The software timer waits for either a bit set in the timer status register, signifying a zero countdown detected status, or will time out. If the software timeout occurs, then an error is reported.

If the bit is set in the timer status register, a check is done to see if a reset clears the bit (see note in 6.3.14.3.5). This is done by issuing a reset and then checking for zero in the timer status register.

6.3.17.3.7 Test 7 - PI/T Timer Countdown Interrupt

This test is similar to the first part of test 6, except that prior to enabling timer countdown, CPU interrupt priority is lowered to level zero. Upon countdown, the occurrence of an interrupt is verified. If no interrupt is detected, an error is reported.

6.3.17.3.8 Test 8 - PI/T External Wraparound

This is a manual intervention test, and requires that the external loopback connector is installed on the parallel port.

The test verifies the ability of the PI/T to send and receive parallel data. The PI/T is programmed for mode 2 with port A as the output and port B as the input. Next, using the loopback connector, a byte of data is sent out port A and read back on port B. Any data miscompares are reported.

6.3.17.3.9 Test 9 - PI/T External Wraparound Interrupt

This is a manual intervention test, and requires that the external loopback connector is installed on the parallel port.

The test is similar to test 8, except that PI/T interrupts are enabled and the CPU interrupt priority is lowered to level zero prior to data transfer, allowing interrupts to occur. Upon completion, the occurrence of an interrupt is verified. If an interrupt is not detected, an error is reported.

6.3.17.3.10 Test 10 - PI/T Parallel Printer

This is a manual intervention test, and requires that a parallel printer is connected to the parallel port.

The test verifies that the PI/T can function as a parallel printer controller. The chip is programmed for mode 0, pulsed submode. Next, all printer status bits are checked to see if the printer is on-line, selected and loaded with paper. When these conditions are verified, output to the printer is started in the form of all printable ASCII characters with each line consisting of the same character.

The expected printer interface can be set by the **INT_FACE** command, and the number of characters per line by the **CHAR_CNT** command (Section 6.3.15.4).

6.3.17.3.11 Test 11 - PI/T Interrupt Driven Parallel Printer

This is a manual intervention test, and requires that a parallel printer is connected to the parallel port.

This test is similar to test 10, except that the PI/T interrupts are enabled and the CPU interrupt priority is lowered to level zero prior to data transfer to allow interrupts to occur. All output to the printer is done within the interrupt service routine. Output is in the form of an ASCII ripple pattern. The test attempts to emulate the device-driver environment of an operating system.

6.3.17.3.12 Test 12 - PI/T Interrupt Driven Dual Parallel Printer

This is a manual intervention test, and requires that a parallel printer is connected to the parallel port.

This test is similar to test 11, except that it arbitrates between both of the PI/T chips, testing the arbitration circuitry and their mutual interaction.

6.3.17.3.13 Test 13 - EIB Read/Write Registers

This test verifies the basic communication of the EIB board by reading and/or writing to all of the registers on the EIB board. These registers include the EIB status/control register, the EIB Interrupt 7 status register, the EIB external power disable register, EIB Interrupt 6 status register, the EIB loopback register, and the EIBJON interrupt vector. If any register does not respond to its read or write, an error is reported specifying the register which is not communicating.

6.3.17.3.14 Test 14 - EIB Loopback Test

This test checks the data path between the CMB through the EIA to the EIB and back. The EIB is put into internal loopback mode and data is then written to the Expansion Interface Input buffer. This data is then sent through to the EIB and looped back by the EB to the Expansion Interface Output buffer. The data is then read and compared to verify that it has not been corrupted. Any data miscompares are reported.

6.3.17.3.15 Test 15 - EIA/EIB Parity Check Test

This test verifies the Parity Error detection code of the EIA and EIB boards. Within the EIA and EIB hardware there exists functionality to force a parity error while transferring data. In this test, the EIA and EIB are programmed to force this interrupt. The data is then transferred and the test checks that the proper parity error interrupt occurred.

6.3.17.4 EXPANSION INTERFACE TEST COMMANDS

The following commands can be entered directly at the EIAB test prompt to perform operations or set test parameters. These commands are in addition to the general diagnostic executive commands.

CHAR_CNT {*n*}

This command sets and displays the number of characters per line used in the printer tests 10 through 12. The argument *n* may be any value 1 to 132 (default is 80). If no argument is entered, the current number of characters per line is displayed.

INT_FACE {*n*}

This command sets and displays the interface to be used in printer tests 10 through 12. If no argument is specified, the current interface is displayed, followed by a menu:

1 = select Centronics Compatible Interface, (CCI)

2 = select Data Products Compatible Interface, (DPCI)

3 = select General Purpose Parallel Interface, (GPPI)

<cr> = no change

enter selection:

The command may be entered with the argument corresponding to the above menu options. The Centronics interface is initially selected at the start of the test.

PIT_SEL

This command specifies which of the two PI/T's on the EIA is tested by tests 2 through 12. If no argument is specified, the number of the current PI/T is displayed, followed by a menu:

1 = select EIA PIT 1

2 = select EIA PIT 2

<cr> no change

enter selection:

The command may be entered with the argument corresponding to the above menu options. PI/T 1 is initially selected at the start of the test.

QUICK {*n*}

This command invokes a quick-run of the test for a fast check of the EIA and PI/T registers (tests 1 through 7). The optional parameter *n*, when *n*=1, includes the manual intervention tests 8 and 9. Both PI/T chips are tested.

6.3.17.5 ERROR MESSAGES

The following error messages may be displayed during the test. The messages are listed according to the subtest displaying them.

6.3.17.5.1 Test 1 - Read/Write EIA Registers

- No EIA Status/Control Register

  This error indicates that a bus error occurred reading or writing to the EIA Status/Control Register.

- No EIA Level 7 Interrupt Status Register

  This error indicates that a bus error occurred reading or writing to the EIA Level 7 Interrupt Status Register.

- No EIA External Interrupt Control Register

  This error indicates that a bus error occurred reading or writing to the EIA External Interrupt Control Register.

6.3.17.5.2 Test 2 - PI/T Register Addressing

- ERROR!! Accessing a PI/T Register Has Caused a Bus Timeout Accessed Register Address XXXXXXXX

  This error indicates that a PI/T register is on the EIA is not communicating properly. The value of XXXXXXXX gives the contents of the register accessed.

6.3.17.5.3 Test 3 - Register Read/Write Test

- ERROR!! Expected Data vs. Actual Data
  Register Address XXXXXXXX Expected Data XX Actual Data XX

  This error indicates that the data written to a register was different that the data read back from the register. The register address and the data written and read are indicated.

6.3.17.5.4 Test 4 - Unique Register Address

- ERROR!! Unable to Uniquely Address a PI/T Register
  Register Address XXXXXXXX Expected Data XX Actual Data XX

  This error indicates that the data written to a register was different that the data read back from the register. The register address and the data written and read are indicated.

6.3.17.5.5 Test 5 - Reset Condition Status

- ERROR!! A Bus Reset Has Failed To Clear a PI/T Register
  Register Address XXXXXXXX Data After Reset XX

  This message reports which registers failed to clear following a bus reset, and the data they contained following the reset.

6.3.17.5.6 Test 6 - Counter Zero Detect Status

- ERROR!! Loading Counter Preload Registers And Enabling Time Countdown Should Set Bit <0> In Timer Status Register But Didn't
  Timer Status Register Address XXXXXXXX Timer Status XX

  This message indicates that the timer malfunctioned by failing to count down to zero. The register address and current data are displayed.

- ERROR!! A Bus Reset Has Failed To Clear a PI/T Register
  Register Address XXXXXXXX Data After Reset XX

  This message indicates that the timer status register failed to clear following a bus reset. The register address and current data are displayed.

6.3.17.5.7 Test 7 - Timer Countdown Interrupt Test

- ERROR!! Loading Counter Preload Registers And Enabling Timer Countdown Should Cause One And Only One Timer Interrupt To Occur, But Didn't
  Timer Interrupt Count XXXXXXXX

  This message indicates that the interrupt didn't occur, or that too many interrupts occurred. The message indicates number of interrupts received.

6.3.17.5.8 Test 8 - External Wraparound

- ERROR!! Received Data Not Equal to Sent Data

Output Register Address	Data Sent	Input Register Address	Data Received
XXXXXXXX	XX	XXXXXXXX	XX

  This message indicates that the data written was different from the data read back through the external loopback connector. The message also reports which register was used for output, which for input, and what the data in each was.

6.3.17.5.9 Test 9 - External Wraparound Interrupt Test

- ERROR!! Received Data Not Equal to Sent Data

Output Register Address	Data Sent	Input Register Address	Data Received
XXXXXXXX	XX	XXXXXXXX	XX

  This message indicates that the data written was different from the data read back through the external loopback connector. The message also reports which register was used for output, which for input, and what the data in each was.

- ERROR!! Upon Completion Of Data Transfer, One And Only One PI/T Interrupt Should Have Occurred But Didn't
  PI/T Interrupt Count XXXXXXXX

  This message indicates that the interrupt didn't occur, or that too many interrupts occurred. The message indicates number of interrupts received.

6.3.17.5.10 Test 10 - Parallel Printer Test

- Select Bit <5> Not Set In Printer Status Byte
  Printer Status Port Address XXXXXXXX Printer Status Byte XX

  This message indicates that the proper select bit in the printer status byte was not set, and displays the status.

- Paper Empty Bit <3> Set In Printer Status Byte
  Printer Status Port Address XXXXXXXX Printer Status Byte XX

  This message indicates that the printer is signalling that it is out of paper.

- Busy Bit <7> Set In Printer Status Byte
  Printer Status Port Address XXXXXXXX Printer Status Byte XX

  This message indicates that the printer is signalling that it is busy.

- Timed Out Waiting For Printer To Empty PI/T Output Latch
  Output Latch Status Byte XX

  This message indicates that the PI/T chip either did not finish emptying its output latch or did not correctly signal that it was empty within the allotted time.

- Timed Out Waiting For Busy To Drop In Printer Status Byte
  Output Latch Status Byte XX

  This message indicates that the printer signalled that it was busy for longer than is allotted.

6.3.17.5.11 Interrupt Driven Parallel Printer Test

- Select Bit <5> Not Set In Printer Status Byte
  Printer Status Port Address XXXXXXXX Printer Status Byte XX

  This message indicates that the proper select bit in the printer status byte was not set.

- Paper Empty Bit <3> Set In Printer Status Byte
  Printer Status Port Address XXXXXXXX Printer Status Byte XX

  This message indicates that the printer is signalling that it is out of paper.

- Busy Bit <7> Set in Printer Status Byte
  Printer Status Port Address XXXXXXXX Printer Status Byte XX

  This message indicates that the printer is signalling that it is busy.

- Timed Out Waiting For Printer To Empty PI/T Output Latch
  Printer Status Port Address XXXXXXXX Printer Status Byte XX

  This message indicates that the PI/T chip either did not finish emptying its output latch or did not correctly signal that it was empty within the time allotted. This error comes from the interrupt driver routine.

- Timed Out Waiting For Busy To Drop Printer Status Byte
  Printer Status Port Address XXXXXXXX Printer Status Byte XX

  This message indicates that the printer signalled that it was busy for longer than the time allotted. This error comes from the interrupt driver routine.

6.3.17.5.12 Test 12 - Interrupt Driven Parallel Printer Test

- Select Bit <5> Not Set In Printer Status Byte
  Printer Status Port Address XXXXXXXX Printer Status Byte XX

  This message indicates that the proper select bit in the printer status byte was not set.

- Paper Empty <3> Set In Printer Status Byte
  Printer Status Port Address XXXXXXXX Printer Status Byte XX

  This message indicates that the printer is signalling that it is out of paper.

- Busy Bit <7> Set in Printer Status Byte
  Printer Status Port Address XXXXXXXX Printer Status Byte XX

  This message indicates that the printer is signalling that it is busy.

- Timed Out Waiting For Printer To Empty PI/T Output Latch
  Printer Status Port Address XXXXXXXX Printer Status Byte XX

  This message indicates that the PI/T chip either did not finish emptying its output latch or did not correctly signal that it was empty within the time allotted. This error comes from the interrupt driver routine.

- Timed Out Waiting For Busy To Drop Printer Status Byte
  Printer Status Port Address XXXXXXXX Printer Status Byte XX

  This message indicates that the printer signalled that it was busy for longer than the time allotted. This error comes from the interrupt driver routine.

6.3.17.5.13 Test 13 - Read/Write EIB Registers

- No EIB Status/Control Register

  This message indicates that a bus error occurred while reading or writing to the Control/Register on the EIB board.

- No EIB Interrupt Level 7 Status Register

  This message indicates that a bus error occurred while reading or writing to the Interrupt Level 7 Status Register on the EIB board.

- No EIB External Power Disable Register

  This message indicates that a bus error occurred while reading or writing to the Interrupt Level 7 Status Register on the EIB board.

- No EIB External Power Disable Register

  This message indicates that a bus error occurred with reading or writing to the External Power Disable Register on the EIB board.

- No EIB Interrupt Level 6 Status Register

  This message indicates that a bus error occurred while reading or writing to the Interrupt Level 6 Status Register on the EIB board.

- No EIB Loop Back Register

  This message indicates that a bus error occurred while reading or writing to the Loop Back Register on the EIB board.

- No EIBJON Interrupt Vector

  This message indicates that a bus error occurred while reading or writing to the EIBJON Interrupt Vector on the EIB board.

6.3.17.5.14 Test 14 - EIB Loopback Test

- EIB Failed Loopback Test

  This message indicates that an error exists in the path from the EIA through the Expansion Interface cables to the EIB and back again. The cables may be faulty or not connected properly, or the EIB is faulty.

6.3.17.5.15 Test 15 - EIA/EIB Parity Check Test

- No Interrupt Received from a Forced EIB Parity Error

  This message indicates that the EIB did not properly issue a parity error interrupt when a parity error was forced. The EIB is faulty.

- Bad Status Received from a Forced EIB Parity Error
  Expected value: 0Bh Received value: XXh

  This message indicates that an interrupt was received from the EIB due to a forced parity error, but that the status from the EIB associated with the interrupt is invalid. The message displays the value received. The EIB is faulty.

- Bad Status Received from a Forced EIA Parity Error
  Expected value: 3h Received value: XXh

  This message indicates that an interrupt was received from the EIA due to a forced parity error, but that the status from the EIA associated with the interrupt is invalid. The message displays the value received. The EIA is faulty.

6.3.18 Dual SCSI Controller Logic Test

Beginning in release 7.5B, SCSI controllers are supported on MAI 3000/4000 systems. The Dual SCSI Controller (DSC) Logic Test verifies the correct functioning of the SCSI controller and an attached disk drive. The test provides comprehensive testing of the SCSI controller. Several of the tests focus on the SCSI bus controller chip (SBIC). Testing of an attached disk drive is not comprehensive, but does help verify its functioning and in isolating faults.

6.3.18.1 Loading and Running the SCSI Controller Logic Test

The Dual SCSI Controller Logic Test is loaded and run in the usual manner for logic test. At either the diagnostic executive prompt, or at the prompt for another logic test, enter:

```
load dsc
```

To run the logic tests in normal mode, enter:

```
run
```

Refer to the command descriptions in Section 6.3.18.4 for additional tests run options.

6.3.18.2 Test Initialization

When the test is loaded, an initialization routine runs that tests for the presence and proper communication of the SCSI controller and the CMB.

The initialization routine performs a SIZE operation to determine whether there are any Winchester Disk or SCSI controller boards installed. If there are, several handshake processes are executed to verify that the I/O addresses of Winchester Disk and DSC controllers do not overlap. When the handshaking matches the handshaking expected for the appropriate controller, the controller is tagged as such. If there is some deviance from the expected handshaking, the software flags the controller involved with ERROR. The results of these checks are displayed in a table. If there are errors, the logic tests will fail. The first three or four tests may appear to succeed, but their results will be unreliable.

If the sizing operation is successful, the test prompt is displayed:

```
<dsc>
```

The test defaults the following conditions:

- Test all SCSI buses (2 per DSC)
- Skip manual intervention tests
- Skip write protected tests
- Only test devices at SCSI ID 5
- Only test SCSI disk devices

These test conditions can be changed using the commands supplied.

6.3.18.3 DUAL SCSI CONTROLLER TEST DESCRIPTIONS

6.3.18.3.1 Test 1 - DSC Register Read Test

This test verifies that the DSC I/O registers can be read without causing a bus error. Only the controller is required. The following registers are checked:

- SBIC aux status register (base address + 1)
- SBIC register file vector (base address + 2)
- Interrupt vector (base address + 5)
- Control register (base address + 7)
- DSC status register (base address + 12)

Failure of this test indicates a hardware addressing fault or that the DSC board is not installed.

6.3.18.3.2 Test 2 - Write/Read DSC Control Register

This test alternately writes and reads back several data patterns to the DSC control register and verifies that the register retained the pattern properly. The patterns test all bits for being stuck high, stuck low, or shorted adjacent bits. Only the controller is required.

This test may fail if test 1 fails

6.3.18.3.3 Test 3 - DSC Channel Reset

This test checks the SCSI bus reset. A reset command is issued, and then checks for the following:

- DSC status is 40h
- SCSI status is 0
- The SBIC successful completion interrupt is generated.

If test 1 or 2 fail, this test may fail also. An error indicates either a faulty SBIC, control logic or bus interface. If this test fails on both controller channels, the fault is likely to be in the bus interface or the common control logic.

6.3.18.3.4 Test 4 - Write/Read DSC Vector Register

This test verifies that the DSC vector register will accept any exception vector index value. The test writes, reads and compares a series of data patterns to check all bits for being stuck high, stuck low, or shorted with adjacent bits.

If test 1 fails, this test may also fail.

6.3.18.3.5 Test 5 - Write/Read SCSI Bus Chip Own ID Register

This test verifies that the Own ID register (register 0 of the SCSI bus chip register file) accepts any value it is given. The test writes, reads and compares a series of data patterns to check all bits for being stuck high, stuck low, or shorted with adjacent bits.

If test 1 fails, this test may also fail. An error may indicate a faulty SBIC or control hardware leading to the chip.

6.3.18.3.6 Test 6 - Write/Read SCSI Bus Chip Register File

This test verifies that the first 23 registers (of 26 1-byte registers) can be accessed, written and read. A series of data patterns is written and verified.

If test 1 fails, this test may also fail. Failure of this test generally indicates a bad SBIC or associated logic.

6.3.18.3.7 Test 7 - Diagnostic DMA Test

This test verifies that the DSC is able to perform DMA transfers from the SCSI device. The test verifies that all data paths are good by performing a DMA write, read and verify. The DSC uses a 4-byte FIFO buffer to translate between the word protocol of the EBUS and the byte protocol of the SBIC.

If tests 1, 2 and 6 pass and this test fails, the error most likely resides in the data path logic or in the bus interface.

6.3.18.3.8 Test 8 - Diagnostic DMA Odd Byte Strobe

If an odd number of bytes are received from the host for a DMA transfer, the last byte remains in the FIFO buffer. An odd byte transfer is used to flush the last byte. This test verifies that the task can be completed correctly.

If tests 1. 2, 6 and 7 pass and this test fails, the fault most likely resides in the data path logic or in the bus interface.

6.3.18.3.9 Test 9 - Bus Error on Invalid DMA Address

This test attempts to do a DMA transfer to a non-existent address (FFFFFFF hex) and verifies that the DSC will post the resulting bus error.

If tests 1, 2, 6 and 7 pass and this test fails, there is a problem in the logic that generates a bus error on the DSC bus.

6.3.18.3.10 Test 10 - Parity Error on DMA Transfer

This test forces a parity error within a block of data, and verifies that the DSC detects it upon a DMA transfer.

If tests 1, 2, 6 and 7 pass and this test fails, there is a problem in the logic that generates a bus error on the DSC bus.

6.3.18.3.11 Test 11 - Clear DMA Bus Error Interrupt Pending

This test checks the ability of the controller to clear pending interrupts. The test first disables interrupts, then forces a bus error, as in test 10, causing a pending bus error interrupt. The test then strobes the DSC bus error clear register to remove the pending interrupt, and enables interrupts, delays for 5 bus cycles and again disables interrupts. A fault occurs is an interrupt is generated.

If test 1, 2, 6 and 7 pass and this test fails, the either the clear pending interrupts logic or the disable interrupts logic is failing.

6.3.18.3.12 Test 12 - DMA Address Register

This test verifies that the DMA address register is functioning properly. The test performs DMA transfers at locations which test each bit of the DMA register for stuck low and stuck high conditions.

If test 1, 2, 6 and 7 pass and this test fails, the DSC DMA address register is bad.

6.3.18.3.13 Test 13 - DSC Interrupt Vectors

This test verifies that all interrupt vectors C0 through FF can be used by the DSC. The vectors must respond to both a bus error interrupt and a reset complete interrupt.

This test aborts is any of tests 1-4, 6-8, 10 or 11 fail. If all previous tests pass and this test fails to generate an interrupt in both cases, the fault is in the interrupt control logic. If the fault occurs for only one type of interrupt, then the problem is in the logic that detects the condition to initiate that interrupt. If the error occurs about 12 percent of the time, then the error is in the DSC vector register.

6.3.18.3.14 Test 14 - SBIC Reset Command

This test verifies the ability of the SBIC to execute a Reset command. The test first disables interrupts and then performs a SCSI reset. The SBIC auxiliary status register is then checked for a pending interrupt. If this succeeds, the test then verifies that the status is 0.

This test aborts if any of tests 1, 2, 4, 5, 6 or 12 fail. If these tests pass and test 14 fails, the SBIC is probably bad.

6.3.18.3.15 Test 15 - Select and Transfer Simple Mode

This is a manual intervention test, and requires a loopback connector between the two SCSI buses on the DSC. The test verifies that the SBIC is able to perform basic simple mode SCSI operations, and also verifies the integrity of the SCSI bus data path.

This test aborts if any of the previous tests fail. If this test fails, either the controller board or a cable is bad.

6.3.18.3.16 Test 16 - Select and Transfer Combination Mode

This test is identical to test 15, except that it runs in SCSI combination mode.

6.3.18.3.17 Test 17 - Size/Inquire SCSI Devices

This test attempts to determine if any devices exist on the active SCSI bus at any ID from 0 to 6. If a device responds, the test attempts to determine the device type (disk, tape or unknown). If a timeout occurs during the inquiry phase, it is assumed that no device exists.

This test aborts if any of tests 1-14 fail.

6.3.18.3.18 Test 18 - Sense Command

This test verifies that the Test Unit Ready and Sense commands are functioning properly. The test first issues a Reset followed by the Sense command. The data returned should have a sense key of 6 hex and an extended sense key of 29 hex. A second Sense command is then issued and should return both a sense key and extended key value of 0.

This test aborts if any of tests 1-14 fail. If tests 1-17 pass and this test fails, the device is probably bad. If tests 1-14 pass and 15-18 fail, the SCSI bus connector may be bad or not making proper contact. If all tests have failed, the DSC is probably bad or configured incorrectly.

6.3.18.3.19 Test 19 - Test Unit Ready

This tests executes a Test Unit Ready command on the active ID of the current SCSI bus. If the command times out, a Sense command is used to attempt to isolate the problem.

This test aborts if any of tests 1-14 fail. If tests 1-17 pass and this test fails, the device is probably bad. If tests 1-14 pass and 15-19 fail, the SCSI bus connector may be bad or not making proper contact. If all tests have failed, the DSC is probably bad or configured incorrectly.

6.3.18.3.20 Test 20 - Rezero Unit

This test executes the Rezero Unit command, which repositions the drive access arm over cylinder 0. A Sense command is used to verify positioning.

This test aborts is any of tests 1-14 or 17 fail. If previous tests pass and this test fails, the problem is in the drive access arm or the read/write head mechanism.

6.3.18.3.21 Test 21 - Verify Superblock

This test reads the disk superblock, verifying that the data path from the computer to the disk is valid, that the superblock is valid, and that the hard disk is able to perform mechanically.

This test aborts if any of tests 1-14 or 17-20 fail. If previous tests pass but test 21 fails, either the disk if not formatted or has been corrupted, or the drive is faulty.

6.3.18.3.22 Test 22 - Seek

This test verifies the ability of the disk to execute Seek commands to specified locations. The test first reads the superblock to determine the maximum logical block address. It then seeks to determined locations up to the maximum, then back to the minimum.

This test aborts if any of tests 1-14 or 21 fail. If the previous tests pass and test 22 fails, the disk is bad.

6.3.18.3.23 Test 23 - Random Seek

This test verifies the ability of the disk to execute Seek commands to random locations for an extended period of time. The test first reads the superblock to determine the maximum logical block address. It then seeks to locations determined by a random number generator. Each pass performs 127 seeks.

This test aborts if any of tests 1-14 or 21-22 fail. If the previous tests pass and test 23 fails, the disk is bad.

6.3.18.3.24 Test 24 - Send Diagnostic

This test causes the device to perform its diagnostic self-test.

This test aborts if any of tests 1-14 or 17 fail. Test results are interpreted as for test 18.

6.3.18.3.25 Test 25 - Read Buffer

This test verifies that the SCSI Read Buffer command can access the SCSI device buffer, and verifies that the buffer maintains data integrity over a period of time.

This test aborts is any of tests 1-14 fail. If tests 1-14 pass and test 25 fails, the device buffer or the cable is bad; use the manual intervention tests to determine which.

6.3.18.3.26 Test 26 - Write Buffer

This test verifies the SCSI device buffer can be accessed by the SCSI Read Buffer and Write Buffer commands, and that the device RAM is good. The test writes random data patterns to the device buffer using the Write Buffer command, then reads the data back with the Read Buffer command and compares the data.

This test aborts if any of tests 1-14 or test 25 fails. If these tests pass and test 26 fails, the device or cable is bad; use the manual intervention tests to determine which.

6.3.18.3.27 Test 27 - Read Disk

This test verifies that the disk device has a valid superblock, and exercises read/write head motion. The test first reads the superblock to determine the minimum and maximum cylinders. It then performs twenty rapid cycles of alternately reading the minimum and maximum cylinders.

This test aborts if any of tests 1-14 fail.

6.3.18.3.28 Test 28 - Write Disk

This test verifies that the disk device has a valid superblock (as in test 27), then tests the disk's ability to perform random data write/read/verify operations. All write/read operations are performed on the diagnostic cylinder. All write/read heads are tested.

This test aborts if any of tests 1-14 or tests 17-27 fail. If these tests pass and test 28 fails, the problem is either a cross-talk problem on the bus or a failure of the write/read heads. In this case, the disk should be backed up immediately. Do not attempt any further write operations to the disk, since this could result in further data corruption.

6.3.18.3.29 Test 29 - Mode Sense/Mode Select

This test verifies that the format parameters and the rigid disk drive geometry parameters returned by the SCSI Mode Sense command are syntactically correct. If the parameters are correct, the test then verifies that the Mode Sense command is functioning correctly by writing patterns to the error recovery page with the SCSI Mode Select command and verifying the data using the SCSI Mode Sense command.

This test aborts is any of tests 1-14 or test 24 fails. If tests 1-14 pass and test 29 fails, the drive or cable are probably bad; use the manual intervention tests to determine which.

6.3.18.3.30 Test 30 - Random Extended Seek

This test performs random seeks within the legal range of block addresses, as determined from the superblock. The test uses a random number generator to determine values for the seeks. The test serves as a check for intermittent failure.

This test aborts if any of tests 1-14 or 17-24 fail. If all previous tests pass and this test fails, the hard disk is bad.

6.3.18.3.31 Test 31 - Long Read

This test performs 32 passes of an alternate incremental read sequence (blocks 1,2,1,4,1,8 ...) within the legal block range, as determined by reading the superblock. This sequence exercises all heads, and increases the likelihood of detecting a failure of a marginal disk.

This test aborts if any of tests 1-14 or 25-27 fail. If this test fails, do not attempt any further write operations. Failure of this test indicates a failing disk device.

6.3.18.3.32 Test 32 - Extended Write to Disk

This test performs write/read/verify test sequences using the diagnostic cylinder. Using data from the superblock to determine the size of the diagnostic cylinder, the test writes, reads and compares data for the entire cylinder. In this way, all write/read heads are tested.

This test aborts if any of tests 1-14, or 17-31 fail. If these tests pass and test 32 fails, there is either a cross-talk problem on the bus or there is a problem with the write/read heads of the disk device. If the test fails, the disk should be backed up, and no further write tests should be attempted.

6.3.18.3.33 Test 33 - Verify

This test verifies that the SCSI Verify command is functioning properly.

This test aborts if any of tests 1-14, 17-22 or 24-25 fail.

6.3.18.3.34 Test 34 - Read Defect Data

This test checks for consistency of the results of the SCSI Read Defect Data command. The test first reads the data from the primary defect list and the growth defect list into the read buffer. It then performs ten loops of reading the primary and growth defect lists into the write buffer and comparing the data in the read and write buffers, which should be identical.

This test aborts if any of tests 1-14, 17-21, 24, 25 or 27 fail.

6.3.18.3.35 Test 35 - Start/Stop Unit

This test verifies the proper functioning of the SCSI Start and Stop commands to take the drive on-line and off-line. It also checks the ability of the drive to spin up to speed within the allowed 30 seconds.

This test aborts if any of tests 1-14 or 17-19 fail.

6.3.18.3.36 Test 36 - Reserve/Release

This test verifies that when an initiator reserves the drive, no other initiator can access the drive. It then verifies that when the initiator releases the drive, other initiators can again gain access to the drive.

This test aborts if any of tests 1-14 or 17-27 fail. If all previous tests pass and test 36 fails, the firmware controlling the SCSI Reserve and Release commands is malfunctioning.

6.3.18.4 SCSI TEST COMMANDS

The following commands can be entered directly at the DSC test prompt to perform operations or set test parameters. These commands are in addition to the general diagnostic executive commands.

ADDRESS {*n*}

This command selects or displays the current SCSI bus. If no parameter is specified, the base address of the current SCSI bus is displayed. If the parameter is specified, it must be a valid SCSI bus base address, as reported by the SIZE command.

BERRC

This command writes to the Clear Bus Error/DMA Reset address of the currently selected SCSI bus, clearing any posted bus error and initializing the DMA direction.

BUSRESET

This command issues a SCSI bus reset command. SCSI bus or device hangs can often be cleared with this command.

CCHANNEL

This command issues a SCSI bus reset followed by a controller channel reset. The command also causes a successful completion interrupt to occur.

CCONTROL

This command reset the SCSI controller chip, but does not reset any devices attached to the SCSI bus.

DISK {*n*}

This command enables or disables access to the disk by tests 17-36. If no parameter is specified, the current status is displayed (0 = disabled; 1 = enabled). **DISK 0** disables access to the disk; **DISK 1** enables access to the disk. The status report includes the status for SCSI tape, which is not supported and may be ignored.

DSC {*n*}

This command selects or displays the current DSC. The diagnostic default is to test all DSC controllers. If a controller is specified, only that controller is tested. If no parameter is specified, the currently active bus is displayed.

ID {*n*|A}

This command instructs tests 17-36 to run on only devices configured to the specified ID (0-6). The parameter A specifies ALL IDs. If no parameter is specified, the currently selected ID is displayed.

INQUIRE

This command requests the SCSI device on the current SCSI channel to return information about its device type, model number and revision number.

LPORT {*n*}

This command selects or displays the currently active SCSI bus. The diagnostic default tests all SCSI buses. When a parameter (0-F) is specified with this command, only that bus will be tested. When the command is entered without a parameter, the current bus is displayed.

NOPROTECT

This command enables tests that write to the user area of disk. The diagnostic default disables disk write tests. Tests that write only to the diagnostic cylinder can run even with disk write tests disabled.

PORT {*n*}

This command selects or displays the currently active SCSI bus on the currently selected DSC. The diagnostic default tests all SCSI buses. When a parameter (A or B) is specified with this command, only that bus will be tested. When the command is entered without a parameter, the current bus is displayed.

PROTECT

This command disables tests that write to the user area of disk. The diagnostic default disables disk write tests. Tests that write only to the diagnostic cylinder can run even with disk write tests disabled.

RECALIBR

This command issues a SCSI Rezero Unit command, which assures that the heads return to a known position. The command is useful to determine if the heads are stuck, and may unstick them.

REGISTER {*a* {*b*}}

This command provides a way to manually program the SBIC. This requires knowledge of the programming sequences, and is generally not required for field diagnostic purposes.

REGS

This command displays a general description of the DSC I/O addresses, as well as the state of the status, control and vector registers.

SENSE

This command issues the SCSI Sense command to the currently active SCSI device, and displays the sense key and extended key values.

SIZE

This command attempts to determine whether there are any Winchester Disk Controller boards or DSCs present in the system, and then determines whether there is any address overlap between these. Refer to the test initialization description (6.3.18.2) for a description of this process.

STATUS

This command displays the status of the SBIC for the currently selected SCSI bus.

SUMMARY

This command displays a simple pass/fail report of the results of the last pass of the DSC logic test. For each tests, the report indicates either PASS, FAIL or ABORT, or leaves the entry empty if the test was not requested.

TAPE {*n*}

This command determines whether tests 17-36 will attempt to access exiting SCSI tape devices. If no parameter is specified, the current status is displayed. Used with a parameter, 0 disables tape tests, 1 enables tape tests. The diagnostic default enables tape tests.

UNIT {*a b*}

This command selects or displays the ID and device types of the devices on the currently selected SCSI bus. The first parameter, a, is the ID and must be in the range 0-6. The second parameter, b, specifies the device type: 0 = none, 1 = disk, 2 = tape. If no parameter is specified, the current device ID and type are displayed.

USTATUS

This command issues the SCSI Test Unit Ready command. If the command is unsuccessful, a hardware time-out error is returned. There is no returned status if the command is successful.

6.3.18.5 ERROR MESSAGES

Errors detected by the logic tests are reported by the following error messages. Additional information may be included, giving further details of the error. The message codes are described in tables following the error message descriptions.

- SCSI Not Ready Prior to Issuing Command!

DSC Status	Command
xx	*yy*

  This message indicates that the SCSI device was not able to respond when the SCSI command yy was issued.

- Command Aborted Before Completion

DSC Status	Command
xx	*yy*

  This message indicates an unexpected condition on the SBIC.

- With Interrupts Enabled, No Interrupt Occurred!

Status	Command	Control Reg	Vector Reg
xx	*yy*	*zz*	*aa*

  A SCSI command was executed with interrupts enabled, but either the CMB did not detect an interrupt when the command completed or the interrupt vector was not initialized properly. If the Status register indicates no pending interrupt but the control register indicates interrupts are enabled, there may be a timing problem with the DSC.

- With Interrupts Disabled, An Interrupt Occurred!

Status	Command	Control Reg	Vector Reg
xx	*yy*	*zz*	*aa*

The CMB detected an interrupt when interrupts were disabled. If the DSC command register indicates that interrupts are in fact disabled, either the board is generating erroneous interrupts or another device is generating interrupts on the same address.

- Buffers Don't Compare

Write	Data	Read	Data
ww	*xx*	*yy*	*zz*

A write/read test produced different data on the write and read cycles. The message indicates the addresses and data for the write and read buffers.

- SCSI Controller Address Bus Error

Fault Address
xx

This message indicates that the DSC I/O address displayed generated a bus error when read by the CMB.

- Unit Write Protected, Enter The NOPROTECT command

The test being run requires that the logic test be in NOPROTECT mode.

- Bad Status Found Before Command Completion

DSC Status	Command	DSC Cont Reg
xx	*yy*	*zz*

The SCSI command was prematurely terminated by some unexpected condition. The cause could be either a bad DSC, a bad SCSI device, or a program error.

- Expected Bus Error Status Not Found

DMAddress	DSC Status
xxxxx	*yy*

A DMA operation was attempted with an intentionally bad address, but did not causer a bus error condition. The address and condition actually returned are shown.

- Sense Data From Check Status

ERR-KEY	CMD	Block	Target	Unit	Ext. Sense
xx	*yy*	*zz*	*aa*	*bb*	*cc*

An unexpected check condition occurred during the execution of a SCSI transfer command. The sense data and extended sense data explain what caused the condition.

- Sense Data From Check Status

ERR-KEY	CMD	Target	Unit	Ext. Sense
xx	*yy*	*aa*	*bb*	*cc*

An unexpected check condition occurred during the execution of a SCSI command. The sense data and extended sense data explain what caused the condition.

- Unexpected Controller Detected Bus Error
  DMAddress DSC Status
  xxxxx *yy*

  An unexpected bus error occurred during a DMA bus transfer. The remainder of the DMA transfer is cancelled. The DSC Status indicates the cause of the error.

- Controller Detected Bus Error Didn't Generate Interrupt
  Status Reg. Vector Reg.
  xx *yy*

  The test forced a bus error condition while interrupts were enabled to verify that the error would generate an interrupt, but the DSC did not generate the interrupt. Either the DSC did not detect the condition or did not generate the interrupt, or the vector register was not initialized.

- Error Processing: SENSE Command Failure

  An error occurred while processing a SCSI command, but the error information also contains errors and so cannot be processed.

- SBIC Register Loopback Error
  Reg. Expected Received
  xx *yy* *zz*

  A data compare error occurred while testing the DSC register file. This usually indicates a bad SBIC.

- Bad DSC Status After Reset or Bus Error Clear
  DSC Status
  xx

  After performing a reset or bus error clear, the DSC status should have been 40 hex. The actual status is shown.

- Non-zero SCSI Status After Reset
  SCSI Status
  xx

  After performing a bus reset, the SENSE command should have returned a zero status. The actual status returned is shown.

- Reset Did Not Generate SCSI Interrupt

  After performing a SCSI bus reset, the DSC should either generate a SCSI bus reset interrupt or have the interrupt pending. Neither situation occurred.

- Timeout Waiting For FIFO Not Busy

  During a diagnostic DMA transfer, the FIFO Busy condition was detected from the DSC Status Register for longer than is allowed.

- Compare Error
  Expected Received
  xx *yy*

  A data miscompare occurred during the test. The expected and actually received data are displayed.

- Data Compare Error
  Expected Received
  xx *yy*

  A data miscompare occurred during the test. The expected and actually received data are displayed.

- Disk Superblock Invalid

  The disk superblock is syntactically incorrect. The usual causes are an unformatted or corrupted disk.

- SCSI Select Failed

  While attempting to execute a SCSI command in simple mode, the SCSI bus selection phase failed. The probably cause is either the device does not exist or is intermittently failing.

- DMA Test Data Compare Error
  Expected Received
  xx *yy*

  A data miscompare occurred during a diagnostic DMA between buffers. The original and received data are shown.

- Vector Reg Data Loop-Thru Error
  Expected Received
  xx *yy*

  A data miscompare occurred during the Write/Read Vector Register test. The original and received data are shown.

- Control Reg Data Loop-Thru Error
  Expected Received
  xx *yy*

  A data miscompare occurred during the Write/Read DSC Control Register test. The original and received data are shown.

- Data Registers Loop-Thru Error
  Expected Received
  xx *yy*

  A data miscompare occurred during the Write/Read SBIC Register File test. The original and received data are shown.

- Modesense Data Doesn't Match the Superblock

  During the Mode Select/Mode Sense Test, fields from each page of data are compared to fields in the superblock. This message indicates that the fields do not match.

- Testing Aborted

  The DSC test automatically aborts after 16 errors.

- DMA Parity Error Expected, Not Received

Address	CMB status	CMB parity reg.
xxxxx	*yy*	*zz*

  The test forced a parity error, but it was not detected by the DSC during the DMA transfer.

- SBIC Register File Compare Error

Register	Expected Value	Received Value
xx	*yy*	*zz*

  A data miscompare error occurred during the Write/Read SBIC test.

- DSC Clear Bus Error Strobe Failed to Clear Pending Interrupt

  The attempt to clear a pending bus error interrupt failed during the Clear DMA Bus Error Interrupt Pending test.

- Bus Error Did Not Generate an Interrupt

  An attempt to generate a pending bus error interrupt failed during the Clear DMA Bus Error Interrupt Pending test.

- Channel Reset Did Not Generate An Interrupt

DSC Status	SBIC Status
xx	*yy*

  The DSC Channel Reset test failed to cause the SBIC to generate an interrupt.

- SBIC Reset Command Did Not Generate SBIC Interrupt

  The SBIC Reset Command test failed to cause the SBIC to generate an interrupt.

- SBIC Reset Command Did Not Generate DSC Interrupt

  The SBIC Reset Command test caused the SBIC Interrupt line to become asserted, but the interrupt never generated a DSC interrupt.

- Unknown Device Error Detected

ID
xx

  While sizing the units attached to the SCSI bus, a responding device indicated that it was either not a disk or tape, or the device was not valid.

- Interrupt Timeout

Driver Error Code
xx

  Some phase of a SCSI command did not complete within its allotted time.

- Bad Status After Interrupt
  Driver Error Code
  xx

  After a operation was performed, the SCSI status was not what it should have been.

- Data Available Timeout
  Driver Error Code
  xx

  During a SCSI command's data out transfer phase, the target device did not respond within the allotted time.

- Non-Zero Status or Message Byte
  Driver Error Code
  xx

  Either the status phase or the message phase of the last SCSI command failed.

- Parity Error
  Driver Error Code
  xx

  A parity error occurred during some phase of a SCSI command.

- Sense Command Generated Erroneous Data

Exp. sense	Rec. sense	Exp Extended sense	Rec. Extended sense
ww	*xx*	*yy*	*zz*

  The Sense Command test received either incorrect sense data or incorrect extended sense data regarding the last SCSI command.

- Device Not Ready

ERR-KEY	CMD	Target	Unit	Ext.Sense
aa	*bb*	*xx*	*yy*	*zz*

  The SCSI Test Unit Ready command generated a Device Not Ready check condition. This condition could occur if the unit is trying to complete another command or has become hung. Perform a channel reset and try the test again.

- Invalid Data Returned From Modesense Command
  Modesense Page
  xx

  The Modesense data returned for the indicated page is syntactically incorrect.

- Compare Error In Page 1 Data After Mode Select

  The last data written to page 1 with the mode select command was not consistent with the most recent data read back.

- Expected a Reservation Conflict. Received following:

END-STATUS	ERR-KEY	EXT. SENSE
xx	*yy*	*zz*

  The test attempted to force a SCSI Own ID conflict, but rather than indicate the conflict, the unexpected condition indicated was detected.

- SBIC Reset Failed

  This message indicates that the SBIC did not respond when a reset operation was performed.

- SCSI Bus Busy

  This message indicates that the SCSI bus is busy when a Level II command is being executed by the SBIC.

- Timeout For Operation To Complete

  During a SCSI command phase, the target device did not respond to the initiator within the allotted time.

- SCSI Select Timeout By Hardware

  A timeout occurred while waiting for the device to respond during either the Select or Reselect phase of a SCSI command. This timeout is controlled by the SBIC timeout period register, rather than by software.

- Invalid SCSI End-Status

End-Status	Command
xx	*yy*

  The Sense Command received either sense data or extended sense data that is not possible for the last SCSI command performed.

- SCSI Reservation Conflict

End-Status	Command
xx	*yy*

  An unexpected SCSI reservation conflict occurred while trying to perform a SCSI command.

- Device Busy

End-Status	Command
xx	*yy*

  During an attempt to perform a SCSI command, a check condition was returned from the command indicating that the target device is busy. The command is aborted when this condition occurs.

- PIT Timer Failed to Start

  This message indicates that the PIT timer is bad. Run the PIT logic test to determine if this is true.

- Unexpected Interrupt
  DSC Status
  xx

  The SBIC interrupt service routine was called, but the routine could not determine the source of the interrupt.

- SCSI Bus Parity Error

  A SCSI Bus Parity Error was detected during the execution of a SCSI command.

- Interrupt With No SBIC Source

  The SBIC Interrupt Service routine was called, but the routine found that the SBIC did not post an interrupt.

The following messages indicate a bug in the logic test program. There is no corrective action, but the problem should be reported.

- Block Selected Exceeds Min/Max Range

- DCB Not Defined

- Unexpected Sequence Or Program Error

- Programmer Error In Call to Mini DSC Driver
  Driver Error Code
  xx

- Error Code Not Found in Routine Mini-Error:
  Driver Error Code
  xx

- User Abort Before Command Complete!

- End Status Non-Zero -- Auto Sense Issue Disabled

End-Status	Command
xx	*yy*

Table 6-2. SCSI Commands

Command	Decimal	Hexadecimal
Test Unit Read	00	00
Rezero Unit	01	01
Request Sense	03	03
Format Unit	04	04
Reassign Blocks	07	07
Read	08	08
Write	10	0A
Seek	11	0B
Inquire	18	12
Mode Select	21	15
Reserve	22	16
Release	23	17
Mode Sense	26	1A
Start/Stop Unit	27	1B
Send Diagnostic	29	1D
Read Capacity	37	25
Extended Read	40	28
Extended Write	42	2A
Extended Seek	43	2B
Verify	27	2F
Read Defect Data	55	37
Write Buffer	59	3B
Read Buffer	60	3C

Table 6-3. DSC Status Register Common Values

Value (hex)	Interpretation
00	SCSI Bus Error
40	All is normal
60	FIFO busy during diagnostic DMA transfer
C0	SBIC Interrupt

Table 6-4. DSC Control Register Common Values

Value (hex)	Interpretation
00,02	Interrupts Disabled, DMA disabled
10,12	Interrupts Disabled, DMA disabled
04,06	Interrupts Enabled, DMA disabled
14,16	Interrupts Enabled, DMA disabled
0C,0E	Interrupts Enabled, DMA Write to Host
08,0A	Interrupts Disabled, DMA Write to Host
1C,1E	Interrupts Enabled, DMA Read from Host
18,1A	Interrupts Disabled, DMA Read from Host

Table 6-5. SCSI Sense Byte Values

Value (hex)	Interpretation
0	No Sense Data Available
1	Recovered Error; see extended sense value
2	Unit Not Ready
3	Media Error; unrecoverable
4	Hardware Error; unrecoverable
5	Illegal Request; usually a program error
6	Unit Attention; the device has been reset by hardware
7	Data Protection Violation; an attempt to write protected medium
8	WORM drive failed to execute command
9	Vendor unique value
A	Copy Aborted; copy, verify or compare command failed
B	Command Aborted; command did not complete
C	Compare Equal; data compare successful
D	Volume Overflow
E	Miscompare; data did not compare correctly
F	Reserved

Table 6-6. SCSI Extended Sense Byte Values

Value (hex)	Interpretation
00	No sense data available
01	No index/sector mark signal
02	No seek complete
03	Write fault
04	Drive not ready
05	N/A
06	Rezero command failure
10	ID Checksum error
11	Uncorrectable read error
12	ID Sync mark not found
13	Data Sync mark not found
14	No record found
15	Seek error
17	Data recovered, using read retries (ECC not used)
18	Data recovered, using ECC (not using disk retries)
19	The Growth Defect List cannot be read
1C	The Primary Defect List cannot be read
20	Program Error, invalid opcode defined in CDB
21	Program Error, illegal block address specified in CDB
24	Program Error, invalid field in the CDB
25	Program Error, invalid logical unit number specified
26	Program Error, invalid field in a parameter list
27	Unit is write protected
28	Media change, the disk has received a Stop and Start command
29	Reset attention, following a unit power-up or reset
2A	Mode select parameters changed
31	The disk format is corrupted, or the disk is unformatted
32	No defect spare locations available
40	RAM buffer failure
41	Serial data path error
42	Power on diagnostics failed
43	Message reject retry failure
44	SCSI Hardware/Firmware error
45	SCSI Select or Reselect failed
46	Unsuccessful soft reset
47	SCSI bus parity error

Table 6-7. SCSI Status Codes

Code (hex)	Interpretation
00	Good Status
02	Check Condition - Issue Sense Command
04	Condition Met
08	Busy
10	Intermediate Status
14	Intermediate Condition Met
18	Reservation Conflict

Table 6-8. Device Driver Error Codes

Code (hex)	Interpretation
00	No error
FF	Invalid operation call to the driver. Programmer error.
01	Port number in long word 4 is invalid.
02	Timeout following DSC channel reset.
03	After a DSC channel reset, SCSI status did not indicate a reset status or 00.
04	SBIC auxilliary status is not 00 after the channel reset.
05	Timeout following a SCSI select indicated in long word 5.
06	Interrupt received after select command, but SCSI status is not 11 hex. Select was successful.
07	Timeout waiting for command phase request interrupt from target.
08	Command phase interrupt received from target, but SCSI status is not 8A hex.
09	Timeout waiting for CDB available status.
10	Block number in long word 1 is invalid for the command.
11	Block count in long word 2 is invalid for the command.
12	Timeout waiting for CDB available status while sending data.
13	Timeout waiting for command completion interrupt status, after receiving good interrupt status and DMA is initiated.
14	Command completion interrupt received, but SCSI status is not 1B.
15	Timeout waiting for CDB available status in aux. status register following a request for status phase from the target.
16	Timeout waiting for the message phase interrupt after status phase completed.
17	An interrupt was received from the target, but SCSI status is not 1F. The message phase interrupt did not occur.
18	Timeout waiting for CDB available following message phase interrupt.
19	Timeout waiting for interrupt after message phase completed successfully.
20	Interrupt received after successful message phase, but SCSI status is not 20 hex. The pause with ack interrupt status was not received.
21	Timeout waiting for interrupt after pause with ack status received. Target should have sent a disconnect interrupt.
22	An interrupt was received after the message phase and pause with ack interrupt, but SCSI status is not 85 hex.
23	Either the ending status byte from the status phase or the message byte from the message phase was not equal to 0. This indicates that all SCSI phases completed successfully, but the status bytes were not correct.
24	Timeout waiting for an interrupt after CDB is sent to the target.
25	An interrupt was received after the entire CDB was sent, but the SCSI status is not 1B, 18 or 19 hex.

Table 6-8. Device Driver Error Codes (Continued)

Code (hex)	Interpretation
101	Parity error detected in aux status while waiting for CDB available in status phase.
102	Parity error detected in aux status while waiting for CDB available in message phase.
103	Parity error detected from the SCSI status of 43 hex as the first interrupt after the read or write command was sent.
104	Parity error detected from SCSI status of 43 hex instead of status phase request interrupt.
105	Parity error detected from SCSI status of 43 hex instead of message phase request interrupt.
106	Perity error detected from SCSI status of 43 hex as the first interrupt after the inquire or sense command is issued.

SECTION 7
FUNCTION SELECT TESTS

7.1 OVERVIEW

This section describes the Function Select tests and how to use them. It also describes the commands available through the diagnostic Executive.

Function Select tests are special, low-level, command driven routines. There are no tests to "run," but only a collection of commands to execute. These commands can, however, be assembled into routines building a test loop. The command structure provides the user with the flexibility to force faults and verify the results. The responsibility to do so resides with the user.

By being low level tests, Function Selects provide assistance in diagnosing system problems to the chip level. A knowledge of system design and theory of operations is necessary to successfully diagnose a system at this level. Accordingly, Function Select tests are intended for use at a service depot, and not in the field.

Function Select tests operate under the control of the diagnostic Executive, like the SIT and Logic Tests. Unlike these other tests, a special mode of operation must be started, using the executive SERVICE command, to enable Function Select.

7.2 GENERAL RUN PROCEDURES

The Function Select tests are all run under the control of the diagnostic Executive. Depending on the system configuration, the executive is booted from, MCS, MTS or floppy diskette.

7.2.1 Load Diagnostic Executive

The Function Select tests run under the control of the diagnostic Executive. Refer to Section 4 for instructions on booting and using the Executive.

7.2.2 Start Printer Output

Diagnostics messages can be output to a parallel printer or a serial printer with an industry standard interface. The printer must be configured as the system printer. The messages are printed in addition to being displayed on the terminal screen.

To print messages to the serial printer, at the <exec> prompt, enter:

```
option sp
```

For the parallel printer, enter:

```
option lp
```

Make sure the printer is on-line. Now proceed with the tests.

7.3 USING FUNCTION SELECT TESTS

7.3.1 Enable Service Mode

On the MAI 2500/3000/4000 systems, you must enable Service Mode before you can load any Function Select Tests. To do this, enter the executive **SERVICE** command at the executive prompt:

```
service
```

You are then prompted for the service mode password:

```
Enter required password for Function Select enable:
```

The password is "b4bus". Type this password and press **ENTER**.

7.3.2 Select Function Select Tests

Function Select tests are loaded using the executive LOAD command followed by the test name. For example, to load the winchester function select test, enter:

```
load diskfs
```

at the executive prompt.

If the diagnostic Executive was booted from a device other than that containing the function select tests (e.g., from the diagnostics partition), you must first change the load device using the executive DEVICE command. For example, to select the MCS as the load device, enter:

```
device  cs0
```

With the DIA cartridge inserted in the drive, you may now load the Function Select tests.

7.3.3 Executing Commands

The Function Select tests are not programs that run like the other diagnostics. Rather, they are sets of commands. The commands are executed simply by entering the command name at the test prompt, followed by any required or optional parameters.

When a Function Select test is loaded, its prompt is displayed. At this prompt, you may enter any command in that test's command set, as well as any of the general diagnostic Executive commands.

7.3.4 Building and Executing Test Loops

The diagnostic Executive provides for one or two test loops, or command files, consisting of commands from the current Function Select test.

To begin building a test loop, enter at the Function Select test prompt:

build *n*

where *n* is either "1" or "2" indicating the test loop to build.

The prompt is changed to <loop*n*>, with *n* being either "1" or "2". At the prompt, you type a test command and press **ENTER**. Repeat for as many commands as you want executed during the test loop.

When you have entered the last command, use the executive **END** command to end building the loop.

The loop is then ready to run. To execute a test loop, use the executive **XLOOP** command. Enter:

xloop *n,m*

where *n* is "1" or "2", the number of the test loop to execute. The second parameter indicate the number of times to execute the loop. Without this parameter, the loop repeats endlessly, until the user interrupts execution by pressing **ESCAPE**.

The following example illustrates the process of executing commands, and then building and executing a short test loop, using the MCS Function Select test:

```
<mcsfs> init
<mcsfs> word 5000
<mcsfs> build
<Loop1> strm_wrt 4
<Loop1> strm_rd 4
<Loop1> end
<mcsfs> xloop 1,100
```

The first command initializes the test to defaults. The second command sets the word count to 5000 hex (50 blocks). A test loop is then built, consisting of two commands. Finally, the command loop (loop 1) is executed 100 times.

7.4 FUNCTION SELECT TEST DESCRIPTIONS

Each Function Select test consists of a set of commands. The following subsections describe the command sets belonging to, and special error messages displayed by, the tests. The Function Select tests described are:

- Winchester Disk Function Select Test (DISKFS)
- MCS Function Select Test (MCSFS)
- MTS Function Select Test (MTSFS)
- Four-way Function Select Test (FWFS)
- Eight-way Function Select Test (EWFS)
- LAN Function Select Test (LANFS)

7.4.1 Winchester Disk Function Select Test

The Winchester Disk Function Select Test provides low level command for testing the fixed disk controller and attached drive(s). Status checking before and after a command is executed provides diagnostic capability. The user is allowed enough flexibility to force faults, and to check or display status.

The Winchester Disk test can be used with either the one or two-board controller (the two-board controller is used on the MAI 2000 only).

7.4.1.1 TEST REQUIREMENTS

This test requires the following minimal hardware:

- CMB with 256 KB of memory
- One or two-board Winchester disk controller
- Winchester disk drive
- Floppy disk drive (MAI 2000 only), MCS or MTS drive

The only software requirements are the diagnostic executive and the **DISKFS** program.

7.4.1.2 LOADING THE TEST

The name of this test is **DISKFS**. To load the test, enter:

```
load diskfs
```

at the diagnostic executive prompt, or at the prompt for any Logic or Function Select test.

The test is on the diagnostic medium labeled DIA. If you have booted the diagnostic from a different medium type, it is necessary to change the load device using the **DEVICE** command (refer to Section 4).

After loading the program, it is ready to process commands.

7.4.1.3 TEST MODES

The Winchester Disk Function Select test operates in two test modes: Auto mode and Immediate mode. Auto mode is the default. Immediate mode must be invoked by the **IMMEDIATE** command.

In Auto mode, in addition to executing a command, several more steps are performed to facilitate major functions like writing to and reading from the disk. For instance, controller DMA registers are loaded, the necessary control command is sent to enable interrupts or start up the DMA data transfer.

Prior to executing the command, the controller status is checked to ensure that the command can be carried out. If the controller is ready, the parameters are then transferred, based on the current settings.

After a data transfer type command (write or read) is sent, the start DMA command is sent. Interrupts are enabled if so selected by the user. After sending the necessary control command, the program waits for the command to complete. The results of the operation are then checked. The controller status is read and interpreted, and the necessary **SENSE** commands are issued to determine if any controller detected errors occurred.

In Immediate mode, this automatic operation is side stepped, in case trouble shooting requires it. In Immediate mode, the user must enter the commands **SELECT**, **LDMA**, **CONTROL** and **STATUS** before entering a disk command. The necessary command and parameters are then sent, including the control command to enable interrupts and start DMA. Afterwards, the user must check the ending status by entering commands like **STATUS** and **RDMA** to get the ending status byte, or **WAIT** and **CES** to check the results of the operation.

If in Immediate mode with interrupts enabled, the interrupt service routine will display the prompt <Interrupt> when an interrupt occurs. No other processing is performed until the user specifies an action, usually to check the result bytes in the controller, or to check the status after the operation to determine the results.

7.4.1.4 WINCHESTER DISK FUNCTION SELECT COMMANDS

There are 8 types of commands: Standard, Buffer, Parameter, Initialization, Control, Immediate, Execute and High Level commands. The commands are described according to these types. The LIST command displays all of these commands, but does not divide them into types.

7.4.1.4.1 Standard Commands

Standard commands work on any controller board and require a good understanding of the controller under test.

ADDRESS {*n*}

This command selects or displays the controller address currently used by the test. If an argument is included, the memory mapped address used by the test is changed to that address. If no parameter is used, the current address is displayed. Specifying an address requires knowledge of the system memory map.

CDATA

This command compares the byte set up by SETEXP to the byte received by the EREAD command. If the comparison is not equal, and error is displayed.

CLOCK *n*

This command starts the clock. The argument specifies the interval in milliseconds, and must be in the range 2-1000 for the time to operate correctly.

EREAD {*n*}

This command reads a byte from the controller. If an argument is not included, the read is taken from the address specified by the **ADDRESS** command. If an argument is specified, the read is from this address plus the argument displacement. The byte read is stored in memory for comparison to the expected byte.

EWRITE {*n*}

This command sends the byte set by **SETEXP** to the address specified by the ADDRESS command. If an argument is not included, the byte is sent to the base address. If an argument in included, the byte is sent to the base address plus the argument displacement.

KEYS

This command downloads commands to the programmable function keys. The functions are described in Table 7-1.

Table 7-1. DISKFS Function Key Assignments

KEY	COMMAND	MEANING
01	spec	Issue the SPECIFY command
02	rcal	Recalibrate the heads
03	gracyl 1	Seek to a random cylinder
04	write	Write to the disk
05	read	Read the disk
06	compa	Compare the buffers
07	stat	Read and display adapter status
08	list	List the DISKFS commands
09	eit	Enable interrupts
10	par	Display active parameters
11	restart	Restart the diagnostic
12	init	Initialize the diagnostic
13	op stop	Enable option stop on error
14	x1 1 10	Execute the build loop 10 times
15	boot	Boot from floppy disk
16	lo dutil	Load the DUTIL disk utilities program
17	lo diskfs	Load the DISKFS program
18	autom	Enable Auto mode
19	imm	Enable Immediate mode
20	crea	Create the next data pattern
21	wdis	Display the write buffer, 128
22	rdis	Display the read buffer
23	dit	Disable interrupts
24	forcyl	For cylinders 0 to max
25	forhead	For heads 0 to max
26	cco	Clear the SASI controller
27	select	Select the SASI controller
28	idma	Display the last DMA address

REGISTER

This command displays the controller address map, the description of each register and the register types.

REVISION

This command displays the program title and revision.

SETEXP *n*

This command sets the expected byte parameter. This byte is sent to the controller by the EWRITE command, and is used for comparisons by the **CDATA** command.

TIME

This command displays the elapsed time since the clock was started. Depending on interrupt level control, the time may not increment.

7.4.1.4.2 Buffer Commands

BUFFREAD {*n*}

This command selects and displays the starting address of the read buffer. If an argument is specified, the starting address is set to this value. If an argument is not specified, the current starting address is displayed. The address must be greater than the start address of the write buffer. The address specified may not cross the program to prevent a read of disk data from writing over the program.

BUFFSIZE {*n*}

This command selects and displays the size of the write and read buffers. The argument is added to the write buffer start address to calculate the start of the read buffer. The command overrides any previous **BUFFREAD** command. The argument is automatically rounded to an even address, as required by the controller.

BUFFWRIT {*n*}

This command selects and displays the starting address of the write buffer. If an argument is specified, the starting address is set to this value. If an argument is not specified, the current starting address is displayed. The address can be set to non-existent memory to test the non-existent memory indication. Care should be used with this command, since the address may cross the program area, causing subsequent **CREATE** commands to overwrite the program itself.

COMPARE {*n*}

This command compares the read and write buffers. If no argument is included, the amount of data specified by the previous **#SECTORS** command is compared. If an argument is included. this specifies the amount of data to compare. If the value of the argument is greater than the amount of data specified in the previous #SECTORS command, the lesser value is used. The maximum amount of data to compare is controlled by the **BUFFSIZE** command.

CREATE {*n*}

This command creates a block of data to be used for the next write or compare operation. The parameter specifies the data type to create, as shown in Table 7-2. If no parameter is specified, the pattern increments to the next type, wrapping around from the last type to the first. This allows the use of all data types in a loop without specifying the type for each loop.

Table 7-2. DISKFS Data Types

TYPE	DESCRIPTION	HEX DATA
0	All zeros	0000 0000 0000 0000 0000
1	All ones	FFFF FFFF FFFF FFFF FFFF
2	Floating 0	FFFE FFFD FFFB FFF7 FFEF
3	Floating 1	0001 0002 0004 0008 0010
4	Random	F946 EA89 12DD 32A0 00FF
5	All fives	5555 5555 5555 5555 5555
6	Di-bit	CCCC CCCC CCCC CCCC CCCC
7	Tri-bit	E383 38E3 8E38 E38E 38E3
8	Quad-bit	F0F0 F0F0 F0F0 F0F0 F0F0
9	Zero di-bit	6DB6 DB6D B6DB 6DB6 DB6D
A	Incrementing	0001 0002 0003 0004 0005
B	Zero tri-bit	7777 7777 7777 7777 7777
C	Zero quad-bit	7BDE F7BD EF7B DEF7 BDEF
D	One di-zero	9249 2492 4924 9249 2492
E	One tri-zero	8888 8888 8888 8888 8888
F	One quad-zero	8241 0824 1082 4108 2410
10	Increase freq.	FF00 FE03 F03E 0F0E 32AA
11	Decrease freq.	24AA 6663 8E38 7878 783F
12	Jitter	9659 6596 5965 9659 6596
13	Worst Case 1	D936 4DB6 4ED9 364D B64E
14	Worst Case 2	6DB6 DB0F 6DB6 DB0F 6DB6
15	Worst Case 3	DADA CA58 C2FE DADA CA58

INCBUFF {*n*}

This command increments the start addresses of the write and read buffers. If an argument is specified, the addresses are incremented by this number of bytes (rounded to an even number). If the argument is not specified, the buffer address are incremented by the number of bytes specified by the previous **BUFFSIZE** command.

RDISPLAY {*n*}

This command displays the contents of the read buffer. The argument specifies the number of words to display. If no argument is given, one page (256 bytes) of data is displayed. If more than one page is displayed, paging forward is controlled by the **ENTER** key. Press "/" to cause the previous page to be displayed.

SWAPBUFF

This command swaps the pointers for the write and read buffers. Care should be taken in using this command so that the **CREATE** command does not wipe out data read into the read buffer. This command is particularly useful in reading disk contents and writing the same data back, or to another drive.

WDISPLAY {*n*}

This command displays the contents of the write buffer. The argument specifies the number of words to display. If no argument is given, one page (256 bytes) of data is displayed. If more than one page is displayed, paging forward is controlled by the **ENTER** key. Press "/" to cause the previous page to be displayed.

ZAPREADB {*x*}

This command sets (zaps) the read buffer to a known pattern. If a hex argument is included, this pattern is used. If an argument is not included, the pattern is BAD1.

ZAPWRITE {*x*}

This command sets (zaps) the write buffer to a known pattern. If a hex argument is included, this pattern is used. If an argument is not included, the pattern is BAD1.

7.4.1.4.3 Parameter Commands

Parameter commands cause no actions to occur, but set general parameters governing the execution of other commands. Parameter commands which use an argument *n* can be used without the argument to display the current value of the parameter.

BLOCK *d*

This command selects the absolute decimal logical block number for the next command. This can be used in place of the **CYLINDER**, **HEAD** and **SECTOR** commands.

CYLINDER {*n*}

This command selects and displays the cylinder to be used for the next Execute command. The user must enter the **SEEK** command to position the heads where the operation is to be done. To test implied seeks, the **SEEK** command is not necessary.

+CYLINDER

This command increments the logical cylinder parameter. If the last cylinder is reached, the cylinder remains at the maximum cylinder, and the message "Last cylinder reached!" is displayed.

-CYLINDER

This command decrements the logical cylinder parameter. If the first cylinder is reached, the cylinder remains at cylinder 0.

DADDRESS {*n*}

This command selects and displays the drive address for subsequent commands. The argument value may be 0 or 1.

DISK

This command indicates that the next command entered will affect a range of cylinders and heads. The range of tracks is from head 1 on cylinder 0 up to the maximums set **MHEAD**. This command will change the settings of the maximum cylinder that might have been previously selected by the **MCYL** command.

DPATTERN {*n*}

This command sets up and displays the data pattern byte to be used by the **FORMAT** command. This pattern is actually written on the disk in the data field during the format operation. The default pattern is E5 hex.

FORHEAD *b,e*

This command is used to specify a range of heads to be used for a multiple head operation. It sets up a "FOR LOOP" from head *b* to head *e*. When no arguments are included, the command assumes a range of heads from 0 to the maximum. The next Execute command will be looped on the specified heads. The first parameter (*b*) specifies the beginning head and the second parameter (*e*) specifies the ending head. The ending head must be less than the maximum head, and greater than the beginning head.

FORCYL *b,e*

This command is used to specify a range of cylinders to be used for a multiple cylinder operation. It sets up a "FOR LOOP" from cylinder *b* to cylinder *e*. When no arguments are included, the command assumes a range of cylinders from 0 to the maximum. The next Execute command will be looped on the specified cylinders. The first parameter (*b*) specifies the beginning cylinder and the second parameter (*e*) specifies the ending cylinder. The ending cylinder must be less than or equal to the maximum cylinder, and greater than the beginning cylinder.

GRCYL *x*

This command generates a random logical cylinder to be used for the next seek operation. If an argument is added to the command, then random seeks will be executed for the number of times specified. The cylinders selected are between 0 and the cylinder specified by the **MCYL** command.

GRHEAD

This command generates a random head to be used for the next operation. The range of values is between 0 and the maximum head.

GRSECTOR

This command generates a random sector for the next operation. The range of values is between 0 and the maximum sector.

HEAD {*n*}

This command selects and displays the head to use for the next Execute command. This is the logical head desired to start a transfer, and remains in effect until another **HEAD** command or the **FORHEAD** command is encountered. The **HEAD** command cancels any **FORHEAD** loops by zeroing the headcount.

+HEAD

This command increments the logical head parameter for the next operation. If the head value is at the maximum, the value wraps around to 0.

-HEAD

This command decrements the logical head parameter for the next operation. If the head value is at 0, the value remains at 0.

INQUIRE

This command sizes and displays the system configuration, and runs the controller quick test. If "ERROR" is displayed, the controller should be replaced.

INTERLV {*n*}

This command selects and displays the sector interleave factor used in formatting. The interleave factor entered implies the increment amount between each logical sector address. An interleave factor of 0 for the controller will cause the controller to default to a factor of 2. An interleave factor of 1 is the default, and interleave factor of 2 or more causes the controller to provide an extra sector per track. The interleave factor takes effect at format time, by use of the **SPECIFY** and **DFORMAT** commands.

LUNIT {*n*}

This command selects and displays the logical unit to test. The argument may be any value in the range 0-7. The command automatically selects the correct controller and unit, based on the logical unit. The command form **LUNIT** *n*,*n*+1 is allowed to select both units on one controller. This command can be used in place of the **WDC** and **UNIT** commands.

PARAMETER

This command displays the currently active parameters.

PRECOMP {*n*}

This command selects and displays the write precomp cylinder to be invoked by the controller. The precomp cylinder takes effect only at format time, by the **SPECIFY** and **DFORMAT** commands.

SECTOR {*n*}

This command selection and displays the logical sector number to start the next transfer command.

#SECTORS {*n*}

This command selects and displays the number of sectors to transfer.

+SECTOR

This command increments the logical sector parameter for the next operation. If the sector value is at the maximum, the value wraps around to 0.

-SECTOR

This command decrements the logical sector parameter for the next operation. If the sector value is at 0, the value remains at 0.

UNIT {*n*}

This is the same as the **DADDRESS** command.

WDC {*n*}

This command selects and displays the controller to test. The parameter may be any value in the range 0-3. The default when the diagnostic is loaded is to test all controllers found. WDC selects a single controller.

7.4.1.4.4 Initialization Commands

Initialization commands use a single parameter, *n*. If the parameter is not given, the current parameter value is displayed.

IFMAXCYL {*n*}

This command sets the next cylinder after the maximum has been reached. If the last seek reached the maximum cylinder indicated by the **MCYL** command, then select cylinder n for the next operation. This allows a loop to continue without the cylinder being stuck at the maximum cylinder.

MAXTOR {*n*}

This command causes the program to select Maxtor drive defaults. If the argument is omitted, the program assumes the 140 MB defaults.

If *n* = 105, select 105 MB defaults.
If *n* = 140, select 140 MB defaults.
If *n* = 190, select 190 MB defaults.

MCYL {*n*}

This command selects and displays the maximum cylinder value to use for ending and controlling multiple cylinder operations. The default is forced to be the same as the physical number of cylinders minus 2 or 3. This is done to protect the defect map on the disk.

MHEAD {*n*}

This command selects and displays the maximum head value to use for ending and controlling multiple head operations. The default is forced to be the same as the physical number of heads minus 1.

MICROPOL {*n*}

This command causes the program to select Micropolis drive defaults. If the argument is omitted, the program assumes the 50 MB defaults (6 heads, 32E hex cylinders and reduced write current cylinder of 190 hex).

If *n*=50, select 50 MB defaults.
If *n*=85, select 85 MB defaults.

MSECTOR {*n*}

This command selects and displays the maximum sector value to use for ending and controlling multiple sector operations. The default is forced to be the same as the physical number of sectors minus 1.

RODIME {*n*}

This command causes the program to select Rodime drive defaults. If the argument is omitted, the program assumes the 20 MB defaults.

If *n*=20, select 20 MB defaults.
If *n*=40, select 40 MB defaults.
If *n*=53, select 53 MB defaults.

SIZE {*n*}

This command selects and displays the sector byte count used for formatting the disk. The default is 200 hex. The legal range is 0-FFFF hex.

7.4.1.4.5 Control Commands

Control commands cause no operation to be performed, except for the EIT command. These commands control the operation of the diagnostic and the controller operation. They are also used to check the results of the last operation.

AUTOMODE {*n*}

This command starts Auto mode. Refer to Section 7.4.1.3. The argument *n* can be the hex digits "E" or "F". "E" causes only the first compare error to be displayed, and "F" stops displaying compare errors.

BMODE

This command selects buffered mode for seek commands, to allow overlapped and faster seeks. This parameter is used by the **SPECIFY** command to allow buffered seeks.

DELAY {*n*}

This command causes a delay of the specified number of milliseconds. The delay must be an EVEN hexadecimal number.

DIT

This command disables interrupts, starting with the next command.

DMA

This command forces DMA type data transfers, causing the correct DMA write or read buffer address to be used.

EIT

This command enables interrupts, starting with the next command. The command loads the vector register with the last vector selected by the **VECTOR** command, or the default value of 40 hex. The vector service address is also loaded into the vector table in low memory to point to the interrupt service routine.

HELP

This command displays a diagnostic help message and sets help mode. With help mode set, some commands can be executed again simply by pressing **ENTER** at the AGAIN? prompt.

IMMEDIATE

This command disables Automatic mode. Refer to Section 7.4.1.3.

LOOP {*n*}

This command causes the next command to loop for the specified number of times (specified in hex). If no argument is included, the loop count is displayed. The range of loops is 0-7FFF hex. Looping can be interrupted by **CTRL + C**. Only certain commands can be looped: **WRDATA**, **WRCONTROL**, **RVECTOR**, **READ**, **WRITE**, **LREAD**, **LWRITE**, **VERIFY** and **WRCOMPARE**.

NDMA

This command forces non-DMA type data transfers. Data transfers are by simple input-output from or to the read or write buffer areas.

NMODE

This command selects normal mode for seek commands. This causes the slowest seeks available to the SASI controller.

NOHELP

This command resets HELP mode, eliminating the repeat function.

READONLY

This command prevents most writes to the disk by diagnostic commands. Some low level commands, like DCB and WDATA, may still write to the disk.

RETRY *d*

This command sets the retry count for the **SURFACE** and **WRCOMPARE** commands. The count must be a decimal number in the range 0-255.

SVERIFY

This command forces all seek commands to be followed by verifying the position of the heads. This feature is disabled by the **DEFAULTS** command.

WAIT

This command causes the diagnostic to wait for either an interrupt to occur (if EIT), or for the controller to post command completion status (if DIT).

WRITEOK

This command disables write protection. This is the default condition for this diagnostic.

WVERIFY

This command forces all writes to be followed by a **VERIFY** command, to add a degree of confidence to the data written to the disk. This feature is disabled by the **DEFAULTS** command.

7.4.1.4.6 Immediate Commands

Immediate commands cause an immediate operation on the controller or the last drive selected by the UNIT command. Many of these commands are low level commands, so the operator must understand the overall operation of the controller.

BERRC

This command resets (clears) the bus error status bit 0 in the status byte.

CCONTROL

This command clears the entire controller board by a SASI reset.

CES

This command checks for errors by checking the ending status of the last command. This is done automatically in Auto mode. The command should be issued only in Immediate mode, and normally follows the **SELECT** and **WAIT** commands.

CONTROL {*n*}

This command writes the hex value specified to the control register by setting the appropriate bits to enable interrupts, control the LED, cause a SASI reset and cause DMA write or read startup. If the interrupt enable bit is OFF, the command disables interrupts as in the DIT command.

DCB {*n*}

This command defines the control block. The argument specifies the actual Device Command Block (DCB) number which was previously entered. If no argument is specified, all current DCBs contained in memory are displayed. These DCBs are primarily intended to supplement the higher level command set, like **SEEK**, **RCAL**, **WRITE** and **READ**. Any DCB which the SASI controller understands can be executed by the DCB command. This command implies Immediate mode, even if in Auto mode, so **SELECT** must be executed first. DCB 0 always refers to the last executed DCB.

DCB *n* 1 2 3 4 5 6 7 8 9 10

This is another form of the DCB command which allows the user to first define and execute a DCB. Once the DCB is entered, it can be executed again by just entering the **DCB** n command, where *n* is the DCB number. Once the **DCB** is executed, the results are available to the user to be read by the **RDA** command or checked by the **CES** command after the **WAIT** command.

Up to 25 DCBs can be defined, with n in the range 1-19 hex. A DCB consists of 6 or 10 bytes, as indicated in Table 7-3 and 7-4. Refer to the documentation on the controller for the individual bytes which make up a DCB.

Table 7-3. 6 Byte DCB

ARGUMENT	USE OR MEANING
n	The DCB number
1	Opcode (Table 7-5)
2	MSB, logical address, bits 7-5 define unit number
3	Logical block address
4	Logical block address, LSB
5	Number of blocks to transfer
6	Always 0

Table 7-4. 10 Byte DCB

ARGUMENT	USE OR MEANING
n	The DCB number
1	Opcode (Table 7-5)
2	Opcode extension (see controller documentation)
3	Logical block address, MSB
4	Logical block address, mid high
5	Logical block address, mid low
6	Logical block address, LSB
7	Always 0
8	Number of blocks to transfer, MSB
9	Number of blocks to transfer. LSB
10	Always 0

Table 7-5. Controller Op-Codes

CODE	MEANING
00	Test Unit Ready
01	Rezero Unit (recalibrate heads)
03	Request Sense Data
04	Format Unit
08	Read
0A	Write
0B	Seek
0F	Translate
13	Write Sector Buffer
14	Read Sector Buffer
15	Mode Select
1A	Mode Sense
1B	Start/Stop Unit
1C	Receive Diagnostic
1D	Send Diagnostic
25	Read Capacity
28	Long Read
2A	Long Write
2E	Write and Verify
2F	Verify
31	Search Data Equal

ECC

This command forces the controller to do ECC checking of data, and enables controller retries. This is the default for this diagnostic.

LDMA {*n*}

This command sends the three byte address for DMA operations to the DMA registers. If no argument is specified, the last DMA address selected is loaded.

NOECC

This command disables error correction of data and retries by the controller. This command should be entered prior to doing a disk surface analysis.

OVERLAP {*n*}

This command enables (*n* = 1) or disables (*n* = 0) the program function of waiting for two interrupts from a **SEEK** command. The dual interrupt generation is done on the single board controller only. The **EIT** command must be used to enable interrupt generation and checking.

RCAL {*u*}

This command recalibrates drive *u*. If an argument is present, it specifies the unit which then remains selected for subsequent operations. If no unit is specified, the last unit selected is recalibrated.

RCONTROL

This command reads the control register and displays the contents.

RDATA

This command reads a byte from the input register and displays the contents.

RESET

This command issues a system reset.

RVECTOR

This command reads and displays the contents of the vector register.

SELECT

This command selects the SASI controller. After selection, the program waits until the controller is ready for the DCB bytes, indicated by the controller returning a C2 hex status. The select command must precede and controller command while in Immediate mode.

STATUS {*n*}

This command, without an argument, reads the adapter status and displays the result. With an argument, the command reads the status and waits for the status to match the argument. The command will wait indefinitely, but may be interrupted by **CTRL + C**.

VECTOR {*n*}

This command loads the vector register with the hex byte *n*. If the parameter is not specified, the register is loaded with the last entry. The interrupt vector address is also initialized.

WDATA {*n*}

This command writes the byte argument to the output register.

WRDATA {*n*}

This command writes a byte to the output register, reads a byte back from the input register and compares the two bytes. If a byte argument is included, this is the byte written. If no argument is specified, a random byte is generated and set to the output register.

WRCONTROL {*n*}

This command writes, reads and compares the data written to the control register. If an argument is specified, only the least significant 4 bits are written and compared. If no argument is specified, random data is written.

WRVECTOR {*n*}

This command writes, reads and compares the data written to the vector register. If an argument is specified, the hex data byte is written and compared. If no argument is specified, a random data byte is used.

7.4.1.4.7 Execute Commands

BURNIN {*n*}

This command does a disk burn-in. If *n* = 1, a controller burn-in is done. If *n* = 2, both a register and controller burn-in is done. If *n* = 3, a register, controller and disk burn-in is done.

BUSYTRY {*n*}

This command specifies and displays the number or retires an operation will perform when the controller posts the busy status in the ending status byte. The range of retries specified is 0-7FFF hex. A retry count of 0 will produce errors when the retry count is exhausted.

CLRLOG

This command clears the error log.

LOGOUT

This command displays the current list of error detected by the controller.

MODESENS

This command issues the Mode Sense command and displays the data received.

RRAM

This command reads the controller sector buffer. The data read is placed in the read buffer for comparison or display. The number of sectors is set to 2.

SEEK {*n*}

This command causes the drive to seek to the specified cylinder. If no cylinder is specified, the seek is to the cylinder specified by the last **SEEK**, **CYLINDER** or **GRCYL** command.

SENSE {*u*}

This command issues the Mode Sense command. If there is an error indicated by the check bit in the sense byte, it is displayed. If the valid bit is set, the cylinder, head and sector where the error occurred is also displayed. If the unit argument is specified, the sense command is issued to that unit and the unit remains in effect. If the check bit is not set, an error message is displayed with error code 0 (no error).

SPECIFY

This command issues the Mode Select command to specify the attached drive characteristics. The diagnostic assures drive definition default values for all the parameters if the user does not specify them (using the **MAXTOR**, **MICROP** or **RODIME** commands).

The next command must be the **DFORMAT** command. Any other command will cause the controller to detect an error, error code 1C, indicating a bad format.

STARTUNIT {*u*}

This command issues the Start Unit command, initializing the drive.

STOPUNIT {*u*}

This command issues the Stop Unit command to position the heads to the landing zone for shipping. The argument *u* specifies unit 0 or 1.

TSTDMA {*n*}

This command executes a DMA test sequence. The command argument specifies the data type to be created for the test. Data is accessed from memory and transferred to the input register, one byte at a time, for comparison to data contained in memory.

TSTINT

This command verifies that the controller generates an interrupt from a forced bus error condition. Two errors can be detected by this command; that the bus error is not generated, or that the controller does not generate an interrupt.

USTATUS {*u*}

This command issues the Test Unit Ready command and checks for errors. The argument specifies the test unit, and remains in effect for subsequent commands. If no argument is given, the last unit selected is used.

WRAM

This command writes the controller sector buffer. The data is specified in the write buffer. The number of sectors is set to 2.

7.4.1.4.8 Multiple Parameter Execute Commands

The following Execute commands require that parameters have been set using the Parameter commands. Which parameters must be set depends on the command.

The unit argument *u* specifies the test unit. If the unit is not specified, the last unit selected is used.

LREAD {*u*}

This command performs a long read, allowing sector transfers in excess of 512 sectors.

LWRITE {*u*}

This command performs a long write, allowing sector transfers in excess of 512 sectors.

READ {*u*}

This command reads data from the disk into the read buffer.

TRANSLAT {*u*}

This command issues the Translate command and displays how the controller translates the logical sector. The value returned specifies the physical location of the logical sector.

VERIFY {*u*}

This command issues the Verify Data command, to verify the ECC of the last data written to the disk.

WRITE {*u*}

This command writes data to the disk from the write buffer.

7.4.1.4.9 High Level Commands

CONVERT

This command translates every entry in the error table and places the results in the write buffer. The results are also displayed. These values can be used to reformat the disk with defects, using the DEFORMAT command.

DEFAULTS

This command initializes the controller by issuing the Clear Controller command, and then loading the vector register and the DMA register. It then restores all the default parameters (Rodime 20) and selects unit 0 and Auto mode.

DEFORMAT {*u*}

This command performs a disk format using a defect track list. This requires that the operator enter the specific defect list into the write buffer prior to doing the **DEFORMAT** command. It is generally better to use the **DUTIL** disk utility for this purpose.

DFORMAT {*u*}

This command formats the entire disk. The **MCYL** parameter determines the last cylinder to format. Formatting begins at cylinder 0. Formatting is aborted if a recalibrate, seek or other format error is detected. The argument specifies the unit to format, and remains active for subsequent commands.

DREAD

This command reads the entire disk. Each track is read and the data transferred to the read buffer. All sectors are forced to be read on each track. All errors are put into a table and displayed on the screen. The **TABLE** command can be used to display the error table.

DVERIFY

This command verifies the entire disk. The disk is verified one track at a time. All errors are stored in a table and displayed. Use the **TABLE** command to display the error table.

INIT

This command initializes the diagnostic.

MEMORY {*n*}

This command tests memory with the WDC controller buffer. The controller 2 sector RAM buffer is used. The argument defines the 256 K memory address space to test, based on the memory board number. Board 0 is not tested. The arguments test memory as follows:

Argument	Start Address (hex)
1	40000
2	80000
3	100000
4	140000
5	180000
6	1C0000

RCOMPARE

This command reads data from the disk into the read buffer and compares the data with what is in the write buffer.

RESTART

This command initializes the diagnostic by issuing the recalibrate command and resetting necessary flags.

SURFACE {*u*}

This command performs a surface analysis of the disk unit specified. The procedure overwrites all data previously on the disk. The disk is checked from cylinder 0 to **MCYL**.

Failing tracks are stored in an error track table. Upon completion, the failing tracks are displayed in a table format. If more than 128 bad tracks are located, the surface analysis is aborted and the errors are displayed.

The retry count set by the **RETRY** command is used by this command. If an error is detected and the retry count is not exhausted, the operation is retried. Only when the retry count is exhausted is the failing track put into the error table.

WRCOMPAR

This command writes data from the write buffer to the disk, reads the data back into the read buffer and compares the buffers. The **RETRY** count is used in case of miscompares. The hex pattern BAD1 is written to the read buffer prior to performing the read operation.

7.4.1.5 ERROR REPORTING

7.4.1.5.1 Bad Status Before Command Issued

- Adapter not ready prior to issuing command!

  Either the controller was busy prior to issuing the command, or the command caused the controller to stay busy after the command was issued. The operation flag displayed with this error is the DCB byte opcode.

- Unit write protected.

  The write protect flag in memory was set when a write operation was attempted.

- Bad status after SASI selected.

  The status should indicate the selected and busy condition and output register free. After the SELECT controller command was issued, the program checked for expected status of C2 hex, and a software timeout occurred.

- DCB not defined.

  The DCB requested has not been defined by the previous entry of the DCB command. DCB 1 through 25 are allowed.

- DCB byte transfer timeout.

  The bytes which make up the DCB are sent to the controller in simple output mode. Correct status is checked prior to sending each byte. A timeout occurred waiting for correct status of C2 hex to transfer the next DCB byte.

- Data transfer timeout.

  If the operation which transfers disk data in the non-DMA mode or the command itself requires sending data associated with the command in simple I/), status must be correct before the data is transferred. If the correct status is not obtained in the allotted time, this error will occur.

- Bad status found before command completion.

  After the command is issued, the program will wait for an interrupt and check status, or will wait for correct ending status from the controller. Correct ending status of CC hex was not found.

7.4.1.5.2 Bad Status After Command Issued

- Expected bus error status not found

  After forcing a bus error, the status bit was not set.

- Unexpected Adapter detected address bus error

  While doing a DMA transfer, the controller detected a bus error.

- Adapter detected bus error didn't generate interrupt

  The Adapter has the bus error bit set in status bit it did not generate an interrupt as expected by the program.

- Can't obtain the ending status

  Because the command completion status was not obtained, the ending status byte was not read from the SASI controller for ending status error checking.

- Message phase timeout

  After the ending status byte is read from the controller, it should indicate message phase with a status of E8 hex.

- Message byte not zero

  The actual message byte read from the controller while in message phase should be 0.

- Error processing: SENSE command failure

  The previous operation had the check bit set in the ending status byte indicating that a SENSE command was necessary. The SENSE command is automatically issued in this state. This error is displayed when the SENSE command also failed to complete correctly.

- Position error

  After the user selected seek verification by the SVERIFY command, the translate command issued by the program indicated that there was a difference between the physical position requested and the actual position that the SASI controller indicated.

- With interrupts enabled, no interrupt occurred!

  After the command is issued, the controller did not generate an interrupt as expected in the time allotted.

- With interrupts disabled, an interrupt occurred

  The program detected an interrupt when the controller was not expected to generate one.

- Non-zero status after reset

  The controller status should be equal to 0 when the clear controller command is issued.

- Ending status non-zero

  The command completion status byte read from the controller after the command completes is not 0. The ending status obtained from the controller is displayed along with the command code. The command code is the DCB opcode byte.

- Command aborted before completion

  The program found operation status bytes in error while trying to check the results of the operation. This normally occurs when commands are entered out of sequence, or if the WAIT command is not entered while in IMMEDIATE mode after a SASI controller command is issued. it also occurs if the operator hits ESCAPE during command processing.

- Unexpected sequence or program error

  Normally caused when DISKFS is not exercising the expected control over the result of the operation. The program should be reloaded in this case, since it is not operating as expected.

- Bad unit x status

  Bad unit status existed after the command. The unit, error code from the SENSE operation, logical sector address, and the cylinder, head and sector where the error occurred is displayed, as applicable.

- Cylinder exceeds maximum

  The cylinder requested is too large, or larger than the cylinder specified in the MCYL command arguments. This prevents the user from trying to seek to an invalid cylinder. The cylinder requested and the program maximum are displayed.

- Buffers don't compare

  The write buffer data is not the same as the read buffer data. The expected words and the received words are displayed, as well as the buffer addresses. This is normally caused by the write buffer being changed after a read operation and then doing the COMPARE command, or that the user did not seek to the cylinder that the CYLINDER parameter specifies.

  If the data in the read buffer is BAD1 hex, then the read operation probably did not occur. The read buffer is configured to the BAD1 pattern by the WRCOMPARE command prior to the read operation.

- Adapter address bus error

  The program received a bus error from a legal controller address.

- Controller detected error

  At the completion of a controller command, the ending status byte indicated an error. The results of the SENSE command are displayed with this error in the format: error code, command causing the error, the unit in error.

- Bus error status won't reset

  The command to reset the bus error bit in status did not cause the status to clear.

- Reset line status not set

  After issuing the SASI reset command, the status did not indicate SASI reset active.

- Compare Error

  The low level command CDATA found that the expected and received bytes didn't compare.

- DMA Test data compare error

  The DMA test, which transfers data from memory in a test environment without involving the disk, found that the data in memory did not compare with the data accessed via the DMA test cycle. This normally means that the DMA register is not correctly accessing the memory area expected.

- Vector reg data loop-thru error

  Data written to the vector register does not compare to the data read back. Expected and received data are displayed.

- Control reg data loop-thru error

  Data written to the control register (least significant four bits) does not compare to the data read back. Expected and received data are displayed.

- Data registers loop-thru error

  Data written through the output data register in test mode does not compare to the data read from the input register. Expected and received data are displayed.

- Device busy timeout after multiple retries

  If the controller indicates that it is busy by the ending status, the operation is retried automatically by the diagnostic. This error is displayed when the controller continually indicates busy status.

- WARNING: BAD SUPERBLOCK!

  The superblock does not contain valid data in order to do an auto-size.

7.4.1.5.3 Interpretive Messages

The following messages may accompany error messages to give additional error information.

- SASI reset active

  During error processing, the reset line status bit in status was found ON.

- Lost BUSY status

  During the processing of the command after the SASI controller selected, the busy status bit was found OFF.

- Command complete timeout

  The status of CC hex was not read by the program in the time allotted.

7.4.1.5.4 Warning Messages

The following messages may be displayed by DISKFS, indicating a special condition of which the user should be aware. These messages do not necessarily indicate that an error has occurred.

- Last cylinder reached!

  When the last cylinder has been accessed in a FOR loop or a test loop, this message is displayed to inform the user that subsequent cylinder increment operations will not work and that the cylinder will remain at the maximum cylinder. Use the IFMAXCYL command to select a new cylinder when the maximum cylinder is reached.

- HELP MODE

  If the user enters the HELP? command for the first time, a user help message is displayed. This message is only displayed once, and subsequent entries of the command will not display the message.

- No results to test

  If the CES command finds invalid flags to indicate that status desired to check does not fit the correct format, this warning is displayed. It is usually caused by the user entering the CES command out of context with an operation.

- Interrupt service routine hung

  When the RTE instruction is executed in the interrupt service routine, control should be returned to the diagnostic test flow. If the interrupt service routine is called 128 times before the diagnostic can reset the counter, this error is displayed.

- Invalid emulation instruction

  The diagnostic uses the emulation vector for certain operations. This error is displayed before a trap 15 when the emulation code is invalid. This normally means the program was corrupted in some way or there is a program error.

- PROGRAM CHECKSUM ERROR!!

  When the program is loaded, a checksum of the DISKFS program area. If the user reinitializes the diagnostic, then a checksum is generated again and compared with the original checksum generated at load time. The above message is displayed when the checksums do not match or when the user has altered the program section if some way. This provides the user with some confidence in the integrity of the program when unexpected errors occur with the program. The DEFAULTS command will recalculate the checksum. The checksum is also checked by the INIT and RESTART commands.

  When this error occurs for no expected reason, the program should be reloaded.

7.4.2 MCS Function Select Test

The MCS Function Select Test provides low level commands for testing the 1/4 inch Cartridge Streamer.

The test provides a full range of commands, from simple register reads and writes to more complex IOPB-driven functions. The test enables the operator to observe and diagnose the functionality of the tape controller and its drive/formatter.

7.4.2.1 TEST REQUIREMENTS

This test requires the following minimal hardware:

- CMB with 128 KB of memory
- MCS tape drive/formatter and controller
- Floppy disk drive (MAI 2000 only)or MTS drive for loading the diagnostic.

The only software requirements are the diagnostic executive and the MCSFS program.

7.4.2.2 LOADING THE TEST

The name of this test is MCSFS. To load the test, enter:

load mcsfs

at the diagnostic executive prompt, or at the prompt for any Logic or Function Select test.

The test is on the diagnostic medium labeled DIA. If you have booted the diagnostic from a different medium type, it is necessary to change the load device using the DEVICE command (refer to Section 4).

After loading the program, it is ready to process commands.

7.4.2.3 TEST COMMAND DESCRIPTIONS

The MCSFS commands will be described in four sets: Programmed Output Commands, IOPB Commands, Customized Commands and Macro Commands.

7.4.2.3.1 Programmed Output Commands

FIRMWARE

This command reads and displays the controller firmware revision level.

GO_ASYNC

This command is identical to the **GO** command except that it does not wait for a done status or interrupt. The command is useful for getting a "read-ahead" or "write-ahead" loop started.

GO

This command causes the controller to execute the next IOPB on the chain. There should be as many **GO** commands issued as there are IOPBs to be processed.

OUT_CHAN

This command transmits the starting address of the first IOPB of a new chain to the controller.

READ_STS

This command reads the STATUS and GO registers, and displays the findings. This command does not read tape drive/formatter status since this information is returned in the IOPB following the controller's processing of the IOPB.

RESET_

This command writes the opcode for a reset command to the controller's command register. This causes the controller to reset all controller and tape drive/formatter hardware with the exception of the controller's Z-80 processor. All parameters previously stored in the controller RAM are initialized to their default values, and the IOPB chains is terminated. The controller's self-test is not invoked.

START

This command causes the controller to execute the first IOPB of a new chain at the starting address. Any **GO** commands remaining active are cleared.

RESET_C

This command is the same as **RESET_**, except that only the controller is reset, not the tape drive.

7.4.2.3.2 IOPB Commands

The following commands are used to construct Input/Output Parameter Blocks (IOPBs) for subsequent queuing to the controller. These commands are not executed immediately. After a number of IOPBs have been built with these commands, they can be processed by issuing the **GO** command.

APPND

This command builds an IOPB which causes the controller to move the tape to the mark and write one file.

ERASE

This command builds an IOPB which causes the tape to be erased from its current position to the end of tape (EOT). All data on the tape is lost.

INIT_

This command builds an IOPB which initializes the controller. The command passes to the controller all the parameters necessary for proper operation, specifically the unit number and the interrupt vector. Only one device, unit 0, is supported by the controller.

RD_DSTS

This command builds a READ DRIVE STATUS IOPB, which simply reads the current drive/formatter status. This command is useful to determine status without having to move the tape with a command such as **READ** or **REWIND**.

RD_XSTS

This command builds an IOPB which reads the extended status. The drive will fill the current buffer with 64 bytes of drive-dependent status. The status can be displayed using the **DMP_BUF** command.

READ

This command builds an IOPB which causes the controller to read one file, starting from the current tape position. Read retries are done automatically by the tape formatter.

READ_FLY

This command builds a READ IOPB with optional command bit 0 set. This causes the controller and drive to do a read-on-the-fly. This requires that the **MBF OTF** command set is supported by the drive; otherwise the command is processed as a normal **READ**.

RETENS

This command builds and IOPB which causes the tape to move from its current position to EOT, and then rewind to BOT.

REWIND

This command builds an IOPB which causes the tape to move from its present position to the beginning of tape (BOT).

SEEK_EOD

This command builds an IOPB which causes the drive to seek to the end of data (EOD). The drive will position the tape heads over the first valid block after the end of recorded media.

SKIP_FRD {*n*}

This command builds an IOPB which causes the controller to skip forward the number of files specified by *n*. The skip is done from the current position, and only if the subsystem is in streaming mode. The tape must be written before this command will function properly.

TST1, TST2

These commands build IOPBs to run self-test 1 and self-test 2, respectively.

WFM_FLY

This command builds a WRITE IOPB with optional command bit 0 set. This causes the controller and drive to do a write-with-filemark-on-the-fly. This requires that the **MBF OTF** command set is supported by the drive; otherwise the command is processed as a normal write with filemark.

WRITE

This command builds an IOPB which causes the controller to write one file, starting with the current tape position.

7.4.2.3.3 Customized Commands

The following commands describe operations unique to the MCS Function Select Test.

BACK_PNT

This command makes the last configured IOPB on the chain point back to the first. This is useful for streaming operations. A single IOPB can point to itself, so the same IOPB operation can be duplicated many times over. Note that if the chain is already back-linked, the chain would loop forever looking for the last IOPB on the chain. This condition is sensed by the program, which then displays a warning is displayed and clears the entire chain.

CMP_BUF *x,y*

This command compares the data in buffer *x* data to the data in buffer *y*, where *x* and *y* are buffer numbers, 1-5. Any data compare errors are reported.

DECL_Q *new_q_addr*

DECL_Q sets a new base address for the IOPB chain. The parameter, *new_q_addr*, specifies the new address. If the new address does not exist in the current memory configuration, the user is warned that this is the case. However, all subsequent IOPB operations are processed as though the address exists. This is useful for checking the controller DMA access to non-existent memory.

DELAY *n*

This command causes the test program to enter a delay loop, where *n* is the delay in increments of 5000 ms. The range of *n* is 0-600 decimal (0-258 hex).

DICTNARY

This command displays some useful definitions used throughout the MCSFS test.

DIS_FM

This command disables the writing of filemarks by resetting bit 6 in the flags byte field of the IOPB.

DIS_INT

This command disables all controller interrupts by clearing bit 9 in the flags byte of each IOPB built following this command.

DIS_STRM

This command causes the controller to disable streaming mode by clearing bit 12 of the flags byte in each IOPB built following this command. This causes the tape to stop on the next filemark.

DMP_BUF *n*

This command displays the contents of buffer *n*, where *n* is a value 1-5. The display is in both hexadecimal and ASCII form.

DMP_IOPB *n*

This command displays the *nth* IOPB on the chain, where *n* is a hexadecimal number 1-F representing the position of the desired IOPB.

DMP_PAR

This command displays the current state of all user-selected parameters.

ENB_FM

This command enables the writing of filemarks by setting bit 6 in the flags byte field of the IOPB. Also, after this command has been issued, any exception detected by the controller caused by a filemark being read will be masked out by the diagnostic. This allows the operator to read over a tape that has filemarks on it.

ENB_STRM

This command causes the controller to enable streaming mode by setting bit 12 of the flags byte of each IOPB built following this command. This causes the tape not to stop on filemarks.

END_INT

This command enables all controller interrupts by setting bit 9 in the flags byte of each IOPB built following this command.

FLUSH_Q

This command clears out the current IOPB queue area and reset the pointers associated with the queue.

INIT_BUF *n*

This command initializes the currently selected buffer with the data pattern *n*, where *n* is one of the following:

n	Pattern
0	all zeros
1	all ones
2	floating 0 byte
3	floating 1 byte
4	incremented word
5	pseudo-random
6	alternating 1's and 0's - bit
7	alternating 1's and 0's - byte
8	incremented long word

SEED *n*

This command specifies the seed value to be used for random number generation associated with the pseudo-random data pattern. The value of *n* can be any valid hexadecimal number between 0 and FFFFFFFF. Setting the seed value prior to two operations using the pseudo-random data pattern assures that the pattern is the same for both operations.

SEL_BUF *n*

This command selects a buffer *n* as the current buffer used in all data transfer operations. The value of *n* may be either 1 or 2.

SEL_UNIT *unit_no*

This command selects the current test unit. Legal values are 0-3, with 0 as the default.

SET_BUFF *buff_no*, *new_base_addr*

This command sets the base address of *buff_no* to the new location, *new_base_addr*. The parameter *buff_no* specifies one of the floating buffers, 3-5. Non-existent memory may be specified, which results in a warning message. This is useful for forcing non-existent memory data transfers.

WORD_CNT *n*

This command specifies the word count of subsequent data transfer operations. It is entered in hexadecimal form.

7.4.2.3.4 Macro Commands

STRM_RD *n* {1}

This command is actually the following series of commands:

RESET, READ, READ, BACK_PNT, OUT, START

followed by a series of **GO** commands. The **GO**'s are sent to the controller until an exception is detected, usually an EOT condition. The parameter *n* specifies the desired pattern (see the **INIT_BUF** command). The optional parameter, 1, causes on-the-fly commands.

STRM_WRT *n* {1}

This command is actually the following series of commands:

RESET, WRITE, WRITE, BACK_PNT, OUT, START

followed by a series of **GO** commands. The **GO**'s are sent to the controller until an exception is detected, usually an EOT condition. The parameter *n* specifies the desired pattern (see the **INIT_BUF** command). The optional parameter, 1, causes on-the-fly commands.

7.4.2.4 ERROR REPORTING

Status errors may be reported for the following conditions:

- Controller busy upon beginning command execution
- Program timed out waiting for controller to complete command
- Anything but bit 15 set in the primary status word of the processed IOPB, indicating the following abnormal conditions:
  - tape error
  - tape unit not ready
  - tape unit overflow
  - data transfer error
  - read/write abort
  - write protected cartridge
  - chain processing terminated
  - IOPB parameter error
  - IOPB format error

In addition, any unexpected interrupt will generate an error.

7.4.3 MTS Function Select Test

The MTS Function Select Test provides low level commands for testing the 1/2 inch Tape Streamer.

The test provides a full range of commands, which allow tests from simple register reads and writes to more complex I/O functions. The test enables the operator to observe and diagnose the functionality of the tape controller, the SCSI adapter and the drive/formatter.

7.4.3.1 TEST REQUIREMENTS

This test requires the following minimal hardware:

- CMB with 256 KB of memory
- MTS tape drive/formatter, SCSI adapter and controller
- Floppy disk drive (MAI 2000 only) or MCS drive for loading the diagnostic.

The only software requirements are the diagnostic executive and the MTSFS program.

7.4.3.2 LOADING THE TEST

The name of this test is MTSFS. To load the test, enter:

load mtsfs

at the diagnostic executive prompt, or at the prompt for any Logic or Function Select test.

The test is on the diagnostic medium labeled DIA. If you have booted the diagnostic from a different medium type, it is necessary to change the load device using the DEVICE command (refer to Section 4).

After loading the program, it is ready to process commands.

7.4.3.3 COMMAND DESCRIPTIONS

The MTS Function Select Test commands are described in four sets: Primitive Commands, SCSI Tape Adapter Commands, General SCSI Commands and Sequential SCSI Commands.

7.4.3.3.1 Primitive Commands

The following commands provide general control features for use within the MTS Function Select test.

BLOCKSIZE *n*

This command sets the blocksize parameter for the next mode select command. The parameter must be in the range of the minimum and maximum blocksize for the drive/controller. The minimum blocksize is initially 0, and the maximum is initially 64 KB, with the default 512. The minimum and maximum are updated by the read block limits command, and the current blocksize is updated by the mode sense command.

BUFFER {*n*}

This command permits the user to specify which buffer area is designated as the read buffer. There are two buffers, numbered 1 and 2. When the read buffer is selected, the other buffer becomes the write buffer. If no parameter is given, the read buffer becomes buffer 1.

BURNIN

This command executes a predefined sequence of commands repeatedly on the tape subsystem.

BYTES {C,P,R,W}

This command prompts the user for a series of byte data to enter into the block specified by the parameter. The available blocks are the **C**ommand block, **P**arameter buffer, **R**ead buffer and **W**rite buffer. If no parameter is specified, the Command block is assumed. Byte data can be entered at each prompt, or multiple bytes may be entered by placing a space between bytes.

COMPARE

This command compares the individual bytes of the read and write buffers, noting any differences on the VDT. A maximum of 20 consecutive errors will be reported.

CREATE {*n*}

This command creates a block of data to be used for the next write or compare operation. The parameter specifies the data type to create, as shown in Table 7-6. If no parameter is specified, the previously selected pattern is used, incrementing to the next pattern for the next use, wrapping around from the last type to the first. This allows the use of all data types in a loop without specifying the type for each loop.

Table 7-6. MTSFS Data Types

TYPE	DESCRIPTION	HEX DATA
0	All zeros	0000 0000 0000 0000 0000
1	All ones	FFFF FFFF FFFF FFFF FFFF
2	Floating 0	FFFE FFFD FFFB FFF7 FFEF
3	Floating 1	0001 0002 0004 0008 0010
4	Random	F946 EA89 12DD 32A0 00FF
5	All fives	5555 5555 5555 5555 5555
6	Di-bit	CCCC CCCC CCCC CCCC CCCC
7	Tri-bit	E383 38E3 8E38 E38E 38E3
8	Quad-bit	F0F0 F0F0 F0F0 F0F0 F0F0
9	Zero di-bit	6DB6 DB6D B6DB 6DB6 DB6D
A	Incrementing	0001 0002 0003 0004 0005
B	Zero tri-bit	7777 7777 7777 7777 7777
C	Zero quad-bit	7BDE F7BD EF7B DEF7 BDEF
D	One di-zero	9249 2492 4924 9249 2492
E	One tri-zero	8888 8888 8888 8888 8888
F	One quad-zero	8241 0824 1082 4108 2410
10	Increase freq.	FF00 FE03 F03E 0F0E 32AA
11	Decrease freq.	24AA 6663 8E38 7878 783F
12	Jitter	9659 6596 5965 9659 6596
13	Worst Case 1	D936 4DB6 4ED9 364D B64E
14	Worst Case 2	6DB6 DB0F 6DB6 DB0F 6DB6
15	Worst Case 3	DADA CA58 C2FE DADA CA58

DELAY *n*

This command causes the test to delay for the specified number of milliseconds.

DICTIONARY {R,S,C,E,P,K,B}

This command displays a list of definitions for various aspects of the SCSI Tape Adapter and Functional Select. If no parameter is specified, all definitions are displayed. The selectable definitions are: **R**egisters, **S**tatus register, **C**ontrol Register, **E**rror codes, **P**hase byte, sense **K**ey and status **B**lock.

DUMP {C,S,P,R,W,M}

This command displays a block of memory on the VDT. The blocks available for display are: **C**ommand (the six byte SCSI command block), **S**tatus (the block of status bytes from the previous command), **P**arameter (the variable length SCSI parameter block used to control some SCSI operations), **R**ead (the current data input buffer), **W**rite (the current data output buffer) and **M**emory (absolute memory locations specified as start and end bytes). If none of the options are specified, the Command block is displayed.

ENABLE {I,D,F,L,X,B}

This command permits the user to select various optional parameters used to build commands and transfer data. The valid options to enable are: **I**nterrupts (host interrupts), **D**MA (allowing DMA transfers), **F**lag (to set the flag bit in commands), **L**ink (to set the link bit in commands), **F**ixed (to set the fixed block size bit in I/O commands), and **B**uffered (to select buffered I/O). Preceding an option with a minus (-) will disable the option. Use the **PARAMETER** command to display the currently enabled parameters.

ERRORCLEAR

This command clears the error counters.

ERRORLOG

This command displays the errorlog counts. The number of fatal errors, status errors, status errors, total errors and soft retry count are displayed. Also, the total counts for all return status word values and sense key values are displayed.

LENGTH {*n*}

This command permits the user to specify the default block length to be used in transfers of the Parameter, Read and Write buffers to the SCSI controller. Any hex value of 1 to 2000 may be specified. If no value is given, the program defaults to a value of 100 hex.

LOOP *n*

This command sets the loop counter for repeated execution of the next eligible command. Loopable commands are indicated in their descriptions.

PARAMETER

This command displays the current value of all parameters and options enabled.

SPEED *n*

This command sets the speed parameter for the next mode select command. The parameter is a 4-bit value 0-3 expected by the SCSI controller:

n	Speed
0	high speed (default)
1	low speed
2	high speed
3	drive selected speed

TIMEOUT {*n*}

This command specifies the timeout loop count. The value must be in the range 0-7FFF hex. If no parameter is specified, the default value of 200 is restored.

UNIT {*n*}

This command specifies the logical unit to use in building the SCSI command. Logical unit number 0-7 may be used. If no unit is specified, the program defaults to unit 0.

7.4.3.3.2 SCSI Tape Adapter Commands

The following commands provide functions which directly control the operation of the adapter or provide data transfer between the host and the adapter. These commands rely on adapter specific addresses (base and register addresses).

CLEAR BUS

This command writes to the bus error clear register, clearing the bus error condition.

CONTROL {*n*}

This command displays the current value of the adapter control register. If the optional parameter is specified, the control register will have the byte written to it.

ISSUE {P,R,W}

This command permits the user to send the command that is in the command block to the SCSI adapter board specified by the previous command. The optional command argument specifies the buffer to use for the transfer, as follows: **P**arameter, **R**ead and **W**rite.

RDATA

This command reads and displays a byte of data from the adapter input register. Loopable.

RESET

This command forces a complete hard reset of the SCSI adapter and all attached controllers.

RREAD *n*

This command causes the specified adapter register to be read and displayed. The display is given in both decimal and hexadecimal. The parameter specifies the controller register byte to be displayed. Loopable.

RWRITE *x,y*

This command writes to the register specified by the first parameter the value of the second parameter. The value specified must be in the range 0-FF to fit a single byte register width. Loopable.

SELECT

This command selects the SCSI Tape Adapter for command output.

STATUS {*n*}

This command causes the status register to be read and compared with the specified value. The status will be poled the number of times specified by TIMEOUT or until the value is matched. If no value is indicated, the current status register contents will be displayed only.

WDATA *b*

This command writes the specified byte of data to the adapter output register. Loopable.

7.4.3.3.3 General SCSI Commands

The following commands are generally available for all device types on a SCSI bus.

COPY

This command issues the SCSI copy command to the SCSI bus adapter. Note that this command requires that the parameter block be initialized by the user. Loopable.

INQUIRY

This command issues the SCSI Inquiry command to the SCSI bus adapter. The returned data is interpreted and stored as required.

RDR or ReceiveDiag

This command receives the diagnostic status from the addressed SCSI controller. The results are interpreted and displayed.

RS or RequestSense

This command issues the SCSI Request Sense command in extended format. The returned data will be interpreted, and the results placed in the appropriate status registers.

SDF *n* or SendDiag *n*

This command sends a diagnostic command to the SCSI controller. The parameter value specified is the bit pattern to use for the command flags. The parameter block and current buffer length are used in this command. Note that the receive diagnostic results command must be used to determine the resulting status.

TUR or TestReady

This command issues the SCSI Test Unit Ready command. The complete status is interpreted and displayed. Loopable.

7.4.3.3.4 Sequential SCSI Commands

The following commands control a sequential access read/write device on the SCSI bus, such as the MTS.

Erase {*n*}

This command issues a SCSI erase command to the device. If the optional parameter is specified and non-zero, a long erase to the end of medium will occur.

LD *n* or LoadTape *n*

This command will issue a SCSI load command to the device. The parameter specified contains the bit pattern for indicating retension and load or unload: 0 = unload, 1 = load, 2 = retension.

MS or ModeSense

This command issues a SCSI mode sense command to the device. The returned data is placed in the parameter buffer. Data is interpreted and displayed.

MSEL *n* or ModeSelect *n*

This command issues a SCSI mode select command to the device. The parameter specifies a short or long buffer for the select: 0 = short, 1 = long. The parameter buffer will be used to pass the selection parameters to the controller. Note that the parameter buffer must be initialized by the use of the primitive commands.

RBD {F} or RecoverBuffer {F}

This command issues a SCSI recover buffered data command to the device. The optional parameter F causes a fixed length block read. The fixed length parameter requires that the device be selected for fixed block mode. The returned data is placed in the current read buffer.

RBL or ReadLimits

This command issues a SCSI read block limits command to the device. The returned data is interpreted and displayed.

READ {F}

This command issues a SCSI read command to the device. The optional parameter F causes a fixed length block read. The fixed length parameter requires that the device be selected for fixed block mode. The returned data is placed in the currently selected read buffer. Loopable.

ReleaseUnit {*n*}

This command issues a SCSI release command to the device. The optional parameter may be used to release a previous third party reservation. The release unit may then be used by other bus controllers.

ReserveUnit {*n*}

This command issues a SCSI reserve command to the device. The optional parameter may be used to reserve the unit for a third party on the bus. The reserved unit may then be used without interruption by any other bus controller.

REWIND

This command issues a rewind request to the SCSI bus. The returned data is interpreted and displayed.

RR {F} or ReverseRead {F}

This command issues a SCSI read reverse command to the device. The optional parameter F causes a fixed length block read. The fixed length parameter requires that the device be selected for fixed block mode. The returned data is placed in the currently selected read buffer. When the command has completed, the data is reversed bytewise to permit comparison with the write buffer. Loopable.

SPACE *x y*

This command issues a SCSI space command to the device. One of the following parameter pairs must be specified: 0 *y* (space the number of blocks), 1 *y* (space the number of file marks, 2 *y* (space the number of sequential file marks). The resulting status is interpreted and displayed.

VERIFY {*n*}

This command issues a SCSI verify command to the device. If the optional parameter is suppled and non-zero, the verify will also compare data. The current write buffer will be used for the compare data. Loopable.

WFM {*n*} or WriteFileMarks {*n*}

This command issues a SCSI write filemarks command to the device. The parameter specifies the number of sequential file marks to write. If no number is specified, one file mark is written.

WRITE {F}

This command issues a SCSI write command to the device. The optional parameter F causes a fixed length block write. The fixed length parameter requires that the device be selected for fixed block mode. The data written is taken from the currently selected write buffer. Note that the data pattern must be selected using the primitive commands. Loopable.

7.4.4 Four-way Function Select Test

The Four-way Function Select Test provides flexible tools for controlling the Serial Communications Controller (SCC). Many of the functions set up data blocks in CMB memory, according to user specifications. Others alter SCC write registers.

7.4.4.1 Test Requirements

This test requires the following minimal hardware:

- CMB with 128 KB of memory
- One or more 8-ways to be tested
- 4-way loopback connector (907529) and cable (907530)
- Floppy disk drive (MAI 2000 only), MCS or MTS drive for loading the diagnostic.

The only software requirements are the diagnostic executive and the FWFS program.

7.4.4.2 Loading the Test

The name of this test is FWFS. To load the test, enter:

load fwfs

at the diagnostic executive prompt, or at the prompt for any Logic or Function Select test.

The test is on the diagnostic medium labeled DIA. If you have booted the diagnostic from a different medium type, it is necessary to change the load device using the **DEVICE** command (refer to Section 4).

After loading the program, it goes through a process of autosizing and self-test. If no boards are found ready to test, a message is displayed reporting this. Following a successful self-test, the test is ready to process commands.

7.4.4.3 Command Descriptions

BAUD {*x*,*y*}

This command sets the baud rate. If arguments are not given, the current baud rate is returned. Baud rates less than 50 or greater than 19200 will result in an "Invalid Baud Rate" message. The arguments are:

x port A, B, C or D
y baud rate in decimal

The default baud rate is 9600.

BOARD *x*

This command changes the board being addressed to the board specified by *x*. The argument value may be 1, 2, 3 or 4. Note that the board number is determined by addressing rather than by position in the card stack.

BUFFER *x*

This command displays the contents of the received buffer. The argument specifies the receive port (A, B, C or D).

CB *x*

This command displays the contents of the command block. The argument specifies the port (A, B, C or D).

CLEARDP *x*

This command clears the data packet specified by *x*.

COMPARE *x,y*

This command compares the two specified buffers. The arguments are:

- *x* the data packet that was transmitted
- *y* the receive port (A, B, C or D)

DATA *w,x,y,z*

This command initializes data packets for transmission. A data packet must be prepared before a transmission can take place. The command arguments are as follow:

- *w* Data packet number, a hex number from 1 to FF
- *x* Byte count for the data, a hex number from 1 to FF
- *y* Data pattern option, described below
- *z* Address location for the data packet (optional)

The data pattern options for the third parameter are:

- 0 Fill the data area with zeros
- 1 Fill the data area with ones (7F)
- 2 Walk ones through byte fields of zeros
- 3 Walk zeros through byte fields of ones
- 4 Increment each byte, starting from 0
- 5 Bytes are equal to lower byte of their respective address
- 6 Alternate zeros and ones
- 7 Fill with printable ASCII set characters

DEFAULT *x*

This command sets the SCC write registers for the port specified by *x* (A, B, C OR D) to default values.

DUMP *x*

This command displays the contents of the data packet specified by *x*. The byte count and address of the packet in memory are also reported.

ECHO

This command causes the ports to echo each character received back through the port. This is useful for hooking up terminals to the 4-way ports and seeing characters typed echo back to the screen. ECHO is terminated when the ESCAPE key is pressed on any terminal on the 4-way. ECHO cannot be used in a BUILD loop.

ERRORS *x*

This command displays the error report. The argument specifies the receive port for which errors are reported (A, B, C or D).

LOCAL *x,y{,z}*

This command sets transmit and receive in local loopback mode. The arguments are:

- *x* number of data packet to transmit
- *y* channel to test, A, B, C or D
- *z* address of receive buffer (optional)

If *z* is not specified, the diagnostic will find a buffer space in memory.

RDATA

This command reads the data out register and writes the data into the D0 register. the hex value of the data read is also displayed on the screen.

REMOTE *w,x,y,z*

This command sets transmit and receive in remote loopback mode. The arguments are:

- *w* number of data packet to transmit
- *x* transmit channel, A, B, C or D
- *y* receive channel, A, B, C or D
- *z* address of receive buffer (optional)

If *z* is not specified, the diagnostic will find a buffer space in memory. External loopback connectors are required for **REMOTE** (see section 7.4.4.1).

RECEIVE *x,y,z*

This command initializes the receive data buffers. The arguments are:

- *x* channel that data is to be received through (A, B, C or D)
- *y* byte count of the buffer
- *z* address of the receive buffer (optional)

The first two arguments must be specified.

RESET {*x*}

This command resets the selected 4-way controller board. If no parameter is specified, a software reset is sent. The parameter may be A, B, C or D, specifying the channel on the board to reset.

RSTATUS

This command reads and displays the contents of the transmit status register.

RXBITS *x,y*

This command changes the number of bits per character to be received. The arguments are:

x port A, B, C or D
y either 5, 6, 7 or 8 bits per character

The default size for all ports is 7 bits.

STATUS

This command reads and returns the contents of the SCC read registers. Status from both SCCs is returned.

TRANSMIT *x,list*

This command transmits data on a serial channel. The arguments are:

x number of data packet to transmit
list list of 1 to 4 channels to use as transmit ports (A, B, C or D)

All arguments must be specified.

TXBITS *x,y*

This command changes the number of bits per character to be transmitted. The arguments are:

x port A, B, C or D
y either 5, 6, 7 or 8 bits per character

The default size for all ports is 7 bits.

WCOMMAND *x*

This command writes a byte, specified in hexadecimal by *x*, to the command register.

WDATA *x*

This command writes a byte, specified in hexadecimal by *x*, to the data register.

WR *x,y,z*

This command sets the value of an SCC write register. The arguments are:

x port, A, B, C or D
y write register number, in decimal
z the new value, in hex, to be written into the write register.

The corresponding register in both SCC ports will be changed.

XOFF {*x*}

This command changes the X-off character to the character specified by *x*. The default character is 13 hex (**CTRL + Q**).

XON {*x*}

This command changes the X-on character to the character specified by *x*. The default character is 11 hex (**CTRL + S**).

7.4.4.4 ERROR REPORTING

7.4.4.4.1 Error Messages

The following message may be displayed by the FWFS test.

- Z80 busy - time out

  The test polls the busy bit in the 4-way transmit status register, waiting for it to clear. This error occurs if the bit does not clear in the allotted time.

- Parity Error

  A special receive interrupt occurred indicating a parity error.

- Receiver Overrun Error

  A special receive interrupt occurred indicating the receive FIFO buffer has overflowed.

- Framing Error

  A special receive interrupt occurred indicating a framing error.

- Unexpected Interrupt

  An interrupt vector number was received for a port not programmed to be active.

- Undetermined Receive Error

  A special receive interrupt occurred, but the diagnostic cannot interpret the status byte.

- Command complete interrupt not received - time out

  The 4-way failed to complete its task or send an interrupt within the allotted time.

- External Status Interrupt

  A SCC port sent an external status interrupt.

- Invalid Input

  The parameters given were invalid.

- Address Range Error

  An invalid address was given for a data packet or receive buffer. Probable causes are that program memory, another data packet or another receive buffer exits at that address, or there is insufficient memory at that address.

- Already Initialized

  The DATA command specified a data packet number which has already been initialized.

- End of DP table - DP not initialized.

  The user attempted to use DATA, but the data packet table space is exhausted. Use CLEAR to clear some data packets, or INIT to clear the entire table, and try again.

- Not Found

  A request was made to use a data packet or receive buffer, but the packet of buffer could not be found in the table.

- Invalid Baud Rate

  An invalid baud rate was specified in the BAUD command.

- Time out on Self-Test

  The initialization routine failed to complete the self-test within the allotted time.

- Fatal Error flagged by Self-Test!

  The self-test determined that a board is completely inoperable. The failed board is also indicated by the message.

- ROM Failure

  The self-test detected an error in the ROM performance. This error is fatal.

- DMA Bus Failure

  The self-test detected an error in the DMA logic. This error is fatal.

- RAM Failure

  The self-test detected a data compare error in RAM. The area of RAM is also indicated by the message.

- SCC Failure

  The self-test determined that an SCC is inoperable. The SCC at fault is also indicated by the message.

- Time Out On Receive Character

  The diagnostic was expecting more receive characters, but did not receive them in the allotted time.

- Error In Command Execution Flagged By 4-way

  Command block status reported either an illegal command or a bus error on writing to the CMB.

- Data Compare Error

  The diagnostic found differences between the data packet (transmit buffer) and the receive buffer.

- Character Received During This Test

  The diagnostic was executing a data transmission and received a character unexpectedly.

- Byte Counts Are Not Equal

  An attempt was made to compare buffers of different size.

- Board Not Present or Not Initialized

  An attempt was made to access a 4-way controller that was either not present or not seen by the host during initialization.

7.4.4.4.2 Message Headers

Some error messages include the following additional information.

- Transmit Status

  The contents of the 4-way status register.

- Rx Port

  The receive port.

- Tx Port

  The transmit port.

- Events

  The number of times an error occurred.

- Bytes

  The number of bytes in the data packet that was to be transmitted.

- Board

  The board found to be in error during the self-test

- Port

  The port found to be inoperable by the self-test

7.4.5 Eight-way Function Select Test

The Eight-way Function Select Test provides flexible tools for controlling the Serial Communications Controllers (SCC). Many of the functions set up data blocks in CMB memory, according to user specifications. Others alter SCC write registers.

7.4.5.1 TEST REQUIREMENTS

This test requires the following minimal hardware:

- CMB with 128 KB of memory (MAI 2000), or 2 MB (MAI 2500/3000/4000)
- One or more 8-ways to be tested
- 8-way loopback connector (907529), cable (907530) and 4-way/8-way adapter (907968)
- Floppy disk drive (MAI 2000 only), MCS or MTS drive for loading the diagnostic.

The only software requirements are the diagnostic executive and the EWFS program.

7.4.5.2 LOADING THE TEST

The name of this test is EWFS. To load the test, enter:

```
load ewfs
```

at the diagnostic executive prompt, or at the prompt for any Logic or Function Select test.

The test is on the diagnostic medium labeled DIA. If you have booted the diagnostic from a different medium type, it is necessary to change the load device using the **DEVICE** command (refer to Section 4).

After loading the program, it goes through a process of autosizing and self-test. If no boards are found ready to test, a message is displayed reporting this. Following a successful self-test, the test is ready to process commands.

7.4.5.3 COMMAND DESCRIPTIONS

ABORT *list*

This command stops all activities on the ports included in list. Ports are 0-7.

BAUD *x*{,*y*}

This command sets the baud rate for port *x* (0-7) to the value *y* (in decimal). If no arguments are given, the current port baud rates are displayed. Baud rates less than 50 and greater than 19200 result in and "Invalid Baud Rate" message. The default baud rate is 9600.

BOARD {*x*}

This command selects the 8-way board to test. The board is specified by *x*, a value in the range 0-5. When no argument is entered, the currently selected board and its address are displayed.

BRDSTS

This command reads and displays the value of the board status register.

BUFFER *x*

This command displays the contents of the buffer specified by *x* (x=0-7).

CLEARDP *x*

This command clears the data packet (DP) specified by *x*. The packets entry is removed from the DP table.

COMPARE *x,y*

This command compares the contents of the data packet (*x*) with the receive port buffer (*y*). The buffers are compared byte for byte, and any discrepancies are reported. This is useful following a remote port to port test. Note that if the transmit packet is larger than the receive buffer, data will be lost.

DATA *w,x,y{,z}*

This command initializes a data packet, preparing it for transmission. Data packets must be established before data can be transmitted. The arguments are:

- *w* data packet number (1-FF hex)
- *x* word count of the data (1-7FFF hex)
- *y* data pattern option, (0-7 as described below)
- *z* address location for the packet (optional)

The data patterns for the third argument are:

0 filled with zeros
1 filled with ones (7F)
2 walking ones through byte fields of zeros
3 walking zeros through byte fields of ones
4 Increment each byte, starting from 0
5 bytes are equal to lower byte of their respective addresses
6 alternate zeros and ones
7 fill with printable ASCII characters

A data packet number cannot be reinitialized until it has been cleared.

DEFAULT {*list*}

This command causes default values to be written to the SCC write registers of the ports in *list* (0-7). Default values are: 7 data bits, 1 stop bit, odd parity and 9600 baud.

DUMPCB *x*

This command displays the contents of the command block for port *x* (0-7).

DUMPDP *x*

This command displays the contents of the data packet *x*. The display includes the byte count and the address of the packet in memory.

ECHO

This command causes the 8-way to echo any character received on a port back out the same port. This is useful for hooking up data terminals and verifying that characters typed in are echoed back. Echoing is terminated when **ESCAPE** is pressed on any terminal connected to the 8-way.

ERRORS *x*

This command displays the error report for the receive port *x*. This command should be used following **REMOTE**. The values given in the report are based on the current error counter values.

POLDATA *x,y*

This command sends the single byte of data (*y*) to the port (*x*) with no interrupt control. It is necessary to issue the **POLLDON** command first.

POLLDOFF

This command turns off polled I/O. Interrupts are re-enabled.

POLLDON

This command turns on polled I/O. No interrupts will occur in this mode. POLLDON is necessary to use the POLDATA command.

PORTID

This command displays the contents of the 8-way port ID register.

PORTOFF *list*

This command disables the ports specified in list (0-7).

PORTON *list*

This command enables the ports specified in list (0-7).

PORTSTS

This command displays the contents of the 8-way SCC status register.

PRTSETUP *w,x,y,z*

This command sets up a port with the specified characteristics. The arguments are:

- *w* port (0-7)
- *x* bits per character (7 or 8)
- *y* XON/XOFF flow control (0 = enable, 1 = disable)
- *z* DTR flow control (0 = enable, 1 = disable)

The 8-way is initialized with both X-on/X-off (soft control) and DTR (hard control) enabled. If DTR is disabled, X-on/X-off remains enabled. If X-on/X-off is disabled, only incoming DTR signals are used. With both X-on/X-off and DTR enabled, DTR takes precedence.

RDATA

This command reads and displays a word from the 8-way data out register. The value is displayed in hexadecimal.

RECEIVE *x,y{,z}*

This command initializes the receive data buffer. The arguments are:

- *x* the port on which data is to be received (0-7)
- *y* the byte count of the buffer (0-FFFF hex)
- *z* address of buffer (optional)

REMOTE *w,x,y{,z}*

This command transmits and received in remote loopback mode. The arguments are:

- *w* the number of the data packet to transmit
- *x* the transmit channel (0-7)
- *y* the receive channel (0-7)
- *z* address of the receive buffer (optional)

If *z* is not given, the diagnostic will find a buffer space in memory. External loopback connectors are required when using **REMOTE** (refer to Section 7.4.5.1).

RESET {*x*}

This command sends a reset command to the 8-way or to an individual port. The argument x specifies the port (0-7). If the port is not specified, the board is reset.

REVLEVEL

This command displays the hardware and firmware revision of the board.

RXBITS *x,y*

This command specifies the number of bits per character (*y*) on the port (*x*). The port is 0-7. The number of bits is either 7 or 8.

SCCSTS

This command displays the contents of the SCC read registers. The status from all SCCs is returned. If a port is hung, it may need to be reset before the status register can be read.

SINGLE *x,y,z*

This command sends a single byte (*y*) to the port (*x*) the number of times (z). The value of the byte is given in hexadecimal (0-FF), and the number of iterations must be in the range 1-7FFF hex. Note that the default character size for all ports is 7 bits.

TRANSMIT *x,list*

This command transmits data packet (*x*) on all ports included in *list*.

TXBITS *x,y*

This command specifies the number of bits per character (*y*) on the specified port (*x*). The available ports are 0-7, and the number of data bits is either 7 or 8.

WCOMMAND *x*

This command writes a word to the command register. The argument is the word to be written. The data must be given in hexadecimal.

XONABL {*list*}

This command enables X-onN/X-off for all ports in the *list*. If no ports are listed, XON/XOFF is enabled on all ports.

XONDIS {*list*}

This command disables X-on/X-off for all ports in the list. If no ports are listed, XON/XOFF is disabled on all ports.

7.4.5.4 ERROR REPORTING

7.4.5.4.1 Error Messages

The following message may be displayed by the EWFS test.

- Z80 busy - time out

  The test polls the busy bit in the 8-way transmit status register, waiting for it to clear. This error occurs if the bit does not clear in the allotted time.

- Parity Error

  A special receive interrupt occurred indicating a parity error.

- Receiver Overrun Error

  A special receive interrupt occurred indicating the receive FIFO buffer has overflowed.

- Framing Error

  A special receive interrupt occurred indicating a framing error.

- Unexpected Interrupt

  An interrupt vector number was received for a port not programmed to be active.

- Undetermined Receive Error

  A special receive interrupt occurred, but the diagnostic cannot interpret the status byte.

- Command complete interrupt not received - time out

  The 8-way failed to complete its task or send an interrupt within the allotted time.

- External Status Interrupt

  A SCC port sent an external status interrupt.

- Invalid Input

  The parameters given were invalid.

- Address Range Error

  An invalid address was given for a data packet or receive buffer. Probable causes are that program memory, another data packet or another receive buffer exits at that address, or there is insufficient memory at that address.

- Already Initialized

  The DATA command specified a data packet number which has already been initialized.

- End of DP table - DP not initialized.

  The user attempted to use DATA, but the data packet table space is exhausted. Use CLEAR to clear some data packets, or INIT to clear the entire table, and try again.

- Not Found

  A request was made to use a data packet or receive buffer, but the packet of buffer could not be found in the table.

- Invalid Baud Rate

  An invalid baud rate was specified in the BAUD command.

- Time out on Self-Test

  The initialization routine failed to complete the self-test within the allotted time.

- Fatal Error flagged by Self-Test!

  The self-test determined that a board is completely inoperable. The failed board is also indicated by the message.

- ROM Failure

  The self-test detected an error in the ROM performance. This error is fatal.

- DMA Bus Failure

  The self-test detected an error in the DMA logic. This error is fatal.

- RAM Failure

  The self-test detected a data compare error in RAM. The area of RAM is also indicated by the message.

- SCC Failure

  The self-test determined that an SCC is inoperable. The SCC at fault is also indicated by the message.

- Time Out On Receive Character

  The diagnostic was expecting more receive characters, but did not receive them in the allotted time.

- Error In Command Execution Flagged By 8-way

  Command block status reported either an illegal command or a bus error on writing to the CMB.

- Data Compare Error

  The diagnostic found differences between the data packet (transmit buffer) and the receive buffer.

- Character Received During This Test

  The diagnostic was executing a data transmission and received a character unexpectedly

- Byte Counts Are Not Equal

  An attempt was made to compare buffers of different size.

- Board Not Present or Not Initialized

  An attempt was made to access a 8-way controller that was either not present or not seen by the host during initialization.

7.4.5.4.2 Message Headers

Some error messages include the following additional information.

- Transmit Status

  The contents of the 8-way status register.

- Rx Port

  The receive port.

- Tx Port

  The transmit port.

- Events

  The number of times an error occurred.

- Bytes

  The number of bytes in the data packet that was to be transmitted.

- Board

  The board found to be in error during the self-test.

- Port

  The port found to be inoperable by the self-test.

- Err Code

  A byte indicating the type of error, as follows:

  1 8-way RAM failure
  81 Bad Command Status
  83 Bus Error on writing to CMB

- Sent,Received

  On a data compare error, this header displays the byte send and the byte received.

- CMND

  The 8-way command that was executed. The valid commands are:

Direct Command	Description
1	Polled I/O Data Valid
2	Abort (Port level)
3	Port Reset
4	Interrupt control
5	Activate Polled I/O
6	Deactivate Polled I/O
7	CMD Block Ready

Indirect Command	Description (requires IOPB)
1	Configure
2	DMA from Host
3	Port Status
4	Load Port Default
7	Single Byte Transfer
8	XON Enable/Disable
9	DTR Flow Control Enable/Disable
A	7-bit/8-bit
B	Download
C	Revision Status
D	Port Lump-Sum Setup
E	DMA to Host

7.4.6 LAN Function Select Test

The Local Area Network (LAN) Function Select Test provides a means for the user to define scope loops and to exercise the LAN controller hardware. This is done by a set of low level commands.

7.4.6.1 TEST REQUIREMENTS

This test requires the following minimal hardware:

- CMB with 128 KB of memory
- One or more LAN controllers to be tested
- Floppy disk drive (MAI 2000 only), MCS or MTS drive for loading the diagnostic.

The only software requirements are the diagnostic executive and the **LANFS** program.

7.4.6.2 LOADING THE TEST

The name of this test is LANFS. To load the test, enter:

```
load lanfs
```

at the diagnostic executive prompt, or at the prompt for any Logic or Function Select test.

The test is on the diagnostic medium labeled DIA. If you have booted the diagnostic from a different medium type, it is necessary to change the load device using the **DEVICE** command (refer to Section 4).

After loading the program, it goes through a process of autosizing and self-test. If no boards are found ready to test, a message is displayed reporting this. Following a successful self-test, the test is ready to process commands.

7.4.6.3 COMMAND DESCRIPTIONS

CAR *x*

This command writes *x* to the command address register. The argument x is three bytes in hexadecimal.

CARH *x*

This command writes *x* to the most significant byte of the command address register.

CARL *x*

This command writes *x* to the least significant byte of the command address register.

CARM *x*

This command writes *x* to the second byte of the command address register.

CLRBINT

This command clears the bus error bit in the status register.

CLRCINT

This command clears the LAN interrupt bit in the status register.

CMP *x*

This command compares the receive buffer for socket *x* against the message used for the last **REC** command. Only the number of bytes received are compared.

ECHO *x*

This command sends a Corvus defined packet to node *x*. Node *x* should then respond with a corvus defined acknowledgement. The sending host is given a return code indicating whether or not the **ECHO** command was acknowledged. The acknowledging node does not inform its host that it responded to an ECHO.

ENDREC *x*

This command causes socket *x* to be uninitialized. Any packets directed to this socket are ignored.

EQUE

This command displays the error queue, indicating the error numbers for the last 80 errors that occurred.

HELP

This command displays a help message pertaining to this diagnostic.

INITC

This command issues the Corvus INIT command to the LAN.

LAN *x*

This command specifies the LAN controller to test. The parameter x is either 0 or 1, specifying the board, according to address. If only one board is present, it is presumed to be 0. At program initialization, board 0 is selected, and remains so until LAN 1 is executed.

MESS *x* (*$y* | *string*)

This command specifies the data pattern for messages 6-9 used by the SEND and REC commands. The pattern can be defined either in hexadecimal, preceded by "$", or as any string of printable ASCII characters.

MONITOR (O | R | H)

This command controls the display of packets as they are received. Only one of the argument options may be used at a time. The arguments are:

- O Turn monitoring OFF (the default)
- R Displays all packets received in hex and ASCII
- H Displays a short header for each packet received, indicating the source host, destination socket and message length

PEEK *x*

This command displays the contents of memory address *x* (two bytes) internal to the Corvus chip set.

POKE *x y*

This command writes the byte *y* to the memory address *x* internal to the Corvus chip set.

RCTRL

This command reads and displays the control register contents.

RVCR

This command reads and displays the contents of the interrupt vector register.

REC *v* {*w* {*x* {*y* {*z*}}}}

This command receives a message. The arguments are the same, except that the destination node is not specified. The socket *v* is initialized to receive a packet of length less than or equal to w, with a control packet of length *y*. Message numbers *x* and *z* are used if the **CMP** command has been used, so that a buffer compare can be made against a pre-defined message pattern.

RNAR

This command reads and displays the node address register contents.

SEND *u v* {*w* {*x* {*y* {*z*}}}}

This command transmits a message according the parameter specifications. The pattern is repeated until the message length is exhausted. The arguments are:

- *u* the destination node address, in hex
- *v* the socket number (80 or 90 hex)
- *w* the length of the packet in bytes (max = 2047 decimal, default = 2047)
- *x* the message pattern number (0-9), as described below (default = 0)
- *y* the length of the control packet in bytes (max = 255 decimal, default = 0, no control packet)
- *z* the message number of the control field

The message numbers to be used for the fourth and sixth fields are:

Message#	Pattern
0	incrementing byte pattern
1	incrementing word pattern
2	incrementing long word pattern
3	all zeros
4	all ones
5	FFFF 0000 C3C3 AA55 AA55
6-9	User defined by the MESS command

STAT

This command reads and displays the status register contents.

WAIT {*x*} *y*

This command causes the controller to wait until either socket *x* receives a packet or *y* milliseconds pass, which ever comes first. If only one argument is used, the pause will for that many milliseconds.

WCTRL *x*

This command writes the hex value specified by x to the control register of the selected controller. A flag is set equal to the value of bit 0 of this hex value (the ENCINT bit). If this bit is zero, polling is used. If the bit is set (1), interrupts are used.

WHO

This command issues the WHO command to the Corvus chip set, and upon completion displays the node address. This command builds a command vector in memory, which is read by the LAN using DMA, and the node address is then returned also using DMA. This is similar to RNAR, which simply reads the register using MMIO.

WVCR *x*

This command writes the byte *x* (hex) to the interrupt vector register.

7.4.6.4 ERROR REPORTING

The following error numbers and messages may be returned by the LAN Function Select test.

ERR #	Message
1	Invalid socket number
2	Receive socket in use
3	Undefined return code
4	Message not acknowledged (retry count exhausted)
5	Message data portion too long for receiver's buffer
6	Message was sent to an uninitialized socket
7	Message control portion had incorrect length
8	Invalid destination node number in command vector
9	Message was sent successfully after xx retries
10	Packet was not acknowledged (retry count exhausted)
11	Tried to compare data for socket, still pending reception
12	Illegal message length encountered while executing CMP
13	Tried CMP on socket without updated return code
14	Buffer miscompare
15	Timeout occurred while waiting to write CAR
16	Timeout occurred while waiting for command to complete
17	Unexpected LAN interrupt
18	Bus error detected by LAN
19	Too many interrupts from LAN

7.4.7 MAI 2000 Floppy Function Select Test

The Floppy Function Select test provides the ability to create and execute test loop sequences to trouble shoot and verify basic operation of the floppy disk controller and drive.

7.4.7.1 TEST REQUIREMENTS

This test requires the following minimal hardware:

- CMB with 128 KB of memory
- VDT
- 5 1/4 inch floppy diskette drive

The only software requirements are the diagnostic executive and the FDFS program.

7.4.7.2 LOADING THE TEST

The name of this test is FDFS. To load the test, enter:

load fdfs

at the diagnostic executive prompt, or at the prompt for any Logic or Function Select test.

The test is on the diagnostic medium labeled DIA. If you have booted the diagnostic from a different medium type, it is necessary to change the load device using the **DEVICE** command (refer to Section 4).

7.4.7.3 COMMAND DESCRIPTIONS

In the following command descriptions, all numeric parameters (*n*) are given in hexadecimal notation.

ADDRT_RD {*n*}

This command reads the diskette and compares the data with the address pattern written by ADDRT_WR. The values for *n* may be:

0 = BOSS/IX format (default)
1 = S/10 format

ADDRT_WR {*n*}

This command writes the address pattern on the diskette. The values for *n* may be:

0 = BOSS/IX format (default)
1 = S/10 format

ADDRTST {*n*}

This command performs an address test on the disk drive and controller by writing the track, sector and side in each sector, then reading it back and verifying the data. The values for *n* may be:

0 = BOSS/IX format (default)
1 = S/10 format

BCOUNT *n*

This command specifies the byte count *n* to be used for subsequent RAM write and read commands. *n* can range from 1 to 2000 hex.

BDSTATUS

This command reads and displays the contents of the board status byte register.

BUFFER *n*

This command specifies which buffer is to be used for the next sector write or read operation, or fo *rthe **SBUFFER** command. *n* may be 1 or 10 hex.

CLRLOG

This command clears the error log accumulated during previous command routine executions.

COMPARE

This command compares the contents of buffer 1 with buffer 2.

DCOMMAND

This command displays the FDC (floppy disk controller) command list, the list of FDC commands which were last output to the FDC.

DDIS

This command disables FDC DRQ (data request) interrupt during **WSECTOR**, **RSECTOR**, **WTRACK** and **RTRACK** operations.

DELAY *n*

This command sets the delay to *n* milliseconds (*n* is in hex).

DENA

This command enables FDC DRQ (data request) interrupt during **WSECTOR**, **RSECTOR**, **WTRACK** and **RTRACK** operations.

DENSITY *n*

This command sets the density parameter. The values are:

0 = low density
1 = high density

DRIVE *n*

This command specifies the drive to be accessed by subsequent commands. When executed, the specified drive's motor is turned ON and delays one second. The values are:

0 = drive 0
1 = drive 1

DUMP *n*

This command displays the contents of the selected buffer, where *n* = 1 or 2. The size is the byte count set by BCOUNT.

EOFF

This command disables the 15 millisecond delay during write sector and read sector operations.

EON

This command enables the 15 millisecond delay during write sector and read sector operations.

FORMAT {*n*}

This command performs track write operations to format the diskette. The values for *n* may be:

0 = BOSS/IX format (default)
1 = S/10 format

FORMV {*n*}

This command performs a diskette format operation followed by a verify operation. The values for *n* may be:

0 = BOSS/IX format (default)
1 = S/10 format

GDATA

This command generates a randomly selected data pattern and fills the currently selected data buffer with the pattern.

GDRIVE

This command randomly selects a drive.

GRAN

This command selects random track, sector and side parameters for use by the following command routines.

IDIS

This command disables FDC controller interrupts at the end of an FDC operation.

IENA

This command enables FDC controller interrupts at the end of an FDC operation.

INCSECT

This command increments the sector number parameter.

INCTRACK

This command increments the track number parameter.

INSTEP

This command issues a step in command to the selected drive.

LBUFF *n*

This command loads the FDC buffer control byte register with the data specified by *n* (hexadecimal).

LCOM *n*

This command loads the command byte *n* in the FDC controller chip.

LCONTROL *n*

This command loads the FDC control latch with the hexadecimal value *n*. *n* can range from 00 to FF hex.

LDATA {*n*}

This command loads the FDC data register with the hexadecimal data given by *n*. If a parameter is not specified, the current contents of the register is read and displayed. *n* can range from 00 to FF hex.

LOGOUT

This command displays the accumulated run statistics from the error log.

LSECTOR {*n*}

This command loads the FDC sector register with the hexadecimal data given by *n*. If a parameter is not specified, the current contents of the register is read and displayed. *n* can range from 00 to FF hex.

LTRACK {*n*}

This command loads the FDC track register with the hexadecimal data given by *n*. If a parameter is not specified, the current contents of the register is read and displayed. *n* can range from 00 to FF hex.

MAXTRACK *n*

This command specifies the maximum track for use in the **GTRACK** and **GRAN** routines.

MINTRACK *n*

This command specifies the minimum track for use in the **GTRACK** and **GRAN** routines.

NSECTOR *n*

This command specifies the number of sectors to be used in the **WSECTOR**, **RSECTOR**, **WBUFFER** and **RBUFFER** commands. It overrides any previous **BCOUNT** command.

OUTSTEP

This command issues a step out command to the selected drive.

PREOFF

This command disables write precompensation during disk write operations.

PREON

This command enables write precompensation during disk write operations.

RADDRESS *n*

This command specifies the starting address in the FDC data RAM to which data is to be written or read by subsequent **RWRITE** and **RREAD** commands. *n* can be any hex value 00 to FF.

RATE *n*

This command specifies the stepping rate to be used in FDC positional commands, where the argument *n* is specified as follows:

n	Rate at 2MHz	Rate at 1 Mhz
0	3 MS	6 MS
1	6 MS	12 MS
2	10 MS	20 MS
3	15 MS	30 MS

RBUFFER

This command preforms a move of data from FDC data RAM to the system memory of the length specified by the previous **BCOUNT** command.

RESET

This command issues a reset to the FDC by programming the reset enable bit of the FDC buffer control byte.

RESTORE

This command issues a restore command (seek to track 00) to the selected drive.

RSECTOR

This command issues a read sector command to the FDC, starting at the sector specified by **SECTOR**, and of length equal to the number of sectors specified by **NSECTOR**.

RTRACK

This command reads a track, placing the data in the memory buffer previously specified by **BUFFER**.

SBUFFER *n*

This command initializes the selected buffer with the data pattern specified by *n*, as follows:

n	Pattern
0	all zeros
1	all ones
2	floating 0, byte
3	floating 1, byte
4	incremented byte
5	pseudo random
6	alternate 1s and 0s, bit
7	alternate 1s and 0s, byte
8	di-bit pattern
9	tri-bit pattern
A	quad-bit pattern
B	zero tri-bit pattern
C	zero quad-bit pattern
D	one di-zero pattern
E	one tri-zero pattern
F	one quad-zero pattern

SECTOR *n*

This command specifies the starting sector number for the next sector write or read command. The value of *n* may be from 1 to the maximum sector number; 8 for BOSS/IX format.

SEEK

This command issues a seek command to the track previously specified by the TRACK command.

SFORMAT *n*

This command specifies the format and sector size for all write and read sector, and diskette format commands. The values for *n* may be:

0 = BOSS/IX format (default)
1 = S/10 format

SIDE *n*

This command sets the side parameter for the **WSECTOR**, **RSECTOR**, **WTRACK** and **RTRACK** commands. The value of *n* may be 0 or 1.

SOFF

This command disables doing a side compare during FDC commands.

SON

This command enables doing a side compare during FDC commands.

STATUS

This command displays the contents of the FDC track, sector, status, control latch and board status registers.

STEP

This command issues a step command to the selected drive.

TRACK *n*

This command specifies the track number to be sought in the next **SEEK** command. n can range from 00 to 4F hex.

UOFF

This command disables performing an update during positional FDC command (**SEEK**, **RESTORE**, **STEP**, etc.).

UON

This command enables performing an update during positional FDC command (**SEEK**, **RESTORE**, **STEP**, etc.).

VERIFY {*n*}

This command performs a diskette verify operation, to verify that there are no media defects or errors. The values for *n* may be:

0 = BOSS/IX format (default)
1 = S/10 format

VOFF

This command disables track verify during positioning operations.

VON

This command enables track verify during positioning operations.

WBUFFER

This command moves data from system memory to the FDC data RAM of the length specified by **BCOUNT**.

WSECTOR

This command issues a write sector command to the FDC, starting at the sector specified by **SECTOR**, and of length equal to the number of sectors specified by **NSECTOR**.

WTRACK

This command performs a write track operation from the currently selected buffer.

7.4.7.4 ERROR MESSAGES

The following errors may be reported. The commands which may cause the error are indicated following the message.

- Invalid Input

  All commands except COMPARE.

- Bus Timeout Error Trap

  ADDRT_RD, ADDRT_WR, ADDRTST, FORMAT, FORMV, INSTEP, LCONTROL, LDATA, LSECTOR, LTRACK, OUTSTEP, RBUFFER, RSECTOR, RESTORE, RTRACK, SEEK, STEP, VERIFY, WBUFFER, WSECTOR, WTTRACK

- FDC Not Ready Prior to Issuing Command
- Timed Out Waiting for Rdy after Issuing Cmd
- Incorrect Status Following Cmd
- No Interrupt Occurred After Issuing Cmd
- Unexpected Interrupt Vector

  ADDRT_RD, ADDRT_WR, ADDRTST, FORMAT, FORMV, INSTEP, OUTSTEP, RSECTOR, RESTORE, RTRACK, SEEK, STEP, VERIFY, WSECTOR, WTTRACK,

- Data Compare Error

  COMPARE

7.4.8 Dual SCSI Controller Function Select Test

The Dual SCSI Controller Function Select test provides the ability to create and execute test loop sequences to trouble shoot and verify basic operation of the Dual SCSI controller (DSC) and drive.

7.4.8.1 TEST REQUIREMENTS

This test requires the following minimal hardware:

- CMB with 256 KB of memory
- Dual SCSI Controller board
- A SCSI drive and cable
- A loopback cable for the SCSI port
- A VDT

The only software requirements are the diagnostic executive and the DSCFS program.

7.4.8.2 LOADING THE TEST

The name of this test is DSCFS. To load the test, enter:

```
load dscfs
```

at the diagnostic executive prompt, or at the prompt for any Logic or Function Select test.

The test is on the diagnostic medium. If you have booted the diagnostic from a different medium type, it is necessary to change the load device using the **DEVICE** command (refer to Section 4).

7.4.8.3 COMMAND DESCRIPTIONS

In the following command descriptions, numeric parameters indicated by n are given in hexadecimal notation, and numeric parameters indicated by d are given in decimal notation. Other parameters are described.

The commands are divided into 12 groups:

- DSC Register Commands
- DSC Test Commands
- Write/Read SBIC Commands
- Program Control Commands
- Functional Unit Selection Commands
- Logical Block Number Selection Commands
- Buffer Definition and Control Commands
- Miscellaneous Commands
- General Purpose Commands
- Low Level Commands
- SCSI Commands
- Range Commands
- Unit Definition Commands

7.4.8.3.1 DSC Register Commands

BERRC

This command strobes the Clear Bus Error/DMA Reset register of the currently selected DSC and Port.

BUSRESET

This command issues a SCSI bus reset to the DSC control register. The control register is then cleared.

CCHANNEL

This command first resets the SCSI bus and then resets the active DSC port.

CCONTROL

This command issues a DSC port reset to the active port.

CONTROL *{n}*

With no argument, this command reads and displays the current value of the control register. With an argument, the value is written to the control register. In either case, the control register of the current DSC and port are used.

DIRECTIO (0 | 1)

This is a multi-function command which allows easy loading of the control register, and establishes the correct DMA address and direction. The alternate arguments do the following:

directio 0
- write 2 to the control register
- load the DMA register with the address of the read buffer
- issue a DMA reset
- write 8 to the control register if interrupts are disabled, or C hex if interrupts are enabled

directio 1
- write 2 to the control register
- load the DMA register with the address of the write buffer
- write 12 hex into the control register
- issue a DMA reset
- write 18 hex to the control register if interrupts are disabled, or 1C hex if interrupts are enabled

LDMA {*n*}

This command writes the DMA register of the selected DSC and Port with a specified address. The optional argument specifies the address. If no argument is given, the last DMA address specified is displayed and used.

ODDBYTE

This command writes FF hex to the Odd Byte Transfer register. This command can be looped.

REGS

This command displays the bit definitions of the DSC status and control registers. The contents of the status, control and vector registers are then displayed for the current DSC and Port.

STATUS {*n*}

With no argument, this command reads and displays the DSC Status register. The command may be looped, if no argument is given. If an argument is given (00-FF hex), the command causes the program to wait until this status is received, or until CTRL + C is entered.

STROBE

This command writes FF hex to the Diagnostic DMA register. This command can be looped.

VECTOR {*n*}

With no argument, this command reads and displays the current value of the vector register. With an argument, the value is written to the vector register. In either case, the vector register of the current DSC and port are used.

WRCONTRO {*n*}

This command writes data to the control register. If an argument is included, that data is written; otherwise, random data is used. The data is then read and compared. This command can be looped.

WRDATA {*n*}

This command writes data to the DSC address register. If an argument is included, that data is written; otherwise, random data is used. The data is then read and compared. This command can be looped.

WRVECTOR {*n*}

This command writes data to the vector register. If an argument is included, that data is written; otherwise, random data is used. The data is then read and compared. This command can be looped.

7.4.8.3.2 DSC Test Commands

BURNIN *x y*

This command executes a burnin test sequence. The first argument, x, specifies the DSC to test:

a port a of the DSC
b port b of the DSC
c both ports combined

The second argument, y, specifies the type of burn in to do:

1 DSC registers only
2 DSC registers plus write/read/compare disk buffer
3 same as 2 plus 50 random write/read/compare disk operations

TSTBIC

This command tests the SBIC chip by writing and verifying the first 16 hex SBIC registers. The tests increments through the data patterns shown in Table 7-7. For each pattern, the pattern is first written to register 16 and verified, then rotated one byte and written to register 15, and so on until all registers are tested. This command can be looped.

TSTDMA {*n*}

This command runs a test of the DMA logic on the selected DSC and port. If a data pattern (2 bytes) is specified, that pattern is used. If no pattern is specified, the pattern is incremented, using the patterns shown in Table 7-7. The data pattern is written to the read and write buffers, and compared. This command can be looped.

TSTINT

This command tests the Bus Error and Interrupt Logic of the selected DSC and port, by forcing a bus error. It also tests the interrupt logic from a channel reset which causes the SBIC to generate an interrupt. This command can be looped.

TSTODD {*n*}

This command runs a test of the DMA logic on the selected DSC and port. If a data pattern (2 bytes) is specified, that pattern is used. If no pattern is specified, the pattern is incremented, using the patterns shown in Table 7-7. The test is like that in TSTDMA, except that only the LSB of the data is used, making both bytes of the data in the registers the same. This command can be looped.

7.4.8.3.3 Write/Read SBIC Commands

AUXSTAT

This command reads and displays the contents of the SBIC Status register.

OWNID {*n*}

This command loads or reads the SBIC OWNID register. If an argument is specified, this value is written to the register. If no argument is specified, the current value is read and displayed.

REGISTER *n* {*b*}

This command reads the specified SBIC register (1-19 hex). If the second argument is also specified, that data is written to the specified register address (in hex). This command can be looped.

RDATA

This command reads and displays the contents of the last address specified by WADDRESS. This command can be looped. In most cases, hardware increments the address with each execution.

SCSISTAT

This command is like REG 17, except that it interprets the status for easier identification.

TCOUNT {*n*}

This command loads the three-byte SBIC transfer count register, if an argument is given (0-FFFFFF hex). If no argument is given, the current contents of the register is read and displayed.

WADDRESS *n*

This command writes a byte address (1-19 hex) to the SBIC address register. This selects the SBIC address to write or read (see WDATA and RDATA).

WDATA *n*

This command writes the specified data (0-FF hex) to the last address specified by WADDRESS. This command can be looped. In most cases, hardware increments the address with each execution.

7.4.8.3.4 Program Control Commands

AUTOMODE {a|d|e|f}

This command enables program auto mode. (See IMMEDIATE). In auto mode, certain commands are automatically executed by the program, such as loading the DMA register and checking ending status. The options do the following:

a abort auto mode on an error, allowing use of the SENSE command
d Don't display errors
e Don't display compare errors after the first
f Don't display any compare errors

BCOUNT {*n*}

This command displays or sets the bytes per sector format for the device. If the argument is specified (1-7FFF hex), the byte count is set. Selecting a byte count different from that expected by the device will cause data transfer errors.

BUSYTRY {*n*}

This command specifies the retry count for a command when it encounters a busy device. The argument may be in the range 0-7FFF hex. To disable retries, specify 0. If no argument is given, the current retry count is displayed.

DIT

This command disables interrupts before the next command is executed.

DMA

This command enables DMA transfers via the DSC's DMA logic.

EIT

This command enables interrupts before the next command is executed.

IMMEDIATE

This command cancels AUTOMODE, and allows you to enter commands directly.

LOOP {*n*}

This command causes loopable commands to repeat the specified number of times (0-7FFF hex). A value of 0 disables looping.

NDMA

This command disables DMA transfers. Non-DMA data transfers are valid only in STEP 1 mode.

NOPROTECT

This command removes write protection from the device during testing, thus allowing write operations. (See PROTECT.)

NOREPEAT

This command turns off the repeat function. (See REPEAT.)

PARAMETER

This command displays the current program parameters and control selections.

PROTECT

This command write protects the device during testing. (See NOPROTECT.)

PROTOCOL {*args*}

This command specifies the mode of operation for SCSI bus commands. The various settings given by the arguments affect which select command is issued, the value loaded into the SBIC source ID register, and the value loaded into the SBIC timeout period register. The arguments are:

a	Raise ATN on selections to cause a message out phase
-a	Do not raise ATN
d	Allow disconnection for SCSI operations
-d	Do not allow disconnects
t	Force the hardware timeout for selections
-t	Disable hardware timeout
h	Display the help message
z	Zero all a, d, and t functions.

This command does not function in a Build loop.

REPEAT

This command allows certain commands to be repeated. After REPEAT has been issued, following completion of a command you are prompted:

Again? (y/n)

Enter "y" to execute the command again, or "n" to exit the command.

RETRY {*d*}

This command specifies the retry count (0-255 decimal). If an error occurs while executing a command, the command will repeat until either the retry count is exhausted or the command executes successfully.

STEP (0|1)

This command causes the driver to use the SBIC step mode, if the argument is 1. When the argument is 0, combination mode is used. Combination mode is valid only when DMA data transfers are selected (see the DMA command).

7.4.8.3.5 Functional Unit Selection Commands

ADDRESS {*n*}

This command selects the address of the board to test. There is no restriction on the value specified, and may select a controller other than a DSC, such as a WDC. Some commands will work on these controllers as well. If no argument is given, the address of the current controller is displayed.

DSC {*d*}

This command selects the DSC to test (d=1-3). If no argument is given, the number of the currently selected DSC is displayed.

LPORT {*n*}

This command selects the logical port (0-F hex), and so performs the same function as the DSC and PORT commands together. The bits are as follows:

Bit	Meaning
0	0 = port a
	1 = port b
1,2	DSC boards 0-3
3	0 = main unit
	1 = expansion unit

PORT {*args*}

This command selects which port(s) to test on the DSC. The ports are designated as a and b, and both may be specified:

port a b

If no argument is specified, the currently selected ports are displayed.

TARGET {*n*}

This command sets the destination ID register in the SBIC. If no argument is given, the current destination ID is displayed.

UNIT {*n*}

This command selects the target logical unit (0-7). If no argument is given, the current logical unit is displayed. Most targets support only one unit and expect the unit number to be 0, the default.

XDSC {*d*}

This command selects the expansion unit DSC to test (d = 1-3). If no argument is given, the number of the currently selected DSC is displayed. If there is no expansion unit, the currently selected main unit DSC is displayed.

7.4.8.3.6 Logical Block Number Selection Commands

BLOCK {*d*}

This command selects the starting block for the next transfer. The block is specified in decimal, but may be specified in hexadecimal by preceding the number with "h".

#BLOCKS {*n*}

This command specifies the number of blocks to transfer. The block size is used to calculate the actual byte count for the transfer.

+BLOCK

This command increments the current block number by 1. When the maximum block number is exceeded, the block number will wrap around to the minimum block number. The new block number can then be used by the next command.

-BLOCK

This command decrements the current block number by 1. When the minimum block number is exceeded, the block number will wrap around to the maximum block number. The new block number can then be used by the next command.

GRBLOCK

This command generates a random block number within the range specified by MINBLOCK and MAXBLOCK. The block number can then be used by the next SCSI command.

MAXBLOCK {*n*}

This command specifies the maximum block for a range of blocks. The GRBLOCK and BLOCK commands use this value to determine the blocks to use in data transfers. (See the #BLOCKS command to specify the size of the range.)

MINBLOCK {*n*}

This command specifies the minimum block for a range of blocks. The GRBLOCK and BLOCK commands use this value to determine the blocks to use in data transfers.

7.4.8.3.7 Buffer Definition and Control Commands

BUFFREAD {*n*}

This command selects or displays the starting address for the read buffer. All following read commands write data to this location. If no argument is given, the current start address is displayed.

BUFFSIZE {*n*}

This command specifies or displays the size of both the read and write buffers. When the buffer size is changed, the starting address of the read buffer is adjusted to immediately follow the write buffer. To override this arrangement of buffers, first specify the buffer size, and then specify the start address for the read buffer.

BUFFWRIT {*n*}

This command selects or displays the starting address for the write buffer. All following write commands write data to this location. If no argument is given, the current start address is displayed.

COMPARE {*n*}

This command compares the write and read buffers following a SCSI operation. If no argument is given, the entire buffers are compared. If an argument, only that many bytes are compared, up to the buffer size.

CREATE {*n*}

This command selects a data pattern from the patterns listed in Table 7-7. The optional argument is in the range 1-15 hex. If no argument is given, the next pattern is selected.

INCBUFF {*n*}

This command increments the starting address of the read and write buffers. If an argument is given (in hexadecimal), that value is added to the start address of both buffers. If no argument is given, the value of BUFFSIZE is added to the start addresses.

PATTERN {*n*}

If an argument is given (1-15 hex), this command selects a data pattern from the patterns listed in Table 7-7, as in the CREATE command. If no argument is given, the available data patterns are shown, followed by the current data pattern.

RDISPLAY {*n*}

This command displays the data in the read buffer. If an argument is given (in hex), this number of bytes is displayed. If no argument is given, a page of 256 bytes is displayed. The display includes the address of the data and the ASCII interpretation of the data.

SWAPBUFF

This command swaps the read and write buffers by exchanging the start addresses.

WDISPLAY {*n*}

This command displays the data in the write buffer. If an argument is given (in hex), this number of bytes is displayed. If no argument is given, a page of 256 bytes is displayed. The display includes the address of the data and the ASCII interpretation of the data.

ZAPREADB {*n*}

This command fills the read buffer with a known data pattern. This is useful to assure that the next compare command is not comparing data from the previous read command. If no argument is given, the buffer is filled with the pattern 0BAD hex.

ZAPWRITE {*n*}

This command fills the write buffer with a known data pattern. If no argument is given, the buffer is filled with the pattern 0BAD hex.

7.4.8.3.8 Miscellaneous Commands

DELAY {*d*}

This command causes a delay for the number of milliseconds specified (0-255 decimal). The timing is accurate to 16 microseconds.

INIT

This command initializes the program and resets the default control parameters. The system is also resized. No display is shown.

KEYS

This command loads the terminal function keys with a set of commands defined for DSCFS. As the keys are programmed, the functions programmed into them are displayed.

MBUILD {*name*}

This command begins a build loop for a DSC macro. Each command entered into the macro must be syntactically correct since no checking is performed by the program. When you have entered the last command, simply press RETURN. DSC macros cannot be executed from a build loop, a command loop or another macro. To display a macro, enter the macro name followed by "d".

A name for the macro, up to 8 letters, may be specified on the command line, or you will be prompted for it. The name must be different from any other DSCFS command.

RESET

This command issues a reset to the entire system. Following the reset, the system console device is reprogrammed.

SIZE

This command causes the system to execute the sizing operation, as it did when the diagnostic was loaded. The results are shown on the screen.

STRUCTURE

This command displays the internal structure of the DSCFS program for the active DSC and port. This information is not useful for diagnostic purposes.

7.4.8.3.9 General Purpose Commands

CDATA

This command compares the byte set by SETEXP with the last byte read by EREAD.

CSCREEN

This command clears the VDT screen.

CSTAT

This command specifies the required status for a device and the address to compare it against. This information is used by WPORT and RPORT before data transfers to verify correct device status.

DB *n*

This command displays a byte of data from the specified memory address, and pauses for you to modify the data. To leave the data unchanged, press RETURN. To display the previous location, enter the slash character, "/". That byte is then displayed, and the program pauses to allow modification.

DEC *d*

This command displays the hexadecimal equivalent of the decimal argument.

DISPLAY

This command displays the byte last read by the EREAD command.

DL *n*

This command displays a long word (4 bytes) of data from the specified memory address, and pauses for you to modify the data. To leave the data unchanged, press RETURN. To display the previous location, enter the slash character, "/". That long word is then displayed, and the program pauses to allow modification.

DUMP {*n*}

This command displays the contents of memory starting at the specified address. Each execution displays one page (256 bytes) of data and interpreted in ASCII, and displays the address.

DW *n*

This command displays a word (2 bytes) of data from the specified memory address, and pauses for you to modify the data. To leave the data unchanged, press RETURN. To display the previous location, enter the slash character, "/". That word is then displayed, and the program pauses to allow modification.

EREAD {*n*}

This command reads data from the current device address, which can be displayed using the ADDRESS command. If no argument is specified, the currently selected controller and port address is used. If an argument is given, it specifies the offset for the controller and port address. The data written is specified by SETEXP. This command can be looped.

EWRITE {*n*}

This command writes data to the current device address, which can be displayed using the ADDRESS command. If no argument is specified, the currently selected controller and port address is used. If an argument is given, it specifies the offset for the controller and port address. The data written is specified by SETEXP. This command can be looped.

HEX *n*

This command displays the decimal equivalent of the hexadecimal argument.

REVISION

This command displays the revision level of the DSCFS program.

RPOINT {*n*}

This command specifies the starting address of a buffer where data which will be read should be placed. This address must be beyond the program and EXEC.

RPORT {*n*}

This command transfers a block of data from the specified address and places it in the buffer specified by RPOINT. The amount of data to transfer is specified with LOOP. The argument specifies the offset for the device base address.

SETEXP

This command sets the expected byte for the next EWRITE command or for the next comparison with CDATA.

WPOINT {*n*}

This command specifies the starting address of a buffer where data is transferred to or written from. This address must be beyond the program and EXEC.

WPORT {*n*}

This command transfers a block of data from the specified address and places it in the buffer specified by WPOINT. The amount of data to transfer is specified with LOOP. The argument specifies the offset for the device base address.

7.4.8.3.10 SCSI Commands

CAPACITY (0|1)

This command issues the SCSI Read Capacity command (SCSI code 25), which reads and displays the device capacity. The optional argument specifies the PMI (partial media) bit. Setting the PMI to 0 returns the last addressable logical block and the block length in bytes. Setting the PMI to 1 returns the actual last logical block and the block length.

DCB {*n*}

This command defines an optional Device Control Block (DCB). This is of little use for diagnostic purposes.

FREE

This command issues the SCSI Release command (SCSI code 17), releasing the device from being reserved.

GLIST

This command issues the SCSI Display Defect Data command, to read and display the drive's Glist (SCSI code 37).

INQUIRE

This command issues the SCSI Inquire command (SCSI 12). The data returned in placed in the Read buffer and displayed.

LREAD

This command issues a SCSI Long Read command (SCSI code 28), and places the data in the read buffer. This command can be looped. 1 to 65535 blocks (decimal) can be read by this command.

LSEEK

This command seeks to the block specified by the last BLOCK command. It has a greater range than the SEEK command.

LWRITE

This command performs a SCSI Long Write command (SCSI code 2A), writing data from the write buffer to the peripheral. This command can be looped, and can be set to execute over a range of blocks. The transfer can be from 1 to 65535 blocks (decimal).

MASTER

This command loads the source ID register with the last ID selected by OWNID, and then issues a SCSI reset command. The command checks for correct SBIC and DSC status after the reset.

PGLIST

This command issues the SCSI Display Defect Data command, to read both the Plist and the Glist and display the combined data (SCSI code 37).

PLIST

This command issues the SCSI Display Defect Data command, to read and display the drive's Plist (SCSI code 37).

RCOMPARE

This command reads the SCSI peripheral and, if the command succeeds, compares the data in the write buffer with the data in the read buffer.

READ

This command issues a SCSI Read command (SCSI code 08), and places the data in the read buffer. This command can be looped. 1 to 255 blocks can be read by this command (see LREAD for larger reads).

RECALIBR

This command issues the SCSI Recalibrate command (SCSI code 01). The device recalibrates the heads to cylinder 0 and returns the block value to 0. .

REMOVE

This command issues the SCSI Reassign Logical Block command (SCSI code 07). The logical block specified by BLOCK is reassigned to one of the spare sectors deallocated for this purpose. The block is added to the defect growth list for use with a later reformatting operation.

RESERVE

This command issues the SCSI Reserve command (SCSI code 16) for the current device. Another device can be reserved by modifying the OWNID value. If successful, the device is reserved until you issue a FREE command.

RRAM

This command issues the SCSI Read Buffer command (SCSI code 3C), causing the data in the device's buffer to be transferred to the read buffer. The header data is zeroed out so the data can be compared.

SEEK {*n*}

This command issues the SCSI Seek command (SCSI code 0B) to the specified cylinder, or to the last block selected if no cylinder is specified. The command cannot seek to the preserved diagnostic cylinders.

SELECT *d*

This command issues the SBIC chip Select command to establish a connection between an initiator and a target. The parameter identifies the target device (0-7). If no target responds, an error is reported.

SENDDIAG

This command issues the SCSI Send Diagnostic command (SCSI code 1D), with the self-test bit set to 1.

SENSE

This command issues and interprets the SCSI Sense command to display the data bytes received (SCSI code 03).

SLAVE *d* {F}

This command begins target emulation. A pseudo superblock is written to memory, and then used to emulate a SCSI disk device. The data used to create the pseudo superblock is taken from the last DEFINE command, which should be run before SLAVE. The first argument specifies from 1 to 8 ID's (0-7), one of which must be the initiator, usually number 7. The optional F parameter specifies that the emulation is to continue "forever"; otherwise, emulation ends after the next command. Interrupts should be enabled for this command.

START {0|1}

This command issues the SCSI Start command (SCSI code 1B) to start the drive motor and take it on line. The optional argument specifies the load bit, and defaults to 0. Some devices reject the load/eject bit, and may post an error.

STOP {0|1}

This command issues the SCSI Stop command (SCSI code 1B) to stop the drive motor and take it off line. The optional argument specifies the eject bit, and defaults to 0.

USTATUS

This command issues the SCSI Test Unit Ready command (SCSI code 00). If no error is displayed, the unit is ready.

VERIFY

This command issues the SCSI Verify command (SCSI code 2F). The data at the specified blocks is verified for correct ECC, and can be run on a range of blocks. This command can be looped.

WRAM

This command issues the SCSI Write Buffer command (SCSI code 3B), causing the data in the write buffer to be transferred to the device's buffer. The data is shifted by 4 bytes to allow for the required header.

WRITE

This command performs a SCSI Write command (SCSI code 0A), writing data from the write buffer to the peripheral. This command can be looped, and can be set to execute over a range of blocks. The transfer can be from 1 to 255 blocks (decimal).

7.4.8.3.11 Range Commands

DISK

This command selects the range of blocks from the first block to the maximum addressable block for the next disk operation. Use the RANGE command to select a smaller range.

DREAD

This command performs multiple reads of the entire disk. Any errors detected are stored in a table and displayed at the end of the operation. The superblock is read to determine the size of the disk. The entire disk is tested, except for the diagnostic cylinder.

DVERIFY

This command is similar to DREAD, except that the SCSI Verify command is used to force an ECC check on every block and no data is transferred. All errors detected are stored in a table and displayed at the end of the operation.

RANGE *start stop* {*n*}

This command specifies a range of blocks for the next disk operation. The start and stop parameters specify the inclusive range for disk access. The optional parameter specifies the block size for transfers (1-FF hex). Use the LPORT command to specify the maximum block number that can be selected by RANGE. Use the DISK command to select the entire disk.

TABLE

This command displays the last error log table built by a RCOMPARE, DREAD or DVERIFY command.

7.4.8.3.12 Unit Definition Commands

DEFINE *cylinders heads sectors bytes/sect*

This command defines a superblock for the target for disk drive emulation by the SLAVE command. (LPORT must be used for an actual disk drive.) If no parameter is specified, the current parameters are displayed.

Table 7-7. Test Data Patterns

Type	Description	Data Words
0	All zeros	0000 0000 0000 0000 0000
1	All ones	FFFF FFFF FFFF FFFF FFFF
2	Floating 0	FFFE FFFD FFFB FFF7 FFEF
3	Floating 1	0001 0002 0004 0008 0010
4	Random	F946 EA89 12DD 32A0 00FF
5	All fives	5555 5555 5555 5555 5555
6	Di-bit	CCCC CCCC CCCC CCCC CCCC
7	Tri-bit	E38E 38E3 8E38 E38E 38E3
8	Quad-bit	F0F0 F0F0 F0F0 F0F0 F0F0
9	Zero di-bit	6DB6 DB6D B6DB 6DB6 DB6D
A	Incrementing	0001 0002 0003 0004 0005
B	Zero quad-bit	7777 7777 7777 7777 7777
C	Zero quad-bit	7BDE F7BD EF7B DEF7 BDEF
D	One di-zero	9249 2492 4924 9249 2492
E	One tri-zero	8888 8888 8888 8888 8888
F	One quad-zero	8421 0842 1084 2108 4210
10	Increase freq	FF00 FE03 F03E 0F0E 32AA
11	Decrease freq	FEAA 6663 8E38 7878 783F
12	Jitter	9659 6596 5965 9659 6596
13	Worst Case 1	D936 4DB6 4ED9 364D B64E
14	Worst Case 2	6DB6 DB0F 6DB6 DB0F 6DB6
15	Worst Case 3	DADA CA58 C2FE DADA CA58

SECTION 8
MICRO DIAGNOSTIC SYSTEM

8.1 OVERVIEW

The Micro Diagnostic System (MDS) is an extensive set of PROM based programs that perform several related system startup and testing tasks on MAI 2500/3000/4000 systems. MDS contains, for instance, the initial program counter and stack pointer for initial CPU operation, and the first instructions executed when the system is first powered on. It performs system pre-tests and provides the system boot routines. It also provides several low level diagnostic tools and programs. This section focuses on the system diagnostic functionality of MDS. MDS also includes some software debugging tools, but these are not covered in this manual.

The MDS functions covered in this section are as follows:

- System Pre-tests
- NVRAM Configuration (<conf>)
- Error Detection and Correction (<edc>)
- Memory Management Test (<mem>)
- Memory Management Unit Test (<mmu>)
- Central Microprocessor Board Test (<cmb>)
- Winchester Disk Controller Test (<wdc>)
- Instruction/Data Cache Test (<cache>)
- Controllers Test (<ctlr>)

8.2 MDS OPERATION

When the system is first powered on, or when the reset switch is activated, the hardware fetches the initial programs counter (PC) and stack pointer from PROM, which begins execution of the In general, system pre-tests.

8.2.1 System Pre-tests

The pre-tests ensure that the processor can execute code, can access and use the NVRAM and the PROM set. The tests are necessary for the rest of MDS to work and to provide reliable operation. For instance, if the NVRAM fails, all other MDS functions will also fail because the stack and variables are located there.

These tests include a minimal 68020 instruction test, a PROM checksum test, NVRAM test and a SCC internal loopback test. All of main memory is cleared, and address and data tests are performed on the first 4 KB only. The LED indicates which pre-test is currently running, as shown in Table 8-1.

Table 8-1. Pre-test Status Displays

HEX DISPLAY	TEST BEING EXECUTED
01	The instructions test is being run with the 68020 instruction cache enabled.
02	The PROM checksum test is running.
03	The NVRAM test is running (non-destructive).
04	Sizing and initialization of main memory is being performed.
05	The CMB SCC internal loopback test is running.
06	The first 4 KB of memory is being tested.
07	The sense switches are being read.
08	The vector table is being initialized in RAM. The Vector Base Register (VBR) is then set to location zero.
09	The serial ports are being programmed according to the NVRAM configuration parameters.

Any fatal errors that occur during the pre-tests cause MDS to halt and display a blinking error code on the LED. A system reset is required to repeat the pre-tests, if desired. Fatal errors include failing of the instruction test, PROM checksum, NVRAM functional test, or processor detected errors such as unexpected address or bus errors that occur before a console is ready to report the error. In some cases, SCC internal loopback and memory test failures may also cause a fatal error.

Error codes are displayed on the LED as a FLASHING CODE. The codes are explained in Table 8-2.

Table 8-2. Pre-test LED Error Codes

HEX DISPLAY	ERROR (WHEN FLASHING)
01	The 68020 instructions test failed.
02	The PROM checksum test failed.
03	The NVRAM test failed.
04	A DMA serial controller is configured in NVRAM, and there is no memory. Make memory available to the system or remove the serial controllers to get the console on SC0 to further diagnose the problem.
05	The SCC loopback test failed, and the NVRAM specifies the SCC as the consoledevice. Either correct the problem with the SCC, or strap the sense switch for the <conf> module with a 4-way or 8-way board zero connected to further diagnose the problem.
06	The first 4 KB of memory has failed while a DMA serial controller is configured in NVRAM. Either correct the memory problem or remove the 4-ways and 8-ways to get a console on SC0 to further diagnose the problem.
07	The sense switches are strapped for a specific test within a test module, and the test requires the first 4KB of memory to exist and pass the pre-tests. Either make memory available to the system, fix the memory problem, or change the sense switch settings.
XX	Any other flashing value indicates the vector number of an unexpected interrupt received before the system console is ready to display the problem. For example, a flashing 08 indicates a bus error.

8.2.2 System Sense Switches

Following successful completion of the pre-tests, MDS checks the system sense switches to determine its next action. Usually the switches are set to zero, indicating a normal self-test, initialization and boot procedure, but this may be changed for special diagnostic purposes. In customer settings, the sense switch should be set to zero so the normal self-test and boot cycle executes.

The sense switches are located on the CMB (location 11H), close to the power supply. Switches s1 through s8 are read by MDS following the pre-tests. When the switches are set to a value other than 0, the normal self-test and boot routine is omitted, and action is taken as shown in Table 8-3. Switches 6 through 8 identify the MDS module to begin, and switches 1 through 5 identify the program to execute.

Table 8-3. Sense Switch Settings

SWITCH SETTINGS 8 7 6 5 4 3 2 1	HEX	PURPOSE
0 0 0 0 0 0 0 0	00	Normal self-test, init and boot
0 0 0 0 0 0 0 1	01	Reserved
0 0 0 0 0 0 1 0	02	Enter MDS keyboard control
0 0 0 0 0 0 1 1	03	Alternate boot routine
0 0 0 0 0 1 0 0	04	Enter <conf>, force NVRAM defaults
0 0 0 0 0 1 0 1	05	Reserved
0 0 0		
0 0 0 1 1 1 1 1	1F	Reserved
0 0 1 0 0 0 0 0	20	Pass control to EDC test
0 0 1 0 0 0 0 1	21	Execute EDC test 1
0 0 1		
0 0 1 1 1 1 1 1	3F	Execute EDC test 1F
0 1 0 0 0 0 0 0	40	Pass control to MEM test
0 1 0 0 0 0 0 1	41	Execute MEM test 1
0 1 0		
0 1 0 1 1 1 1 1	5F	Execute MEM test 1F
0 1 1 0 0 0 0 0	60	Pass control to MMU test
0 1 1 0 0 0 0 1	61	Execute MMU test 1
0 1 1		
0 1 1 1 1 1 1 1	7F	Execute MMU test 1F
1 0 0 0 0 0 0 0	80	Pass control to CMB test
1 0 0 0 0 0 0 1	81	Execute CMB test 1
1 0 0		
1 0 0 1 1 1 1 1	9F	Execute CMB test 1F
1 0 1 0 0 0 0 0	A0	Pass control to WDC test
1 0 1 0 0 0 0 1	A1	Execute WDC test 1
1 0 1		
1 0 1 1 1 1 1 1	BF	Execute WDC test 1F
1 1 0 0 0 0 0 0	C0	Pass control to CACHE test
1 1 0 0 0 0 0 1	C1	Execute CACHE test 1
1 1 0		
1 1 0 1 1 1 1 1	DF	Execute CACHE test 1F
1 1 1 0 0 0 0 0	C0	Pass control to CTLR test
1 1 1 0 0 0 0 1	C1	Execute CTLR test 1
1 1 1		
1 1 1 1 1 1 1 1	DF	Execute CTLR test 1F

8.2.3 Self-Tests

With the sense switches in default position (set to zero), the system goes through the normal self-test and boot routine following the pre-test. MDS carries out the tests using the system console as the output device, indicating current status and displaying error messages if failures occur.

Before the tests are executed, the system I.D., MDS revision level and the system serial number are displayed on the console. The revision level displayed is the level of all MDS modules combined, not the level of a single module within MDS.

As the tests are executing, MDS displays the current test module being executed with the message:

Now testing xxxx

This message is followed by dots indicting the number of tests which have been completed. Not all tests in a test module are executed during the self-test.

During the self-test, the most significant digit of the LED displays indicates the test module currently executing. The least significant digit normally displays 0, and changes only if an error occurs. The LED always reflects the last error (if any) that occurred during the test sequence. The code for the test modules (most significant digit) are shown in Table 8-4. Refer to the descriptions of the test modules for the interpretation of the least significant digit. The value of X in the display indicates the error, and may be a value 1 to F.

Table 8-4. Self-test LED Codes

HEX Display	Test Executing
0X	Pre-test status (an error only if flashing)
1X	Error Detection and Correction (EDC) tests
2X	Memory tests
3X	Memory Management Unit tests
4X	CMB tests
5X	Winchester Controller tests
6X	Cache tests
7X	Controller test (4-way, 8-way, LAN, CS, TS)

The following tests are included in the Self-Test:

- EDC, tests 1 through E
- MEM, tests C-E, 15-17
- MMU, all tests
- WDC, tests 4, 6, 9, B, C, 10, 11, 13, 14, 16
- CACHE, none
- CTRL, all tests

8.2.4 Shutdown and Boot Prompt

Following the self-tests, MDS displays the system boot menu:

1) Boot
2) Alt-load
3) Micro-diagnostics
4) Power down

Enter option (1-4):

The options respectively do the following:

1) Perform the boot procedure

2) Begin the alternate boot procedure

3) Enter MDS command mode

4) Power down the system

Options 1, 2 and 4 are described in the MAI 3000/4000 user documentation. The remainder of this section addresses only operation in MDS command mode, including running tests and executing commands.

To select MDS command mode, type **3** and press **ENTER**. The <mds> prompt is then displayed.

8.2.5 System Console Control

MDS supports the system console on any 4-way, 8-way or CMB SCC port, as specified in the NVRAM configuration. In a few specific cases, MDS will switch the console to another port.

- If NVRAM is configured with the console on a 4-way or 8-way port other than board 0 and port 0, and that board fails or doesn't exist, MDS attempts to switch to port 0 of board 0. Failing that it switches to SCC port 0 on the CMB.

- If NVRAM is configured with the console on a 4-way or 8-way board 0 port 0, and that board fails the self-test, MDS switches to SCC port 0 on the CMB.

- If a port is configured on a 4-way or 8-way, and the memory test fails, a 06 flashes on the LED, indicating that the controller works but the required memory has failed. Remove the 4-way or 8-way and reset the system to force MDS to switch to SCC port 0 on the CMB.

- If NVRAM is configured with the console on SCC port 0 on the CMB, and that port fails self-test, a 05 flashes on the LED. MDS does not attempt to switch ports.

Main memory is not generally required by MDS to run, except in the case where the system console is configured on a 4-way or 8-way. A few individual tests do required main memory, however, as stated in the test descriptions later in this section.

8.2.6 Control Sequences

MDS uses very few control sequences to control its operations:

- If X-on/X-off flow control is configured in NVRAM, you can use **CTRL + S** to stop the display and **CTRL + Q** to begin the display again.
- **ESCAPE** is generally used to escape out of programs or prompts.
- **CTRL + U** is used by the **HOST** command (described later).
- **CTRL + C** is used by the **TRAC** command (described later).

8.3 MDS TEST MODULE CONTROL

MDS tests are selected and run by entering commands at a MDS prompt. Additional commands select and deselect run options, controlling several aspects of test execution.

When MDS is first started, the prompt is <mds>. When a test is selected, the prompt is changed indicating the current test. For example, when the EDC test is selected the prompt becomes <edc>.

Tests in the selected test module are started with either the ALL or RUN commands. ALL runs all tests is the test module. RUN runs the only the selected tests within the test module.

Predetermined sets of tests from all modules are started with the SELF and BURN commands. Which specific tests are included from each module are described in the sections describing the module.

Tests are run in an order such that the most basic operations are tested first, such as addressing registers, building up to more complex functions. This order applies to both the ordering of the modules as well as to the ordering of tests within a module.

Certain tests require manual intervention, such as to confirm that an external loopback connector has been installed. These tests are only run if the MAN command has been run first.

8.3.1 Help Commands

HELP - List MDS Commands

This command displays all MDS commands which are available regardless of which test prompt is displayed. This command does not list commands specific to a test module (see LISC).

LISC - List Test Module Commands

This command lists the commands available as part of the current test module. These commands are not available when the module is changed.

LISP - List Program Modules

This command lists the name, revision level and base address of all program and test modules in MDS. The module names displayed can be entered as commands to invoke the module.

LIST - List Tests Within a Test Module

This command lists the tests available within the current test module. The number of each test is displayed along with a description of the test. The number is used to specify the test before executing the RUN command. If a test description ends with an asterisk, it is a manual intervention test, and so can only be run after the MAN command has been executed.

8.3.2 Module Selection Commands

ALT - Invoke Alternate Load Routine

This command causes the alternate load procedure to begin. You are prompted for the boot device (wc, cs, ts, fw or sc, followed by the two digit unit number) and the system file. This procedure is fully described in the system user documentation.

CACH - Invoke the <cache> Test Module

This command invokes the CACHE test module and executes the test initialization routine. Refer to Section 8.11 for a description of this module.

CMB - Invoke the <cmb> Test Module

This command invokes the CMB test module and executes the test initialization routine. Refer to Section 8.9 for a description of this module.

CONF - Invoke the <conf> Program Module

This command displays the current NVRAM configuration on the screen. Following the display, you can enter any of the configure command, as described in Section 8.5. Executing the CONF command does not change the configuration record in NVRAM (unlike selecting conf by sense switch settings, which restores default settings).

CTRL - Invoke the <ctrl> Test Module

This command invokes the CTRLB test module and executes the test initialization routine. Refer to Section 8.12 for a description of this module. The system is sized for controllers and the results are displayed on the console screen.

EDC - Invoke the <edc> Test Module

This command invokes the EDC test module and executes the test initialization routine. Refer to Section 8.6 for a description of this module.

MDS - Invoke the <mds> Program Module

This command causes a warm start of MDS and displays the <mds> prompt. This command is required before running MDS tests if you exit to MDS from any other program. Failure to do so may cause address or bus errors, due to the location of the stack and vectors in memory.

MEM - Invoke the <mem> Test Module

This command invokes the MEM test module and executes the test initialization routine. Refer to Section 8.7 for a description of this module. The amount of memory in the system is displayed on the console screen.

MMU - Invoke the <mmu> Test Module

This command invokes the MMU test module and executes the test initialization routine. Refer to Section 8.8 for a description of this module.

WDC - Invoke the <wdc> Test Module

This command invokes the WDC test module and executes the test initialization routine. Refer to Section 8.10 for a description of this module. The system is sized for disk controllers and the results are displayed on the system console.

8.3.3 Test Selection Commands

ALL - Execute All Tests in the Current Test Module

This command executes all of the tests contained in the currently selected test module. The loop forever option (LP) is turned off, and the test default option (DEF) is turned on. Manual intervention test are run only if the MAN command has been executed.

BURN - Execute the System Burnin

This command is used for extended run testing, running all tests in all modules, except for the manual intervention tests. All tests control options except MAN are in effect for burnin testing. The tests run in a continuous loop until the user presses ESCAPE. When testing is interrupted, the error log is displayed (see LOG command), showing the number of passes run and any errors detected.

RUN - Execute a Range of Tests

This command runs either a single test or a range of tests in the currently selected test module. MDS prompts you for the number of the first test to run (see the LIST command). MDS then prompts you for the number of the last test to run. If you enter a higher test number, all tests in the range are executed. If you press ENTER alone, only the first test is run.

SELF - Execute the System Self-test

This command executes the normal power on self-test cycle. The system will automatically boot after completion of the self-tests, just like the power on sequence. The specific tests run from each module are listed in the descriptions of the test modules.

8.3.4 Test Option Commands

The following MDS commands select options during the running of tests. These options are selected by entering the command name, and deselected by entering a minus sign ("-") in front of the command.

The only general default option is ERR, to display errors on the console. When the self-test is run, either by command or by switch setting, the ERR and BEL options are defaults. When a specific test is selected by switch settings, the LPE, LP and MAN options are defaults.

BEL/-BEL - Enable/Disable Bell on Error

With the bell enabled, the bell character is output before and after each error message is displayed on the console. With the bell disabled, the bell character is suppressed.

DEF/-DEF - Enable/Disable Test Defaults

Default parameter values are defined for each test and scope loop. With defaults enabled, these default values are used. With defaults disabled, the user can specify parameter values for some, but not all, scope loops and tests. With defaults disabled, each time a test or scope loop runs which allows the user to change the parameters, the command will prompt the user for the necessary data to override the default.

ERR/-ERR - Enable/Disable Error Display

With error display enabled, each error is displayed on the console screen. With error display disabled, the errors are not displayed. Disabling the error display is useful for a tight scope loop, since the display takes a considerable amount of time. Error codes continue to be displayed on the LED, test point 2 (TP2) is strobed, and the error tables are updated even with the display disabled.

ILOG - Initialize the Error Log

This command initializes the error tables used by the LOG command. After issuing the ILOG command, the LOG command will display zero values for each module.

LP/-LP - Enable/Disable Looping Forever

This command enables and disables the continuous loop feature. A continuous loop is useful particularly for scope loop tests. With looping enabled, each test repeats until ESCAPE is pressed. At the top of each loop, test point 1 (TP1) is strobed. When ESCAPE is pressed, the next test in the sequence is executed, looping until it is interrupted. This repeats until the last test has been interrupted, when the testing ends.

With looping disabled, scope loops and other tests are executed sequentially, and ESCAPE terminates the entire test sequence.

LPE/-LPE - Enable/Disable Loop on Error

With loop on error enabled, the test begins looping forever upon occurrence of an error. To exit the loop, press ESCAPE twice, or reset the system. With loop on error disabled, the test does not repeat when an error occurs.

LOG - Display the Error Log

This command displays the number of errors in each module and the error number of the first error that occurred in each module. For example:

Mod	Error Count	Error #
edc	00	00
mem	00	00
mmu	00	00
cmb	01	42
wdc	00	00
cach	00	00
ctlr	02	71

In this example, the <ctlr> module reported two errors, the first being an error 1 in module seven. The <cmb> module reported one error, an error 2 in module 4. If no errors occur in a module, the error count is zero and the error number is zero.

MAN/-MAN - Enable/Disable Manual Intervention Tests

This command enables and disables running of the manual intervention tests. With manual tests disabled, those tests requiring manual intervention are skipped by when tests are run by the ALL and RUN commands.

OPS - Display Current Options

This command displays the test options that are currently enabled.

PAU/-PAU - Enable/Disable Pause Before Loop

With pause enabled, the test pauses before executing a loop. This allows the user to set up test equipment prior to executing the loop. The loop is executed when the user presses ENTER. With pause disabled, testing does not pause before executing the loop.

STP/-STP - Enable/Disable Stop on Error

With stop enabled, MDS pauses upon detection and display of an error. The user may respond as follows:

ENTER - Continue testing
ESCAPE - End testing, and return control to the user
. (Period) - Continue testing, but disable Stop

With stop disabled, MDS continues testing, possibly causing error messages to scroll off of the console screen.

8.3.5 Software Debugging Commands

The following commands are primarily intended for software debugging, but some are useful for more general purposes. For instance, BOOT begins the standard boot procedure, SHUT performs a system shutdown, and REG displays the contents of registers.

Command names are from 1 to 4 characters. Only enough characters to distinguish the command from others need to be entered, usually two. If MDS does not recognize the command, it displays the invalid command response:

```
What?
```

Some commands accept arguments, as explained in the command descriptions. The general format for the commands is:

CMD param param param

Command parameters are separated by a space, and the command line is terminated by pressing ENTER.

Simple expressions can be formed with the commands by using the "+" and "-" symbols for addition and subtraction. No spaces are allowed in the expressions. Register symbols (An, Dn, PC, SP) are allowed in expressions followed by "+" or "-" to represent the value of the register. For example:

```
DN A0+
```

displays the byte pointed to by register A0. Without the "+" or "-" the register name refers to the memory location. For example:

```
DB A0
```

displays the contents of memory location A0.

The program names listed by **LISP** can also be used to form expressions, though this is not generally useful for fault detection.

AN, DN - Display or Change Register Contents

Syntax:
- AN a*x* — where a*x* is the address register number (0-7)
- DN d*x* — where d*x* is the data register number (0-7)

This command displays the contents of the specified register. Register numbers are A0 through A7 for address registers, and D0 through D7 for data registers. After the contents are displayed, you can enter a new value by simply entering the hexadecimal data and pressing **ENTER**.

After displaying or entering data, the following characters can be used in place of ENTER:

SPACE	Display the next register
"/"	Display the previous register
"."	Display the memory location pointed to by the register

BOOT Boot From Default Boot Device

Syntax: BOOT

This command causes the system to boot to the default boot device, as defined by the NVRAM configuration.

BR Set or Display a Software Breakpoint

Syntax: BR *n addr{/cnt} addr addr addr*

where:

n = 0-3	for a maximum of 4 breakpoints
addr	the address to set the breakpoint
cnt	the number of times to execute the instruction at the location prior to breaking (optional)

This command sets software breakpoints at up to 4 addresses. If no addresses are specified, the current breakpoint addresses are displayed. If addresses are specified, then the breakpoints are set starting at the first breakpoint specified, and the existing breakpoint is cleared. An address of zero will clear the breakpoint.

When a breakpoint is set in main memory, BR saves the instruction and places a trap #15 instruction at the address. When the trap is executed, the original instruction is restored, the breakpoint is cleared, and the register values are displayed.

When a breakpoint is set in PROM, tracing is turned on. As each instruction is executed, the program is temporarily stopped to check for a software breakpoint in the PROM. The code that may be executed in RAM is also run in trace.

CC Display or Change the Condition Codes

Syntax: CC

This command displays the current condition codes that are contained in the current status register of processor. After the display, you can enter a new value for the condition codes. The next GO or STEP command loads the new condition codes into the status register.

CLR Clear All Breakpoints

Syntax: CLR

This command clears all software breakpoints currently set by the BR command.

DB, DW, DL — Display or Modify Memory

Syntax: D*x addr*

where:
x is B, W or L
addr is the address

This command displays the byte (DB), word (DW) or long word (DL) beginning at the specified address. Data is displayed in hexadecimal. After the data is displayed, you can modify it by typing the new data and pressing **ENTER**.

After displaying or entering data, the following characters can be used in place of ENTER:

SPACE	Display the next location
"/"	Display the previous location
"."	Display the memory location pointed to by the data (DL only)

DC — Display or Modify Characters in Memory

Syntax: DC *addr*

where *addr* is the address

This command displays the address argument followed by the four ASCII characters at that address (long word). Non-displayable characters are represented by a period ("."). Following the display, you can enter new characters by typing the characters followed **by** ENTER. Any characters, displayable or not, can be entered except **BACKSPACE**, **ESCAPE** and **ENTER**. These characters are used as follows:

- **BACKSPACE** corrects an entry
- **ESCAPE** after display terminates the command
- **ESCAPE** after entry displays the next location
- **ENTER** terminates entry

DCAC — Disable 68020 Instruction Cache

Syntax: DCAC

This command turns off the 68020 Instruction Cache. It remains off until the ECAC command is issued.

ECAC — Enable 68020 Instruction Cache

Syntax: ECAC

This command turns on the 68020 Instruction Cache. It remains on until the DCAC command is issued.

GO — Begin or Continue Execution

Syntax: GO {*addr*}

where *addr* is the address to be execution (optional)

If no address is given, this command begins execution at the current value of the PC register. If an address is specified, execution begins at that address by loading it into the PC register.

HOST — Download Port Communications

Syntax: HOST

This command causes the system console to be attached to the download port. Any character entered at the system console is sent to the host attached to the download port. Any character received from the download port is displayed on the system console.

To return control back to the normal console port, enter **CTRL + U**.

LOAD — Load Memory From the Download Port

Syntax: LOAD {*string*}

This command loads memory from the download port. The options string parameter is used to send a command to the download port prior to accepting characters from the port. If the parameter is not provided, there is no output to the port prior to the download. The port used for downloading is defined in the NVRAM configuration.

MOVE — Move a Block of Memory From One Location to Another

Syntax: Move *source destination count*

where:
source is the source address
destination is the destination address
count is the number of bytes to move

This command moves the specified number of bytes from one address in memory to another. Overlapping moves are permitted.

PC — Display or Change the Program Counter

Syntax: PC

This command displays the contents of the processor program counter. After the display you may change the contents by typing the new value and pressing ENTER. Press ENTER alone to leave the pointer unchanged.

REGS Display All Registers

Syntax: REGS

This command displays the contents of all processor registers. The contents are current as of the last breakpoint or interrupt into MDS.

REMO Display or Change the REMOTE Byte

Syntax: REMO {*type*}

This command is not supported for field use.

RESE Reset the System

Syntax: RESE

This command issues a reset instruction. All CMB components and attached controllers are reset. The reset is automatically followed by MDS initialization of components that need to be reprogrammed after a reset.

SEAR Search for a Pattern

Syntax: SEAR *start end pattern*

where:
start is the start address
end is the end address
pattern is the pattern to search for in memory

This command searches for the pattern in the range of memory specified by the start and end addresses. The search is for a byte, word, three bytes or long word, depending on the number of characters in the pattern. All addresses where the pattern is found are displayed on the system console.

SHUT Perform a System Shutdown

Syntax: SHUT

This command performs a system shutdown. It raises the interrupt level to seven, initializes the vector table, performs a reset instruction, re-programs the serial ports configured in NVRAM, and displays the shutdown menu.

SP Display or Change the Stack Pointer

Syntax: SP

This command displays the value of the stack pointer. After the value is displayed, you may enter a new value by typing the new value in hexadecimal and pressing ENTER. The next GO or STEP command will cause the new value to take effect. Press ENTER alone to leave the value unchanged.

STEP — Single Step a Program at the Current PC

Syntax: Step {*n*}

where *n* is the number of steps to take.

This command causes the specified number of steps to be executed in the current program. If no number is specified, a single step is executed. After the steps are performed, the contents of the processor registers is displayed.

TRAC — Enable/Disable Trace Mode

Syntax: TRAC {*param param param*}

where each *param* is one of the following:

ON	Turn tracing on
-ON	Turn tracing off
STOP	Causes program to stop after each instruction
-STOP	Ends stopping after each instruction
PC	Displays the PC register while the programs runs

In trace mode, register information is displayed giving a status of the program running. You also have the option of having the program stop after each step and displaying the contents of all registers. With STOP disabled (-STOP) you may use the PC parameter to cause display of the program counter following each instruction, without stopping the program. STOP overrides PC, causing display of all register information. TRACE -ON turns off trace entirely.

TRAP — Set or Clear the Conditional Trap 14 Flag

Syntax: TRAP

Each time the TRAP command executes, the conditional breakpoint control is toggled from ON to OFF or from OFF to ON. The status of the control flag is displayed each time the command is entered.

When trap mode is ON, a trap 14 instruction in a program calls the interrupt service routine in MDS, which stops the program and displays the register contents. A GO command resumes execution. When trap mode is OFF, the interrupt is still processed, but the program continues processing immediately.

TYPE — Type or Display Memory on the System Console

Syntax: TYPE *start* {*end*}

This command displays the range of memory specified by the start and end addresses. If the end address is not specified, the page of memory starting at the start address is displayed.

The address of the block of data is displayed on the left of the screen, followed by the hex data and then the interpreted (ASCII) data. Unprintable characters are represented by a period (".").

ZAP Fill Memory With a Pattern

Syntax: ZAP *start end* {*pattern*}

This command fills the range of memory specified by the start and end addresses with a data pattern. If the optional pattern is specified, this pattern is used, repeated until the entire range is filled. If the pattern is not specified, the range is zero filled.

8.4 <alt> BOOT AND ALTERNATE LOAD PROGRAM MODULE

This section describes the <alt> MDS program module, which performs the system boot procedures. It is responsible for booting loaders into main memory from WD, CS and TS controllers.

Instructions for performing default and alternate loads are given in the user and service documentation. In this section we describe the error messages that may occur during the <alt> routine.

csss-xxx

This message is displayed if a bad status word is received from the 1/4 streamer controller. "*xxxx*" is the status word received.

tscs-*xx-cc-ssssssss*

Where: *xx* is the ending status
cc is the command sent to the controller, which is either:
00 (test unit ready)
01 (rewind command)
08 (read command)
11 (space one filemark forward command)
ssssssss is four bytes of sense data

This message is displayed when an unexpected status is received from the 1/2 inch steamer tape during a load. If the check bit of the ending status is on, the request sense command is sent to the controller to get the four bytes of the sense data (*ssssssss*).

tsss-*ee*

This message is displayed when a handshake problem occurs between the 1/2 inch streamer controller and the MDS driver routines. The code *ee* is one of the following:

20	Timeout waiting for bus free status 00
21	Timeout waiting for the command phase status after select
22	Timeout waiting for status or message phase (stuck in cmd phase)
23	Skipped to the message phase over status phase
24	Timeout waiting for the message phase
25	The message byte received was not zero, as expected
26	A TS parity error occurred between the adapter board and the tape drive

wdss-*bb-ee*

Where: *bb* is the board number
ee is the error code

This message is displayed when a handshake problem occurs between the hard disk controller and the DMA driver routines. The error code *ee* is one of the following:

20	Timeout waiting for bus free status 00
21	Timeout waiting for the command phase status after select
22	Timeout waiting for status or message phase (stuck in cmd phase)
23	Skipped to the message phase over status phase
24	Timeout waiting for the message phase
25	The message byte received was not zero, as expected

wdcs-*bb-xx-cc-ssssssss*

Where: *bb* is the board number
xx is the ending status
cc is the command byte issued to the controller, which is either:
00 (test unit ready command)
08 (read command)

This message is displayed if there is an unexpected status received from the hard disk controller status register. If the check bit of the ending status is one, the request sense command is sent to the controller to get the four bytes of sense data (*ssssssss*).

Boot Device Not Found

This message is displayed if the boot device controller cannot be found during a boot.

Invalid Loader Format

This message is displayed under these conditions:

- A loader has been loaded successfully, but its header format is invalid
- An unexpected filemark or end of tape condition occurred while trying to load from 1/2 inch streamer tape. In this case, the tscs error is displayed before this message.
- During a diagnostics boot from disk, the diagnostics loader or its address in the superblock is invalid.

timeout err

This message is displayed during an attempt to load data from a serial port if it takes to long to received a data record.

8.5 <CONF> NVRAM CONFIGURATION PROGRAM MODULE

The CONF module configures the system console, boot, download port and printer port parameters in NVRAM. These parameters are used by MDS and diagnostic programs. The operating system uses only the system console parameters, and uses on-line utilities to configure other ports.

8.5.1 Running <conf>

Whenever MDS is in command mode, the **CONF** command can be used to invoke the configuration module. When CONF has been executed, the <conf> prompt is displayed, and the configuration commands are available. These commands can be listed by using the **LISC** command.

All changes made to the NVRAM configuration take effect the next time the system is powered on or reset.

8.5.2 NVRAM Defaults

Defaults are placed in the NVRAM in the following conditions:

- The sense switches are set for the <conf> module (see Table 8-3).
- The NVRAM checksum is incorrect, as in the case of an unprogrammed NVRAM chip.
- A 4-way or 8-way port is configured as one of the NVRAM options, but the board does not exist or has failed self-test.

The default system console is either port zero or 4-way/8-way board zero, or CMB SCC port 0, in that order of preference. Other defaults are as follow:

```
boot device     :  wd00      (Hard disk board 0 unit 0)

terminal type   :  evdt
console port    :  fw00 or ew00 or sc0
baud rate       :  9600
data bits       :  7
parity          :  odd
stop bits       :  1
flow control    :  xon

download port   :  sc1 (cmb serial port B)
baud rate       :  9600
data bits       :  7
parity          :  odd
stop bits       :  1
flow control    :  xon

printer port    :  pit
```

8.5.3 Serial Port Parameter Options

The console, boot device, download port and printer can all be configured on serial ports using the CONS, BOOT, DOWN and PRIN commands. When these are configured on serial ports, the baud rate, data bits, stop bits, parity and flow control parameters must be specified.

The program prompts you for these parameters if you specify the device as on a serial port. At each prompt, and lists the permitted values. Type the parameter value and press ENTER. Press ENTER alone to accept the default value. The parameters specified must be consistent with those set at the device itself.

The baud rate may be any of the following:

50	75	110	134.5	150	300	600	1200	1800	2400	3600
4800	7200	9600	19200							

The data bits are either 7 or 8.

The stop bits are either 1 or 2.

Parity is either odd, even, on or off. The off parameter disables parity (no parity bit transmitted). On parity is the same as odd parity.

The flow control options are none, xon, dtr, xon_dtr, modem, xon_modem, and b4printer. The b4printer parameter should only be used for the printer, and the printer has the Basic Four Printer Protocol.

8.5.4 <conf> Commands

DISP - Display Current Configuration

This command displays the current NVRAM configuration

TERM - Configure Terminal Type

This command configures the terminal type of the system console. The terminal types supported are: evdt, edt, vdt7270 and other.

CONS - Configure System Console Port

This command configures the port to be used as the system console. Diagnostic programs allow the console to be on any serial port, but the BOSS/IX operating system only allows SC0 (CMB serial port A) and port 0 on 4-way or 8-way board 0 (fw00 or ew00).

When CONS executes, it prompts you first for the console device (sc, fw or ew), and then for the unit number (usually 0). It then prompts you for the serial port configuration parameters listed in Section 8.5.3.

DOWN - Configure Download Port

The download port is used for booting, downloading from or communicating with a remote system using the **ALT**, **LOAD** and **HOST** MDS commands.

When DOWN executes, it prompts you first for the download device (sc or fw), and then for the unit number (usually 1). It then prompts you for the serial port configuration parameters listed in Section 8.5.3.

PRIN - Configure System Printer

This command configures the system printer port. These parameters are not used by the operating system, but are used for printed output by the diagnostic programs.

When PRIN executes, it prompts you first for the download device (sc, fw or pit). No further parameters are required for the parallel port (pit). If you select sc or fw, you are then asked for the unit number, and then for the serial port configuration parameters listed in Section 8.5.3.

BOOT - Configure Boot Device

This command configures the default boot device and parameters, which is used to boot the operating system during a normal load.

When BOOT executes, it prompts you first for the boot device (wd, ts, cs, sc or fw), and then for the unit number. The unit number can be any valid controller and port pair (usually 00). If the device is a serial port, you are prompted for the serial port parameters described in 8.5.3.

8.5.5 CONF Error Reporting

The following error messages may be reported by the <conf> module.

Nvram checksum error

This message is displayed if the configuration module fines that the checksum of the NVRAM configuration parameters is incorrect. If this message is displayed, the configuration parameters are corrupted and default parameters are automatically written to the NVRAM.

Invalid entry

This message is displayed whenever the user makes an invalid entry at one of the <conf> parameter prompts. The error causes the command to abort.

8.6 EDC TEST MODULE

The EDC Test checks the basic error detection and correction functions of the MAI 2500/3000/4000. If necessary, this test can be run without a VDT, using the LED display for output.

Refer to Section 8.2.2 for instructions on starting EDC by sense switch setting. To begin EDC from the <mds> prompt or another test prompt, enter:

edc

The test module then starts, and the <edc> prompt is displayed.

8.6.1 EDC Test Routines

8.6.1.1 TEST 1: EDC REGISTER ADDRESSING

This test determines if accessing an EDC related register causes a bus error. The registers checked are CBR, SYR EDC enable/disable control, and level 2 interrupt enable/disable. If a bus error occurs, the address of the register causing the bus error is displayed.

8.6.1.2 TEST 2: EDC CHECK WORD GENERATE - LONG WRITE

This test writes long word (4 byte) data patterns to a memory location with EDC enabled. Then the check word for the data is verified. Long word data pattens are used to generate all 128 possible check words.

8.6.1.3 TEST 3: EDC CHECK WORD GENERATE - WORD WRITE

This test writes word (2 byte) data patterns to a memory location with EDC enabled, using known data for the other half of the long word. Then the check word for the entire long word is verified. The test pattern is first written to the first word of the long word, and then to the second word.

8.6.1.4 TEST 4: EDC CHECK WORD GENERATE - BYTE WRITE

This test writes 1 byte data patterns to a memory location with EDC enabled, using known data for the other three bytes of the long word. Then the check word for the entire long word is verified. The test pattern is first written to the first byte of the long word, and then to the second byte, the third and finally to the fourth.

8.6.1.5 TEST 5: EDC DETECT SINGLE BIT ERROR

This test forces a single bit error and checks that the condition shows in the memory status register. This is done by altering a bit in a known long data word and check word combination. Then, with EDC disabled, this pattern is written to memory. EDC is then enabled and the memory location is read by a long read. This should cause a single bit error to occur. This test is done for all 32 data bits and 7 check bits.

8.6.1.6 TEST 6: EDC VERIFY SINGLE BIT SYNDROME CODE

This test forces a single bit error in the same manner as test 5. Then it checks for the memory status register to be correct. Next it checks that a level 2 interrupt has occurred and, if so, checks the syndrome code for correctness. This is done for all 32 data bits and 7 check bits.

8.6.1.7 TEST 7: EDC CORRECT SINGLE BIT ERROR

This test forces a single bit error in the same manner as test 5. It then checks that a level 2 interrupt has occurred and, if so, checks that the data which was read has been corrected. This is done for all 32 data bits and 7 check bits.

8.6.1.8 TEST 8: EDC DETECT DOUBLE BIT ERROR

This test forces a double bit error and checks that the condition shows in the memory status register. This is done by altering two bits in a known long data word and check word combination. Then, with EDC disabled, this pattern is written to memory. The EDC is then enabled and read back. The test checks for a level 7 interrupt, and then for the correct status in the NMI register. Different double bit error patterns are made so all 63 double bit syndrome codes are generated.

8.6.1.9 TEST 9: EDC DOUBLE BIT SYNDROME GENERATION

This test forces a double bit error as in test 8. The test checks for a level 7 interrupt occurring, and then for the correct status in the NMI register. Next it checks for the correct syndrome code to be returned. Different double bit error patterns are made so all 63 double bit syndrome codes are generated.

8.6.1.10 TEST A: EDC DETECT GROSS ERROR CONDITION

This test checks the ability of the EDC to detect the occurrence of either all zeroes or all ones in the 32 bits of data and 7 check bits. With EDC disabled, the pattern of all zeros and ones is written to memory. Then, with EDC enabled, a read to that location is done. The test verifies that a level 7 interrupt occurs and that the correct NMI status is returned.

8.6.1.11 TEST B: EDC WORD WRITE - SINGLE BIT ERROR

This test checks the system's ability to detect a single bit error in a read/modify/write cycle. A single bit error is injected into memory. A word write is performed to that location, causing a single bit error condition to occur on the read portion of the cycle. The location is then read to assure that the data was corrected and correctly written. Both words of a long word are used for testing.

8.6.1.12 TEST C: EDC BYTE WRITE - SINGLE BIT ERROR

This test checks the system's ability to detect a single bit error in a read/modify/write cycle. A single bit error is injected into memory. A byte write is performed to that location, causing a single bit error condition to occur on the read portion of the cycle. The location is then read to assure that the data was corrected and correctly written. All bytes of a long word are used for testing.

8.6.1.13 TEST D: EDC WORD WRITE - DOUBLE BIT ERROR

This test checks the system's ability to detect a double bit error in a read/modify/write cycle. A double bit error is injected into memory. A word write is performed to that location, causing a double bit error condition to occur on the read portion of the cycle. The test checks the NMI status and for a level 7 interrupt to verify the operation. The test checks both words of a long word.

8.6.1.14 TEST E: EDC BYTE WRITE - DOUBLE BIT ERROR

This test checks the system's ability to detect a double bit error in a read/modify/write cycle. A double bit error is injected into memory. A byte write is performed to that location, causing a double bit error condition to occur on the read portion of the cycle. The test checks the NMI status and for a level 7 interrupt to verify the operation. The test checks all bytes of a long word.

8.6.1.15 TEST F: EDC MEMORY ERROR ADDRESS STATUS

This test checks the ability of the Memory Error Address Status register to correctly identify the megabyte of memory where the last memory error occurred. An error is injected into each megabyte of memory on the system. The status is read and compared to the expected status. This test checks non-contiguous memory up to 24 MB. The only restriction is that one board must be strapped as the first megabyte of memory (0-FFFFF hex).

This test is not included in the system self-test.

8.6.1.16 TEST 10: EDC FILL MEMORY

This test first initializes memory by disabling EDC, loading the CBR with data and writing a long word to a long word boundary. The CBR data is the known check word corresponding to the long word data. This data is written from 400 hex to the end of the first megabyte of memory.

The default pattern written in 0 hex, with the check word 07 hex. If the defaults option is OFF, the user can select a range of memory to test and the long word pattern.

This test is not included in the system self-test.

8.6.1.17 TEST 11: EDC READ MEMORY

This test reads memory and reports and single of double bit errors. if an error occurs, the memory location in error is displayed. The default memory read is from 400 hex to the end of the first megabyte. If the defaults option is turned OFF, the user may select the range of memory to test.

8.6.2 EDC Commands

cbrw - Write Byte to Check Bit Register

This command prompts for a one byte input which is then written into the Check Bit register. The command is intended for use with EDC disabled, and with a long word write to memory. With EDC disabled, whatever is in the Check Bit register is written to memory with the next long word write.

enl2/dsl2 - Enable/Disable Level 2 Interrupts

These commands enable and disable level 2 interrupts, respectively. The interrupt mask is not raised or lowered by these commands. The enl2 command is most useful when deliberately setting single bit errors in memory.

eon/eoff - Enable/Disable EDC

These commands turn the EDC hardware ON and OFF, respectively. To enable EDC, a byte of 00 hex is written to the EDC control byte (0450 0054 hex). To disable EDC, the byte value written is FF hex. With EDC disabled, the user can manipulate the 32 bit long word and its associated 7 bit check word in memory.

lrd - Read Long Word From Memory

This command prompts for the memory address to be read, and then reads the long word at that address. The command is intended to be used with EDC disabled, followed by reading the Syndrome Register (rsyr command), allowing the user to see the check bits associated with the long word in memory.

lwrt - Write Long Word to Memory

This command prompts for the address in memory to be written with the long word. It then prompts for the data to be written. This command is most useful when used after loading the Check Bit register using the cbrw command.

rsyr - Display Syndrome Register

This command displays the Syndrome Register. With EDC enabled, the Syndrome Register is read after an EDC error is detected, and indicates either which single bit was in error, or whether a double bit or uncorrectable multiple bit error has occurred. With EDC disabled, the Syndrome Register returns the actual 7 check bits in memory associated with the last long word read from memory.

vint - Initialize Interrupt Vectors

This command initializes the MDS interrupt handling vectors.

8.6.3 EDC Error Reporting

When an error occurs, EDC always outputs the error number to the LEDs as a two hex digit code. The first digit is a 1, indicating that the EDC module is running. The second digit is the error code itself. When the sense switches are set for running a single EDC test, this is the only output from the system; otherwise, output is also sent to the VDT.

- Error 11

  This error occurs only in test 1. VDT output is:

  Bus Error on Access to EDC Register
  Register
  addr

  where addr is the address of the register.

- Error 12

  This error occurs only in test 2. VDT output is:

  Check Word Error on Long Word Write

LW Written	LW Received	CW exp	CW rec
data	data	xxh	xxh

  where data is the data pattern written or received and xxh is the check word expected or received.

- Error 13

  This error occurs only in test 3. VDT output is:

  Check Word Error on Word Write to Long Word

Word No.	Word Wrote	LW exp	LW rec
x	xxxxh	xxxxxxxxh	xxxxxxxxh

CW exp	CW rec
xxh	xxh

  where: Word No. is either 0 or 1
  Word Wrote is the actual word written
  LW exp is the long word in memory
  LW rec is the long word actually received
  CW exp is the check word expected
  CW rec is the check word actually received

- Error 14

This error occurs only in test 4. VDT output is:

Check Word Error on Byte Write to Long Word

Byte No.	Byte Wrote	LW exp	LW rec
x	xxxxh	xxxxxxxxh	xxxxxxxxh

CW exp	CW rec
xxh	xxh

where: Byte No. is 0 through 3
Byte Wrote is the actual byte written
LW exp is the long word in memory
LW rec is the long word actually received
CW exp is the check word expected
CW rec is the check word actually received

- Error 15

This error may occur in tests 5 and 6. The VDT will display one of the following three error messages:

Single Bit Error on Long Read Not Shown in Status
Data Bit (hex) Forced Error: xxh

where xxh is the data bit which should have caused the error, or

Single Bit Error on Long Read Not Shown in Status
Check Bit (hex) Forced Error: xxh

where xxh is the check bit which should have caused the error.

MSR Not Clear From Previous Read

- Error 16

This error may occur in tests 6 and 7. The VDT will display one of the following messages:

Single Bit error on Long Read Did Not Generate Level 2 Interrupt

The following message is displayed by test 6 only:

Syndrome Code Error on Single Bit Error

Syndrome exp	Syndrome rec
xxh	xxh

where xxh is the Syndrome codes that were expected and received, respectively.

- Error 17

This error occurs in test 7 only. The VDT will display one of the following messages:

```
EDC Failed to Correct Single Bit Error
Data Bit (hex) Forced        Data exp        Data rec
      xxh                      xdata           rdata
```

where xxh is the data bit forced in error, and xdata is the expected corrected data, and rdata is the data actually received.

```
EDC Failed to Correct Single Bit Error
Check Bit (hex) Forced       Data exp        Data rec
      xxh                      xdata           rdata
```

where xxh is the check bit forced in error, and xdata is the expected corrected data, and rdata is the data actually received.

- Error 18

This error may occur in tests 8 and 9. The VDT will display one of the following messages:

```
Double Bit Error Condition Did Not Generate lvl 7 Interrupt
CW Written          Data Written          NMI Status
  xxh                  data                  yyh
```

where xxh is the check word written, data is the data pattern written and yyh is the NMI status.

```
Double Bit Error Condition Did Not Set Double Bit
CW Written          Data Written          NMI Status
  xxh                  data                  yyh
```

where xxh is the check word written, data is the data pattern written and yyh is the NMI status.

- Error 19

This error occurs in test 9 only. The VDT output is:

```
EDC Syndrome Code Error With Forced Double Bit Error
exp Syndrome        rec Syndrome          CW Written
  xxh                  yyh                   zzh
Data Written        NMI Status
  data                status
```

where xxh is the expected syndrome, yyh is the actually received syndrome, \\h is the check word written, data is the data pattern and status is the NMI status received.

- Error 1A

This error occurs in test A only. The VDT will display one of the following messages:

Gross Error Condition Did Not Generate lvl 7 Interrupt

CW Written	Data Written	Data Received	NMI Status
xxh	wdata	rdata	yyh

where xxh is the check word, wdata is the data written, rdata is the data received and yyh is the NMI interrupt status.

Gross Error Condition Did Not Set Double Bit Error Bit in NMI Reg

CW Written	Data Written	Data Received	NMI Status
xxh	wdata	rdata	yyh

where xxh is the check word, wdata is the data written, rdata is the data received and yyh is the NMI interrupt status.

- Error 1B

This error occurs in test B only. The VDT will display one of the following messages:

Read of r/m/w Did Not Generate Level 2 Interrupt

Org CW	Check Bit Flipped	CW in Mem
xxh	yyh	zzh
Data in Mem	Word Written	
mdata	wdata	

where xxh is the original check word, yyh is the bit in the check word that was flipped, zzh is the check word read from memory, mdata is the data in memory and wdata is the data written to memory.

Read of r/m/w of Word Write to LW Did Not Generate Level 2 Interrupt

Word No.	Word Wrote	Org Data	Data Bit Forced
xxh	wdata	odata	yyh
LW Wrote	CW Wrote		
ldata	zzh		

where xxh is the word number to which the data was written, wdata is the data written, odata is the original data, yyh is the bit forced into error in odata, ldata is the long word in memory before r/m/w and zzh is the check word in memory before r/m/w.

Unexpected Level 2 Interrupt Occurred After Word r/m/w

Expected	Actual
edata	adata

Bad Data Read After Word Write With Distorted Check Word

Data exp	Data rec	Org CW	CW in Memory
edata	rdata	xxh	yyh

Bad Data Read After Word Write With Distorted Data

Data Exp	Data rec	CW in Memory	Word Written
edata	rdata	xxh	wdata

- Error 1C

This error occurs in test C only. The VDT will display one of the following messages:

Read of r/m/w Did Not Generate Level 2 Interrupt

Org CW	Check Bit Flipped	CW in Mem
xxh	yyh	zzh
Data in Mem	Byte Written	
mdata	wdata	

where xxh is the original check word, yyh is the bit flipped in xxh, zzh is the check word read from memory, mdata is the data in memory before the byte write and wdata is the data written to memory.

Read of r/m/w of Byte Write to LW Did Not Generate Level 2 Interrupt

Byte No.	Byte Wrote	Org Data	Data Bit Forced
xxh	wdata	odata	yyh
LW Wrote	CW Wrote		
ldata	zzh		

where xxh is the byte of the long word written, wdata is the byte written, odata is the data before the error was forced, yyh is the bit flipped in odata, ldata is the entire long word written and zzh is the check word in memory before r/m/w.

Unexpected Level 2 Interrupt Occurred After Byte r/m/w

Expected	Actual
edata	adata

Bad Data Read After Byte Write With Distorted Check Word

Data exp	Data rec	Org CW	CCW in Memory
edata	rdata	xxh	yyh

Bad Data Read After Byte Write With Distorted Data

Data exp	Data rec	CW in Memory	Byte Written
edata	rdata	xxh	wdata

- Error 1D

This error occurs in test D only. The VDT will display one of the following messages:

Read of r/m/w to Word Did Not Generate a Level 7 Interrupt

Word No.	Word Wrote	Org Data	LW Wrote
xxh	wdata	odata	ldata
CW Wrote			
yyh			

where xxh is the word number in the long word to which the data was written, wdata is the data written, odata is the original data before the error was forced, ldata is the long word in memory before r/m/w and yyh is the check word in memory before r/m/w.

Double Bit Error During Read of r/m/w Did Not Set Double Bit Error in NMI Reg

Good Data	Data in Memory	CW in Memory	Word Written
gdata	mdata	xxh	wdata

- Error 1E

This error occurs in test E only. The VDT will display one of the following messages:

Read of r/m/w to Byte Did Not Generate a Level 7 Interrupt

Byte No.	Byte Wrote	Org Data	LW Wrote
xxh	wdata	odata	ldata

CW Wrote
yyh

where xxh is the byte number in the long word to which the data was written, bdata is the data written, odata is the original data before the error was forced, ldata is the long word in memory before r/m/w and yyh is the check word in memory before r/m/w.

Double Bit Error During Read of r/m/w Did Not Set Double Bit Error in NMI Reg

Good Data	Data in Memory	CW in Memory	Word Written
gdata	mdata	xxh	wdata

- Error 1F

This error may occur in tests F and 11. If the error occurs in test F, the VDT will display this message:

MEAR Not Correct After Error in Memory

MEAR	Expected	Address Used
xxh	yyh	addr

If the error occurs in test 11, the VDT will display one of the following messages:

Level 7 Interrupt Occurred While Scanning Memory

Address Read	NMI	SYR	MEAR	Data Read
addr	xxh	yyh	zzh	data

Level 2 Interrupt Occurred While Scanning Memory

Address Read	NMI	SYR	MEAR	Data Read
addr	xxh	yyh	zzh	data

8.7 MEMORY TEST MODULE

The Memory Test Module, <mem>, consists of a series of test and scope loops.

The test loops allow the user to test memory from PROM and to define a memory test for a specific range of memory on a byte, word or long word boundary, and provide scope loops for testing the memory bus. The memory tests are designed to indicate single and double bit ECC errors, and to indicate the failing board and syndrome code. The tests are not exhaustive, but provide a quick and comprehensive test to find the majority of memory failures.

The scope loops all the user to access a single location or range of locations on a byte, word or long word boundary. The scope loops can be configured to test any location in the system memory map, including controllers and peripheral chips.

The only hardware requirements for running the Memory Test Module are the CMB and at least on memory board to test. A VDT is useful for error messages, but simple error codes are displayed in the LED.

Refer to Section 8.2.2 for instructions on starting MEM by sense switch setting. To begin MEM from the <mds> prompt or another test prompt, enter:

```
mem
```

The test module then starts, and the <mem> prompt is displayed.

The test default is to test all contiguous memory found during the initial sizing routine. This default can be overridden by using the **-def** or **form** commands (refer to Section 8.7.2).

8.7.1 MEM Test Descriptions

The MEM scope and test loops are executed on the range of addresses specified in the defaults. The data to write to memory or to expect from memory is also specified by the defaults. The NVRAM defaults can be overridden by entering the "-def" command prior to the "run" or "all" commands, or by using the "form" command.

8.7.1.1 TEST 1 (SCOPE LOOP) - LONG WRITE

This test writes one long word to memory to a single location, the start address. With defaults OFF, you may specify a range of memory addresses to test. Press **ENTER** alone at the TO: prompt to specify a single address. If a range of addresses is tested, the data is rotated right one bit for each successive location.

8.7.1.2 TEST 2 (SCOPE LOOP) - WORD WRITE SINGLE

This test writes one word to memory at a single location, the start address. With defaults OFF, you may specify a range of memory addresses to test. Press **ENTER** alone at the TO: prompt to specify a single address. If a range of addresses is tested, the data is rotated right one bit for each successive location.

8.7.1.3 TEST 3 (SCOPE LOOP) - BYTE WRITE SINGLE

This test writes one byte to memory at a single location, the start address. With defaults OFF, you may specify a range of memory addresses to test. Press **ENTER** alone at the TO: prompt to specify a single address. If a range of addresses is tested, the data is rotated right one bit for each successive location.

8.7.1.4 TEST 4 (SCOPE LOOP) - BYTE-WORD-LONG WRITE SINGLE

This test writes one byte to memory at a single location, the start address. It then writes a word, and then a long word to the same location.

8.7.1.5 TEST 5 (SCOPE LOOP) - LONG READ SINGLE

This test reads one long word from memory, at the start address or at an address specified by the user.

This test reads one word from memory, at the start address or at an address specified by the user.

8.7.1.7 TEST 7 (SCOPE LOOP) - BYTE READ SINGLE

This test reads one byte from memory, at the start address or at an address specified by the user.

8.7.1.8 TEST 8 (SCOPE LOOP) - BYTE-WORD-LONG READ SINGLE

This test reads one byte to memory at a single location, the start address. It then reads a word, and then a long word from the same location.

8.7.1.9 TEST 9 (TEST LOOP) - BYTE WRITE, READ, COMPARE SINGLE

This test accesses memory four times. First it writes the background data to memory, then reads the location and compares it. The data is then replaced with data formed by performing an exclusive OR on the least significant bit byte address of the location to write and the background data pattern. This modified data is then read back and compared. The access can be performed on a single location (the start address), or on a range of addresses specified by the user. A single location is the default. To specify a single location, press ENTER alone at the TO: prompt.

8.7.1.10 TEST A - WORD WRITE, READ, COMPARE SINGLE

This test accesses memory four times. First it writes the background data to memory, then reads the location and compares it. The data is then replaced with data formed by performing an exclusive OR on the least significant bit word address of the location to write and the background data pattern. This modified data is then read back and compared. The access can be performed on a single location (the start address), or on a range of addresses specified by the user. A single location is the default. To specify a single location, press **ENTER** alone at the TO: prompt.

8.7.1.11 TEST B - LONG WRITE, READ, COMPARE SINGLE

This test accesses memory four times. First it writes the background data to memory, then reads the location and compares it. The data is then replaced with data formed by performing an exclusive OR on the least significant bit long word address of the location to write and the background data pattern. This modified data is then read back and compared. The access can be performed on a single location (the start address), or on a range of addresses specified by the user. A single location is the default. To specify a single location, press **ENTER** alone at the TO: prompt.

8.7.1.12 TEST C - ADDRESSING

This test performs two passes through memory. The first pass writes each long word address with data equivalent to that address. Each address is then read back and verified. The second pass writes the inverted address to each address, then reads it back and verifies the data. Only the test range may be specified by the user, by using the **FORM** command.

8.7.1.13 TEST D (SCOPE LOOP) - BYTE WRITE RANGE

This test fills memory, one byte at a time, for the specified range. This scope loop must be run before the next test loop (test E) is run. It is separated from the read and compare loop to allow looping on the read. With defaults turned off, the user may specify the start and maximum addresses and the background pattern.

8.7.1.14 TEST E - BYTE READ AND COMPARE RANGE

This test reads and compares the data in a range of memory, one byte at a time. Test D must be run to write the background pattern before this test is run. With defaults turned off, the user may specify the start and maximum address, and the background pattern to expect. The test may be repeated (**EVER** command) to detect drop outs or intermittent ECC errors.

8.7.1.15 TEST F - BYTE WRITE, READ, COMPARE RANGE

This test first fills memory with a known pattern. It then reads and verifies a byte, writes the byte with new data, reads it back and compares it to the expected data, repeating for each byte in the range. With defaults OFF, the user may specify the range to test. The start and maximum addresses and the background pattern may be controlled by the **FORM** command.

8.7.1.16 TEST 10 - BYTE ROTATE

This test verifies memory one byte at a time. Refer to the description of the **RORDAT** test under the **TEST** command (Section 8.7.2). All parameters are controlled by the defaults, and can be changed by using the **FORM** command.

8.7.1.17 TEST 11 (SCOPE LOOP) - WORD WRITE RANGE

This test fills memory, one word at a time, for the specified range. This scope loop must be run before the next test loop (test 12) is run. It is separated from the read and compare loop to allow looping on the read. With defaults turned off, the user may specify the start and maximum addresses and the background pattern.

8.7.1.18 TEST 12 - WORD READ AND COMPARE RANGE

This test reads and compares the data in a range of memory, one word at a time. Test 11 must be run to write the background pattern before this test is run. With defaults turned off, the user may specify the start and maximum address, and the background pattern to expect. The test may be repeated (**EVER** command) to detect drop outs or intermittent ECC errors.

8.7.1.19 TEST 13 - WORD WRITE, READ, COMPARE RANGE

This test first fills memory with a known pattern. It then reads and verifies a word, writes the word with new data, reads it back and compares it to the expected data, repeating for each word in the range. With defaults OFF, the user may specify the range to test. The start and maximum addresses and the background pattern may be controlled by the **FORM** command.

8.7.1.20 TEST 14 - WORD ROTATE

This test verifies memory one word at a time. Refer to the description of the **RORDAT** test under the **TEST** command (Section 8.7.2). All parameters are controlled by the defaults, and can be changed by using the **FORM** command.

8.7.1.21 TEST 15 (SCOPE LOOP) - LONG WRITE RANGE

This test fills memory, one long word at a time, for the specified range. This scope loop must be run before the next test loop (test 16) is run. It is separated from the read and compare loop to allow looping on the read. With defaults turned off, the user may specify the start and maximum addresses and the background pattern.

8.7.1.22 TEST 16 - LONG READ AND COMPARE RANGE

This test reads and compares the data in a range of memory, one long word at a time. Test 15 must be run to write the background pattern before this test is run. With defaults turned off, the user may specify the start and maximum address, and the background pattern to expect. The test may be repeated (EVER command) to detect drop outs or intermittent ECC errors.

8.7.1.23 TEST 17 - LONG WRITE, READ, COMPARE RANGE

This test first fills memory with a known pattern. It then reads and verifies a long word, writes the long word with new data, reads it back and compares it to the expected data, repeating for each long word in the range. With defaults OFF, the user may specify the range to test. The start and maximum addresses and the background pattern may be controlled by the **FORM** command.

8.7.1.24 TEST 18 - LONG ROTATE

This test verifies memory one long word at a time. Refer to the description of the **RORDAT** test under the **TEST** command (Section 8.7.2). All parameters are controlled by the defaults, and can be changed by using the **FORM** command.

8.7.1.25 TEST 19 - LONG RANDOM

This test is similar to test 18, except that random data is used. The range of memory is first filled with a random data pattern. The data in each location is then replaced with a new random data pattern, but the contents of the location is first read and compared for the first random number. The second random data pattern that is written is formed by rotating the previous random data pattern right one bit. The final phase is to read the data back at each location, over the range to be tested. The range address may be controlled by the **FORM** command.

8.7.2 MEM Test Commands

The following commands may be entered when the <mem> prompt is displayed.

EVER

This command is just like **TEST**, except that the test runs continuously until the reset button is pressed. ESCAPE is not sensed during the test cycle. Normal MDS test controls are in operation.

FORM

This command allows you to change the MEM test default parameters. The test defaults are stored in NVRAM to retain the test parameters. The parameters include the test size (byte, word or long word), whether the test is a data or an addressing test, the starting and ending memory addresses to test, and the data patterns to use.

For each parameter, FORM displays the current parameter value. Some parameters toggle between acceptable values. Press the **SPACE** bar to toggle the value, then press **ENTER** to accept the displayed value. When a parameter requires an address or data value, press the **SPACE** bar to enter a new value, or **ENTER** to keep the current value. After typing in a new value, press **ENTER**. Motor bar keys may be used, as follows:

CTL-I	Accept the parameter (except following address or data entry).
CTL-II	Begin again at the first parameter.
CTL-III	Accept the remaining parameters unchanged.
CTL-IV	Not used.

INIT

This command initializes the NVRAM memory test definitions to the defaults. These defaults are the same as those used when the memory self-tests are executed. The defaults are as follows:

Start address	:	1000 hex
End address	:	(last contiguous memory location)
Background pattern	:	0BAD2EAF hex
Check pattern	:	12345678 hex
Controls	:	ECC ON, Instruction Cache ON

The start address is set at 1000 hex since the first 1000 locations are tested by the MDS pre-tests before the self-tests execute.

SHOW

This command displays the MEM test default parameters, as stored in NVRAM, including the address and data patterns.

TEST

This command executes the Rotated Or Random Data and Addressing Test (RORDAT) according to the default parameters specified by the **FORM** command. A test pass consists of 8 cycles, Each cycle writes data to the test range to initialize memory with the background pattern. Each location is then read and compared, and then replaced with the check pattern, rotated right one bit for successive locations. The final step of each cycle is to read the data back and verify the check pattern written throughout the test range.

The eight cycles are made up of the above steps in combinations of testing from top of memory down or from bottom up, using a normal or inverted fill pattern, and using a normal or inverted check pattern. Multiple passes are executed rotating the check pattern through all bits of the test definition size. For example, a long word test will do 32 passes, but the byte test will do only 8.

8.7.3 MEM Error Reporting

When an error occurs, MEM always outputs the error number to the LEDs as a two hex digit code. The first digit is a 2, indicating that the MEM module is running. The second digit is the error code itself. When the sense switches are set for running a single MEM test, this is the only output from the system; otherwise, output is also sent to the VDT.

The error number (LED displayed hex code) are:

Error	Meaning
21	Byte compare error
22	Word compare error
23	Long word compare error
24	Double bit ECC error
25	Single bit ECC error
26	Level 2 interrupt - unknown source

- Error 21-23 Memory compare error

These are memory compare errors. The VDT display is:

```
DATA CMP ERROR
ADDR        EXP        REC
xxxxxxxx    yy         zz
```

where xxxxxxxx is the address where the error occurred, yy is the expected data, and zz is the data actually received, with the data in byte, word or long word format.

- Error 24 Double Bit ECC Error

  The VDT display is:

  ```
  Double Bit ECC Error
  NMI stat                MEAR
  ```

  where NMI stat is the NMI status register, and MEAR is the Memory Error Address Register indicating the board where the failure occurred.

- Error 25 Single Bit ECC Error

  The VDT display is:

  ```
  Single Bit ECC Error
  NMI stat                MEAR          MSR           SYR
  ```

  where NMI stat is the NMI status register, MEAR is the Memory Error Address Register, MSR is the Memory Status Register, and SYR is the Syndrome Register.

- Error 26 Level 2 Interrupt - unknown source

  This message is displayed if a level 2 interrupt occurs and the source of the interrupt is not a single bit ECC error.

  The VDT display is:

  ```
  Level 2 Interrupt - unknown source
  NMI stat                MEAR          MSR           SYR
  ```

  where NMI stat is the NMI status register, MEAR is the Memory Error Address Register, MSR is the Memory Status Register, and SYR is the Syndrome Register.

8.8 MMU TEST MODULE

The MMU Test Module provides the ability to test the basic operation of the MMU. Messages are displayed on the VDT and as a hexadecimal code on the LED.

The MMU Test Module only requires a MAI 3000 CMB with 1 MB of memory. Presence of a VDT is optional.

Refer to Section 8.2.2 for instructions on starting MMU by sense switch setting. To begin MEM from the <mds> prompt or another test prompt, enter:

mmu

The test module then starts, and the <mmu> prompt is displayed.

8.8.1 MMU Test Descriptions

8.8.1.1 TEST 1 - WRITE/READ/COMPARE REGISTERS

This test determines if the MMU registers are accessible and retain the values written to them. If a data value mismatch is detected, the test displays the register value, and the expected and actual data values.

8.8.1.2 TEST 2 - WRITE/READ/COMPARE TC

This test determines if the MMU Translation Control (TC) register is accessible and retains the value written to it. If a data value mismatch is detected, the test displays the expected and actual data values.

8.8.1.3 TEST 3 - MODIFIED AND USED STATUS

This test enables MMU address translation with a 1:1 memory map. Memory within each page of the memory map is accessed with reads and writes. After each memory access, the appropriate page descriptor is checked to verify that the modified and used status bits are set/reset as expected. Bus errors when the MMU is enabled or when the test memory is accessed are reported. Incorrectly set/reset status bits are reported by displaying the logical memory address, page descriptor address and the expected and actual status bytes.

8.8.1.4 TEST 4 - WRITE PROTECT PAGE DESCRIPTOR

This test enables MMU address translation with a 1:1 memory map. The page descriptor of each page within the memory map is modified to indicate write protection, and memory within the page is accessed with reads and writes. After each memory access, the test verifies that a bus error did or did not occur as expected. If the expected result is not detected, the memory location address is displayed.

8.8.1.5 TEST 5 - WRITE PROTECT POINTER DESCRIPTOR

This test enables MMU address translation with a 1:1 memory map. The pointer descriptor of a page within the memory map is modified to indicate write protection, and memory within the page is accessed with reads and writes. After the memory access, the test verifies that a bus error did or did not occur as expected. If the expected result is not detected, the memory location address is displayed.

8.8.1.6 TEST 6 - INVALID POINTER DESCRIPTOR

This test enables MMU address translation with a 1:1 memory map. A pointer descriptor is modified to have an invalid type, and a memory access is attempted using that descriptor. If the expected bus error is not detected, the address in error is reported.

8.8.1.7 TEST 7 - INVALID PAGE DESCRIPTOR

This test enables MMU address translation with a 1:1 memory map. A page descriptor is modified to have an invalid type, and a memory access is attempted using that descriptor. If the expected bus error is not detected, the address in error is reported.

8.8.1.8 TEST 8 - UPPER LIMIT

This test enables MMU address translation with a 1:1 memory map. A pointer descriptor is modified to have an upper limit, and a memory access exceeding that limit is attempted using that descriptor. If the expected bus error is not detected, the address in error is reported.

8.8.1.9 TEST 9 - LOWER LIMIT

This test enables MMU address translation with a 1:1 memory map. A pointer descriptor is modified to have a lower limit, and a memory access exceeding that limit is attempted using that descriptor. If the expected bus error is not detected, the address in error is reported.

8.8.1.10 TEST 10 - TRANSLATION

This test writes the physical address of the long word to several pages of memory. It then enables MMU address translation with memory map swapping pages of memory to different addresses. The memory locations are read to verify that the translation has occurred, and then rewritten with the new logical address. The MMU address translation is disabled and the logical addresses are verified to be in the expected physical locations. When an address value error is detected, the expected and actual values are displayed.

8.8.1.11 TEST 11 - USER READ ACCESS LEVEL

This test moves user program code into RAM and enables MMU translation. When control is passed to the user code, an attempt is made to read a memory location which is validated for supervisor read access only. If the expected bus error is not received, the error is reported.

8.8.1.12 TEST 12 - USER WRITE ACCESS LEVEL

This test moves user program code into RAM and enables MMU translation. When control is passed to the user code, an attempt is mode to write a memory location which is validated for supervisor write access only. If the expected bus error is not received, the error is reported.

8.8.1.13 TEST 13 - EARLY TERMINATION

This test runs only when the PMMU is installed in the system. The root pointers are set to cause an early termination in the table search at the root pointer level. Various memory locations are accessed to verify that direct 1:1 translation is occurring properly.

8.8.2 MMU Test Commands

The following commands may be entered whenever the <mmu> prompt is displayed.

CRP - Display/Alter CRP

This command displays the current contents of the PMMU CRP (code root pointer) or MMB RP (root pointer) when no arguments are supplied. To set the MMB RP, one long word hexadecimal argument must be supplied. To set the PMMU CRP, two long word hexadecimal arguments must be supplied. Extreme caution must be used in setting the CRP/RP, since an incorrectly set up translation table could force the system into an immediate halt.

PSR - Display PSR

This command displays the current contents of the PMMU PSR (pmmu status register) when no arguments are supplied. To set the PMMU PSR, one long word hexadecimal argument must be supplied. This command is valid only on systems with a PMMU installed.

SRP - Display/Alter SRP

This command displays the current contents of the PMMU SRP (supervisor root pointer) when no arguments are supplied. To set the PMMU SRP, two long word hexadecimal arguments must be supplied. Extreme caution must be used in setting the SRP, since an incorrectly set up translation table could force the system into an immediate halt. This command is valid only on systems with a PMMU installed.

TC - Display/Alter TC

This command displays the current contents of the PMMU or MMB TC (translation control) register. To set the TC, on long word hexadecimal argument must be supplied.

8.8.3 MMU Test Error Reporting

When an error occurs, MMU always outputs the error number to the LEDs as a two hex digit code. The first digit is a 3, indicating that the MMU module is running. The second digit is the error code itself. When the sense switches are set for running a single MMU test, this is the only output from the system; otherwise, output is also sent to the VDT.

- Error 31

  This test occurs only in test 1. VDT output is:

  Register data miscompare
  Reg: nn Num: mm
  Expect: xxxxxxxx Actual: yyyyyyyy

  where nn is the register number, mm is the register number for BAD and BAC registers, xxxxxxxx is the expected data and yyyyyyyy is the actual data received.

- Error 32

  This error may occur in tests 3 through 5 and 13. VDT output is:

  Unexpected Address Error
  Address: xxxxxxxx

- Error 33

  This error may occur in tests 4 through 12. VDT output is:

  Not Received: Expected Address Error
  Address: xxxxxxxx

- Error 34

  This error may occur in tests 3 through 5 and 13. VDT output is:

  Unexpected Bus Error
  Address: xxxxxxxxx

- Error 35

  This error may occur in tests 3 through 13. VDT output is:

  Enabling MMU: Unexpected Bus error
  Address: xxxxxxxx

- Error 36

  This error may occur in tests 4 through 12. VDT output is:

  Not Received: Expected Bus Error
  Address: xxxxxxxx

- Error 37

  This error occurs in test 3 only. VDT output is:

  ```
  Wrong Status Byte in Page Descriptor
  Logical: xxxxxxxx        Descriptor: yyyyyyyy
  Expect: aa               Actual: bb
  ```

  where aa is the expected byte value and bb is the actual byte contents of the page descriptor at address yyyyyyyy.

- Error 38

  This error occurs in test 2 only. VDT output is:

  ```
  TC register miscompare
  Expect: xxxxxxxx         Actual: yyyyyyyy
  ```

- Error 39

  This error occurs in test 1 only. VDT output is:

  ```
  RP register miscompare
  Expect: xxxxxxxx         Actual: yyyyyyyy
  ```

- Error 3A

  This error may occur in tests 11 and 12. VDT output is:

  ```
  PROC argument error
  Value: nnnn
  ```

  where nnnn is the value of the argument supplied by the user task making the supervisor call. This is an abnormal software error in the test logic.

- Error 3B

  This error occurs in test 13 only. VDT output is:

  ```
  Memory data miscompare
  Address: xxxxxxxx
  Expect: nnnnnnnn         Actual: mmmmmmmmm
  ```

- Error 3F

  This error may occur in tests 11 and 12. VDT output is:

  ```
  User task did not start
  ```

8.9 CMB TEST MODULE

The CMB Test Module provides the ability to test the basic functions of the PI/T, RTC and SCC components of the CMB. Messages are displayed on the VDT and as a hexadecimal code on the LED.

The MMU Test Module only requires a MAI 3000 CMB. Presence of a VDT is optional.

Refer to Section 8.2.2 for instructions on starting CMB by sense switch setting. To begin MEM from the <mds> prompt or another test prompt, enter:

```
cmb
```

The test module then starts, and the <cmb> prompt is displayed.

When the CMB module is selected, it checks the Valid Time/Ram (VTR) bit. If power has failed to the RTC, the program displays the message:

```
VTR bit not initially set
use SET_TIME command to start clock before running rtc tests.
```

8.9.1 CMB Test Descriptions

8.9.1.1 TEST 1 - PI/T REGISTER WRITE/READ

This test writes data to selected registers and reads it back to test the ability of each register to store data and pass data back to the CPU. Each register is first written with 55 hex, and then read to verify the data. When each register has been so checked, another pass is completed using inverted 55 hex data.

8.9.1.2 TEST 2 - PI/T REGISTER ADDRESSING

This test writes, reads and compares each selected register with the least significant 8 bits of its own address. All registers are written to, and then all are read and compared. This verifies the ability to uniquely address each PI/T write/read register. Failures in the address selection causes one register to be overwritten with the address of another. The inverted address is used on the second and subsequent passes.

8.9.1.3 TEST 3 - PI/T TIMER INTERRUPT

This test determines whether the PI/T autovector for timer interrupt works correctly. The 68020 interrupt mask is set to level 0 and PI/T Counter Autovectored Interrupt is generated by loading the timer with a precount of 000001 hex. The counter is then enabled to count down to 00000000. The test verifies that the interrupt occurred, and checks for correct timer status.

8.9.1.4 TEST 4 - RTC RAM WRITE/READ

This test writes and reads data to the fifty bytes of user RAM on the RTC chip to test the ability of each to store and pass data back to the CPU. The test writes 55 hex to the first byte, then reads it back and verifies the data. This is repeated for each byte of user ram. The test is then repeated using inverted AA hex data.

8.9.1.5 TEST 5 - RTC RAM ADDRESS

This test writes, reads and verifies each of the fifty bytes of user RAM on the RTC chip, using the least significant 8 bits of a byte address as the data for that address. This verifies the ability to uniquely address each RTC user byte as a unique location. Failures in the address selection cause one location to be overwritten with the address of another. The inverted address is used on the second and subsequent passes.

8.9.1.6 TEST 6 - RTC STATUS

This test checks the normal running status of the RTC. First the VTR bit in RTC register D is verified to be set. Then the UIP bit in register A is checked to see that it functions, by switching from ON to OFF and from OFF to ON.

8.9.1.7 TEST 7 - RTC DISPLAY TIME

This test reads the status to get the current time and calendar of the RTC, and displays the time, day and date.

8.9.1.8 TEST 8 - RTC ALARM STATUS

This test determines whether the RTC alarm functions correctly. The current alarm values are saved and a "don't care" condition is inserted as alarm values. The alarm is checked after the next RTC update.

8.9.1.9 TEST 9 - SCC READ ONE REGISTER

This test allows scoping a read of one SCC register of one channel. The default is register 0 of channel B. If defaults are OFF, the channel and register to be scoped can be specified.

8.9.1.10 TEST A - SCC WRITE ONE REGISTER

This test allows scoping a write to one SC register of one channel.. The default is register 0 of channel B with data of 00 hex. If defaults are OFF, the channel, register and data can be specified.

8.9.1.11 TEST B - SCC READ ALL REGISTERS

This test allows scoping reads of all SCC registers on one channel. The default is channel B. If defaults are OFF, the channel to be scoped can be specified.

8.9.1.12 TEST C - SCC WRITE ALL REGISTERS

This test allows scoping writes of all SCC registers on one channel. The default is channel B with data of 00 hex. If defaults are OFF, the channel and data can be specified.

8.9.1.13 TEST D - SCC INTERNAL LOOPBACK

This is a manual intervention test. The test sets up a SCC channel for internal loopback mode. Then characters are transmitted and, if successful, received. The data received is verified against what was sent. The default is to test channel B. If defaults are OFF, the channel can be specified.

8.9.1.14 TEST E - SCC TX 55 HEX ON DATA PINS

This is a manual intervention test. The test checks the ability to transmit to the data pins of a SCC channel. The default is channel B. If defaults are OFF, the channel to use can be selected. A 55 hex is always written to the ports.

8.9.1.15 TEST F - SCC TX 55 HEX WITH INTERRUPTS

This is a manual intervention test. The test checks the occurrence of a transmit buffer empty interrupt after a character is sent. The channel is set up to transmit interrupts and to give a vector during interrupt service. Checks are made for an interrupt occurring, the transmit buffer empty bit set and transmit interrupt pending. The default is channel B. If defaults are OFF, the channel can be specified.

8.9.1.16 TEST 10 - SCC ECHO PORT

This test echoes data from input to the port selected. The default is channel B. If defaults are OFF, the channel can be specified.

8.9.2 CMB Test Commands

The following commands can be entered whenever the <cmb> prompt is displayed.

SET - Set/Start RTC

This command sets and starts the RTC chip. The user is prompted for the information required. The first prompt asks for the enable byte. Press **ENTER** for the default value (81 hex) which selects decimal, 12-hour mode with daylight savings. The next prompt asks for the control byte. Press **ENTER** for the default value (20 hex) which is the standard value for the clock divider entry. The remaining prompts ask for the day of the week, the month, the date, year, hour, minutes and seconds. In 12-hour mode, enter 01 through 12 for AM times, and 81 through 92 for PM times.

8.9.3 CMB Test Error Reporting

When an error occurs, CMB always outputs the error number to the LEDs as a two hex digit code. The first digit is a 4, indicating that the CMB module is running. The second digit is the error code itself. When the sense switches are set for running a single CMB test, this is the only output from the system; otherwise, output is also sent to the VDT.

- Error 41

  This error occurs in test 1 only. VDT display is:

  ```
  PIT Reg. Data Miscompare
  PIT Ref        EXP        REC
  addr           xxh        yyh
  ```

  where addr is the register address, xx is the expected data and yy is the data actually received.

- Error 42

  This error occurs in test 2 only. VDT display is:

  ```
  PIT Unique Address
  ADDRESS        EXP        REC
  addr           xxh        yyh
  ```

  where addr is the register address, xx is the expected data and yy is the data actually received.

- Error 43

  This error occurs in test 3 only. VDT display is:

  ```
  PIT Timer Interrupt Did Not Occur
  ```

  or

  ```
  Bad PIT Int.              status: xxh
  ```

  where xx is the status of TSR in hexadecimal.

- Error 44

  This error may occur in tests 4 and 5. VDT display is:

  ```
  RTC RAM Error at ADDR:  addr
  EXP            REC
  xxh            yyh
  ```

  where addr is the register address, xx is the expected data and yy is the data actually received.

- Error 45

  This error occurs in test 6 only. VDT display is:

  RTC Power Failure

- Error 46

  This error may occur in tests 6 through 8. VDT display is:

  RTC Update Failure

- Error 47

  This error occurs in test 8 only. VDT display is:

  RTC Alarm Failure

- Error 48

  This error occurs in test D only. VDT display is:

  Timed Out Waiting for RX Available

- Error 49

  This error occurs in test D only. VDT display is:

  TX Char Not Equal to RX Char

TX Char	RX Char	SCC Base
xxh	yyh	addr

  where addr is the SCC channel base address, xx is the character transmitted and yy is the character received.

- Error 4A

  This error occurs in test F only. VDT display is:

  No TX Interrupt - Channel Base Addr: addr

  where addr is the SCC channel base address.

- Error 4B

  This error occurs in test F only. VDT display is:

  TX Int Pending Not Set - Channel Base Addr: addr

  where addr is the SCC channel base address.

- Error 4C

  This error occurs in test F only. VDT display is:

  TX Buffer Empty Not Set - Channel Base Addr: addr

  where addr is the SCC channel base address.

8.10 WDC TEST MODULE

The WDC Test Module provides test and scope loops for the EBUS.

The WDC Test Module requires a MAI 2500/3000/4000 CMB with at least one memory board, at least one WDC board and a disk drive attached as unit 0. The disk drive must be formatted, and may be a system disk with the O.S installed. Presence of a VDT is optional.

Refer to Section 8.2.2 for instructions on starting WDC by sense switch setting. To begin MEM from the <mds> prompt or another test prompt, enter:

wdc

The test module then starts, and the <wdc> prompt is displayed.

When the WDC module is selected, it sizes for the number of controllers present. The default is to test all controllers present. With defaults turned OFF, the user can specify which controllers to test.

8.10.1 Data Protection

The WDC tests are designed to preserve data on the disk drive. Only functions which do not cause writes to disk are executed when the test is in the default mode of operation.

The last two tests perform writes to disk, but are executed only if manual intervention mode is selected, using the MAN command (refer to Section 8.3.4). These tests are never executed during the power on self-test. If MDS is in manual intervention mode and one of these tests is selected to run, the test pauses before execution and asks you if you wish to continue:

Continue: ("yes"/CR)

You must enter **yes** to continue with the test; anything else causes the write operation to be skipped.

8.10.2 WDC Test Descriptions

The WDC Test Module consists of a number of tests and scope loops. Some of the low numbered loops are not tests, but are provided for scoping only. Generally the low level tests do not require the disk drive, but only the controller.

When more than one controller board is found in the system, the test and scope loops will do the operation on all boards found, by default. If defaults are turned OFF, you are asked which controllers to test, and you may specify a single controller.

A listing of the data patterns used in these tests is provided in Section 8.9.4.

8.10.2.1 Test 1 - WDC Write Vector Register

This is a scope loop, and allows scoping the writing of the vector register on the WDC controller to test EBUS write cycles. The default data is 75 hex, and is shifted left one bit for each successive write to the first and other controllers. This changes the data for each loop, returning to 75 for the eighth loop. If defaults are OFF, the controller(s) to test and the data can be specified.

8.10.2.2 Test 2 - WDC Read Vector Register

This is a scope loop, and allows scoping the reading of the vector register on the WDC controller to test the EBUS read cycles. The data read is placed in D2. If defaults are OFF, the controller(s) to test can be specified.

8.10.2.3 Test 3 - WDC Read and Compare Vector Register

This is a scope loop, and allows scoping the writing and reading of the vector register on the WDC controller. The data is compared on each loop. This tests the EBUS ability to transfer byte data across the EBUS. The default data is 01 hex, and is rotated one bit left for each loop. The data is written to the register twice, and then read back once for each loop. If defaults are OFF, the data and the controller(s) to test can be specified. If alternate data is specified, it is not modified (rotated) for successive loops.

8.10.2.4 Test 4 - Write the Control Register

This is a scope loop, and allows scoping the writing of the control register on the WDC controller. The default data is 01 hex, and is rotated each loop. Only the 4 least significant bits are used. If defaults are OFF, the controller(s) to test and the test data can be specified.

NOTE

The reset line is activated during this test, and may cause the controller to enter a bad state. After running this test, issue the **reset** command to reset the controller (refer to Section 8.3.5).

8.10.2.5 Test 5 - Read the Control Register

This is a scope loop, and allows scoping the reading of the WDC control register and the EBUS byte read cycle. The data read is placed in D2. If defaults are OFF, the controller(s) to test can be specified.

8.10.2.6 TEST 6 - WDC WRITE, READ AND COMPARE CONTROL REGISTER

This test tests the WDC control register and allows scoping the write and read cycles of the EBUS. The data is compared in each loop. The default data written is 01 hex, rotated for each loop. Only bits 1, 2 and 3 are tested. The data is written to the register twice and read once. If defaults are OFF, the data and controller(s) to test can be specified. If data is specified, it is not rotated for successive writes.

8.10.2.7 TEST 7 - WDC WRITE THE DATA REGISTER

This is a scope loop, and allows scoping the writing of the WDC data register. The scope loop can be used to scope write cycles of the EBUS. Default data is 75 hex, and is rotated for each loop. If defaults are OFF, the data and the controller(s) to test can be specified.

8.10.2.8 TEST 8 - WDC READ DATA REGISTER

This is a scope loop, and allows scoping the reading of the WDC data register. The data read is placed in D2. If defaults are OFF, the controller(s) to test can be specified.

8.10.2.9 TEST 9 - WDC WRITE, READ AND COMPARE DATA REGISTER

This test checks the WDC data paths. The test writes the output register twice, reads the input data register once, and compares the data for each loop. This test can be used to scope the EBUS byte write and read cycles. Default data is 01 hex, and is rotated for each loop. All eight bits are used. If defaults are OFF, the data and controller(s) to test can be specified. If the data is specified, it is not rotated.

8.10.2.10 TEST A - WDC WRITE THE LONG WORD DMA REGISTER

This is a scope loop, and writes to the WDC DMA register. Three bytes are written to the register, one byte at a time. Default data is 10000 hex, the address of the register, inverted and shifted right one bit. If defaults are OFF, you can specify the data and the controller(s) to test.

8.10.2.11 TEST B - WDC RESET CONTROLLER, CHECK NULL STATUS

This test sends the Clear Controller command to cause a null status. The test sets the clear controller bit and checks to see that it is set. It then initializes the sequencer by writing the output register and reading the input register twice, and checking for a 00 hex status, indicating the reset and status feedback work correctly. If defaults are OFF, you can specify the controller(s) to test.

8.10.2.12 TEST C - WDC SELECT

This test verifies that issuing the Select command causes the WDC to return Busy and Command Phase status. The test first initializes the controller by clearing the controller, and then selects the controller and tests for selected status.

Following this test, the controller is in command mode, waiting for a command. When the test is executed in a loop, the reset before select causes the command mode to be aborted. When the test completes normally, or after an "escape", the controller remains in command mode.

If defaults are OFF, the user may specify the controller(s) to test.

8.10.2.13 TEST D - WDC TEST MODE DMA

The test verifies correct function of the DMA logic by forcing the sequencer to do a DMA cycle without involving the disk interface or handshake. 512 bytes are read by the DMA logic. For each byte, the data accessed by DMA is compared with the expected data. The test displays all bus errors detected as well as any compare errors if the DMA logic fails to access the correct data.

Default data is a decrementing data pattern. The default address is 10000 hex. If defaults are OFF, the user may specify the memory address and the controller(s) to test.

8.10.2.14 TEST E - WDC DMA BUS ERROR

This test checks the controller's ability to indicate a bus error condition in its status byte when doing a test mode DMA transfer while bus master. It also checks that the bus error clear command clears the bus error status bit. If defaults are OFF, the user may specify the controller(s) to test.

8.10.2.15 TEST F - WDC BUS ERROR INTERRUPT DETECT

This test checks the controller's ability to indicate a bus error condition in its status byte when doing a test mode DMA transfer while bus master. It also checks that the bus error clear command clears the bus error status bit, and that the bus error caused a vectored interrupt. If defaults are OFF, the user may specify the controller(s) to test.

8.10.2.16 TEST 10 - WDC SASI HANDSHAKE

This test checks that the controller passes through all the SASI handshake phases of the sense command with no timeouts. This test uses the WDC driver and sends a complete WDC command. If defaults are OFF, the user may specify the controller(s) to test.

8.10.2.17 TEST 11 - WDC UNIT STATUS

This test determines the status of the attached unit 0 disk drives. The test issues the Unit Status command, and verifies that drive 0 is ready and that the controller handshake works. This test checks all unit 0 drives in the system. If unit 0 is not ready, the test fails. If defaults are OFF, the user may specify the controller(s) and attached unit 0 drives to test.

8.10.2.18 TEST 12 - WDC WRITE, READ AND COMPARE SECTOR BUFFER IN NON-DMA

This test checks the WDC sector buffer without involving the WDC DMA logic. It is intended to be used with tests 13 and 14 to determine if the DMA logic is functioning correctly or if a timing problem exists in a DMA data transfer. The test transfers data to and from the WDC sector buffer by directly transferring data between the WDC input and output registers. If defaults are OFF, the user may specify the data to be written and expected for the comparison by specifying the data type (default is type A), and the controller(s) to test.

8.10.2.19 TEST 13 - WDC WRITE SECTOR RAM IN DMA

This test issues the Write Sector Buffer command to transfer data to the WDC sector buffer. The test verifies that the normal DMA read cycles work on the EBUS and that the WDC can transfer data in DMA. The success of the DMA transfer is tested by test 14, which reads and compares the data. The test requires that the disk drive is attached and ready.

The default data is type A, incrementing word pattern, 1024 bytes in length. If defaults are OFF, the user may specify any data type and the controller(s) to test.

8.10.2.20 TEST 14 - WDC READ AND COMPARE SECTOR BUFFER DATA IN DMA

This test checks the ability of the sector buffer to transfer data, and that the data is the same as the expected data. It also is a check that the DMA EBUS write cycle works. The data expected is the data written by test 13. If defaults are OFF, the user may specify the expected data type and the controller(s) to test.

8.10.2.21 TEST 15 - WDC WRITE, READ AND COMPARE SECTOR BUFFER DATA IN DMA

This test checks the ability of the DMA logic to transfer data to and from the sector buffer, and that the data read is the same as the data written. When looping is enabled, the data pattern is changed for each loop. The test begins by writing data pattern A, the incrementing word data pattern. If defaults are OFF, the user may specify the data type to be written and expected and the controller(s) to test.

8.10.2.22 TEST 16 - WDC DISK RECALIBRATE

This test checks the ability of the drive to execute the recalibrate command, and that the correct status is indicated by the disk drive and controller. The test recalibrates the heads on unit 0 of the first controller. The test does not check all controllers. If defaults are OFF, the user may specify the controller and unit 0 to test.

8.10.2.23 TEST 17 - WDC DISK READS IN DMA

This test checks that the controller can read the disk with no controller detected errors. The test reads logical block 17, then logical block A000 hex, to cause implied seeks before each 16 block read. The test performs disk reads on unit 0 of the first controller found. If defaults are OFF, the user may specify the controller and unit 0 to test.

Failing this test may indicate that the disk drive is not formatted or is otherwise failing.

8.10.2.24 TEST 18 - WDC RANDOM SEEKS

This test performs random seeks on unit 0 of the first controller found. If defaults are OFF, the user may specify the controller and unit 0 to test.

8.10.2.25 TEST 19 - WDC DISK WRITE, READ AND COMPARE DATA

This is a manual intervention test. You must confirm running the test before it is executed. The test performs writes to disk, and is destructive of data.

This is a disk data reliability test. The test begins by writing, then reading and comparing the incrementing word data pattern (type A). Succeeding loops use the next data type, until all data types have been used. If defaults are OFF, the user may specify the data type to use, or enter his own data. If the user specifies the data pattern, the pattern remains the same through succeeding loops. The controller and unit 0 may also be selected.

8.10.2.26 TEST 1A - WDC RANDOM READ, WRITE-READ-COMPARE

This is a manual intervention test. You must confirm running the test before it is executed. The test performs writes to disk, and is destructive of data.

This test checks the controller's ability to execute random seeks and to maintain data reliability using different data patterns. The test first reads the data at a location, then writes a pattern, reads it back and verifies the data. 18 sectors per transfer are used to force head transition, and in some cases cylinder boundary changes. The test performs 50 loops, causing 50 random seeks and data transitions.

If defaults are OFF, the user may select a data pattern, which is then not changed during successive loops. The controller and unit 0 to test may also be specified.

8.10.3 WDC Test Error Reporting

8.10.3.1 ERROR CODES

When an error occurs, WDC always outputs the error number to the LEDs as a two hex digit code. The first digit is a 5, indicating that the WDC module is running. The second digit is the error code itself. When the sense switches are set for running a single WDC test, this is the only output from the system; otherwise, output is also sent to the VDT.

- Error 51

  This error displays the message:

  ```
  Bad WDC status after reset issued
  Status: xx
  ```

  where xx is the controller status.

- Error 52

  This error displays the message:

  ```
  Bad WDC status after Select
  Status:  xx
  ```

  where xx is the controller status.

- Error 53

  This error displays the message:

  ```
  WDC reg miscompare
  EXP     REG     Wrt addr        Read addr
  xx      yy      Cx000n          Cx000n
  ```

  where xx is the expected data, yy is the actual data read and Cx000n is the controller register used.

- Error 54

  This error displays the message:

  ```
  WDC bus error detected
  ```

  It indicates that an unexpected bus error occurred while the WDC was bus master. The bus error is indicated in the WDC status register.

- Error 55

  This error displays the message:

  ```
  Test DMA data compared error
  EXP             REC
  xx              yy
  ```

  where xx is the data expected and yy is the data actually received.

- Error 56

  This error displays the message:

  ```
  Forced DMA bus error did not occur
  Status: xx
  ```

  where xx is the contents of the WDC status register.

  This message indicates that a DMA register was loaded with an address that should have caused a bus error during a DMA transfer, but the error did not occur.

- Error 57

  This error displays the message:

  WDC bus error status did not reset
  Status: xx

  where xx is the contents of the WDC status register.

- Error 58

  This error displays the message:

  No WDC interrupt from forced WDC bus error

  This message indicates that a DMA register was loaded with an address that should have caused a bus error during a DMA transfer, but the interrupt was not sent.

- Error 59

  This error displays the message:

  Bad DOB status: xx

  where xx is the DOB status, as described in Section 8.10.3.3.

- Error 5A

  This error displays the message:

  WDC detected error
  Ending status: xx

  where xx is the WDC SASI ending status, as described in Section 8.10.3.4.

- Error 5B

  This error displays the message:

  WDC Status Handshake timeout

  This message indicates that a timeout occurred while waiting for a normal status phase of the SASI bus. This can occur for the command phase, status phase (command complete), message or bus free phase. Any timeout in the handshake will result in this error.

- Error 5C

  This error displays the message:

  WDC Message Byte Non-Zero

  This message indicates that the actual message byte received from the WDC during the message phase was non-zero, but should have been zero.

- Error 5D

  This error displays the message:

  WDC DMA Write/Read Error

  This message indicates that the check condition in the ending status occurred during a DMA write or read operation.

- Error 5E

  This error displays the message:

  WDC Stuck Busy

  This message indicates that the ending status from a WDC command remained busy after retrying the operation.

- Error 5F

  This error displays the message:

  Buffer Miscompare

  This message indicates that the data in the buffers used in main memory to compare data transferred to and from the WDC controller is not the same. Use the MDS "type" command to display the buffer content. The read buffer is in memory location 200000 hex, and the write buffer is in 100000 hex.

8.10.3.2 ERROR MESSAGES

The following messages may be displayed on the system console.

- No board found to test.

  This message is displayed if the sizing routine found no controller boards when the WDC module is called.

- WDC intr. svc. routine hung!

  This message is displayed by the interrupt service routine when the WDC interrupt will not clear. Attempting to do a RTE instruction in the interrupt service routine caused the routine to be called again. The message is displayed after the service routine is called 128 times.

- Sense: ss bb bb bb

  This message displays the sense information from the sense command in response to the check condition. In the data, ss is the WDC detected error code, and bb are the bytes that define the disk block involved in the error.

- WARNING: This test WILL write on the disk
  Continue? ("yes"/CR):

  This message is displayed before executing a test that writes to the disk (tests 19 and 1A). These tests run only if MDS is in manual mode, invoked by the MAN command. To continue with the test, enter "yes". Any other response terminates the test.

8.10.3.3 DEVICE ORDER BLOCK (DOB) STATUS

Some error messages displayed by the WDC test module include the DOB status of the driver. The DOB is generic, and is used by other devices also. The codes are shown in Table 8-5.

Table 8-5. DOB Status Codes

BIT	MEANING
7	Error. An error was detected by the driver, the meaning of which is in the other status buts. The operation was not performed.
6	Format Error. The DOB for the device is formed incorrectly. The DOB is generated by the program, and so reflects a program error.
5	Fatal Device Error. The error, as defined by the other status words, is an error the driver cannot recover from.
4	Init Error. The driver call to initialize the driver and the device resulted in an I/O error. The device cannot be used.
3	Bad Unit. The DOB defines an invalid unit.
2	Bad Operation. The DOB defines an invalid operation from the device indicated.
1	DCB Format Error. The Device Control Block created for the device is formed incorrectly. This is usually a program error.
0	DCB Transfer Error. The driver detected an error transferring the DCB to the device. This is an I/O error.

8.10.3.4 WDC ENDING STATUS

Some error messages indicate the WDC ending status when the controller detects an error. The ending status is the SASI status, as it is used by the WDC controller. The normal ending status is 02 hex, indicating a unit check condition. The codes are shown in Table 8-6.

Table 8-6. WDC Ending Status Codes

BIT	MEANING
7-4	Reserved (not used)
3	Unit Busy
2	Equal (set when a search is Equal)
1	Check (The sense bytes define the error)
0	Reserved (not used)

8.10.3.5 WDC STATUS

The WDC controller status is displayed by most WDC tests. The status byte indicates normal handshaking status during SASI operations, as well as controller errors. The codes are shown in Table 8-7.

Table 8-7. WDC Status Codes

BIT	MEANING
7	CMB. The controller is in the command, status or message phase.
6	BUSY. The SASI bus is busy.
5	MSG. The controller is in the message phase.
4	RESET. The controller is in the reset state. This status is normally present when the soft reset command is issued by the program.
3	INPUT. The input data register is full. This is normal for the status, message and data transfer phases.
2	COMMAND COMPLETE. The operation is complete and at the beginning of the status phase. The ending status is available when bit 3 comes on.
1	OUTPUT. The output data register is empty. The controller is ready for data or command.
0	BUS ERROR. The controller detected a bus error while it was bus master.

8.10.4 WDC Disk Data Patterns

The WDC tests use 16 standard data patterns. The user can select these patterns when defaults are turned OFF. The data patterns are described in Table 8-8.

Table 8-8. WDC Test Data Patterns

TYPE	DESCRIPTION	DATA WORDS
0	All zeros	0000 0000 0000 0000 0000
1	All ones	FFFF FFFF FFFF FFFF FFFF
2	Floating 0	FFFE FFFD FFFB FFF7 FFEF
3	Floating 1	0001 0002 0004 0008 0010
4	Random	F946 EA89 12DD 32A0 00FF
5	All fives	5555 5555 5555 5555 5555
6	Di-bit	CCCC CCCC CCCC CCCC CCCC
7	Tri-bit	E38E 38E3 8E38 E38E 38E3
8	Quad-bit	F0F0 F0F0 F0F0 F0F0 F0F0
9	Zero di-bit	6DB6 DB6D B6DB 6DB6 DB6D
A	Incrementing	0001 0002 0003 0004 0005
B	Zero quad-bit	7777 7777 7777 7777 7777
C	Zero quad-bit	7BDE F7BD EF7B DEF7 BDEF
D	One di-zero	9249 2492 4924 9249 2492
E	One tri-zero	8888 8888 8888 8888 8888
F	One quad-zero	8421 0842 1084 2108 4210
10	Increase freq	FF00 FE03 F03E 0F0E 32AA
11	Decrease freq	FEAA 6663 8E38 7878 783F
12	Jitter	9659 6596 5965 9659 6596
13	Worst Case 1	D936 4DB6 4ED9 364D B64E
14	Worst Case 2	6DB6 DB0F 6DB6 DB0F 6DB6
15	Worst Case 3	DADA CA58 C2FE DADA CA58

8.11 INSTRUCTION/DATA CACHE TEST MODULE (<CACHE>)

The Cache Test Module provides thirteen scope loops to progressively test the functionality of the CMB Cache. For diagnostic purposes, the cache consists of 8KB of high speed memory, divided into two sets of 4KB (1 K long words) each. Each addressable location of this memory has its own associated Tag Word for determining whether a given data value is correct or not.

The configuration of the CMB Cache is controlled through ten write registers:

READ	Enables/disables the cache from storing data.
WRITE	Enables/disables the cache from outputting data.
DATA	Enables/Disables the cache to consider data operands only.
INSTRUCTION	Enables/disables the cache to consider instruction operands only.
TAGTEST	Forces the cache to set the associated tag word parity error of the data operated on.
DCTEST	Forces the cache to set the associated data word parity error of the data operated on.
INHIBIT LVL3	Enables the cache to send a level 3 interrupt to the CPU. This is used to signal a CMB Cache parity error.
FORCE SET 0	Forces the CMB Cache to use set 0 only.
CACHE FLUSH	Clears the CMB Cache memory.
CACHE INHIBIT	Enables the CMB Cache following a parity error.

8.11.1 Cache Test Descriptions

8.11.1.1 TEST 1 - CACHE DATA TEST

This test checks the ability of the cache to detect data errors between the cache and memory. The test flushes the cache and then configures caching for data operands, enables reads and writes, and forces a set 0. It then writes 1K long words to, and reads them from memory, thus loading the data into the cache. The reads and writes to the cache are then disabled, and another 1K long words with different data values are written to the same location, so the cache contains one data pattern and the memory buffer has another, both with the same addresses. Cache reads and writes are re-enabled, and the 1K long words are read from the data buffer. If the values received are either the second pattern or an unrecognized pattern, an error is reported as occurring while caching data.

8.11.1.2 TEST 2 - CACHE INSTRUCTION TEST

This test checks the ability of the cache to cache instruction operands only. The test flushes the cache and then configures caching for data operands, enables reads and writes, and forces a set 0. A subroutine which sets all the CPU data registers is loaded to a specific location in memory. The reads and writes on the cache are then disabled, and another subroutine, identical to the first except that it clears the CPU data registers rather than setting them. When the subroutine is executed, whether it is run from the cache or from memory can be determined by checking the values in the data registers. If they are not set, then an error has occurred.

8.11.1.3 TEST 3 - DUAL SET CACHE TEST

This test uses two data buffers of 1K long words each. The test configures the cache by first flushing it, enabling data operands only and enabling reads and writes. It then writes two different data patterns in the data buffers. The buffers are then cached, the first in set 0 and the second in set 1. Reads and writes are then disabled, and two new data patterns are written to the buffers. The test then re-enables reads and writes, and reads data from both buffers. The test reports errors if the data received is not the expected data.

8.11.1.4 TEST 4 - LEAST RECENTLY USED TEST

This test checks the cache's Least Recently Used replacement logic. The test writes data to three data buffers and then caches the data. The first buffer is cached to set 0, the second to set 1, and the third, because of the least recently used algorithm, should be cached to set 0. The three buffers are then written with new data, so there are now five data patterns (three in memory and two in cache). With cache reads enabled, each of the data buffers is read. From the first data buffer data should come from memory, and from the second and third data should come from cache. Any discrepancies are reported as errors.

8.11.1.5 TEST 5 - READ/WRITE ENABLE TEST

This test verifies that the cache is read only when reads from cache are enabled, and is written only when writes to cache are enabled. A data pattern is first written to a data buffer with cache writes enabled, and so is cached. Reads and writes are then disabled, and another pattern is written to the data buffer. Reads and writes are re-enabled, and the data buffer is read and checked to come from the cache. Again the cache is disabled and the data buffer is read, this time checked that it is the data in memory. Reads are enabled again and the buffer read to make sure the data still comes from cache. Then, with writes disabled, new data is written to the buffer, reread and checked to still come from cache (reads are still disabled). Reads are then disabled, and the buffer is read and checked to come from memory. Errors at any point are reported.

8.11.1.6 TEST 6 - INJECT TAG WORD PARITY ERROR TEST

This test verifies proper operation of the Tag Word Parity Bit setting logic, and proper operation when a set parity bit is encountered. During test setup, a level 3 interrupt vector processing routine is provided. When the data buffer is first written and cached, it is written with the TAGTEST bit set. Writes to cache are then disabled and a new pattern is written to the buffer. Reads and writes are then re-enabled and the cache is enabled to send a level 3 interrupt. The data buffer is then read, causing a Tag Word Parity error interrupt to occur. If the signal is not received from the interrupt routine, an error is reported.

8.11.1.7 TEST 7 - INJECT DATA WORD PARITY ERROR TEST

This test is identical to test 6, except that the DCTEST bit is set in place of the TAGTEST bit.

8.11.1.8 TEST 8 - INTERRUPT DISABLE TEST

This test repeats the checks performed in tests 6 and 7, except that level 3 interrupts are disabled. The test reports an error if it received an interrupt.

8.11.1.9 TEST 9 - CACHE DMA WRITE-HIT REPLACE TEST

This test checks that data is properly cached when a DMA write-hit occurs. A write-hit occurs when the cache updates the data for a valid address that it already contains. A DMA device is initialized with a 1 KB data pattern. With cache reads and writes enabled, a data buffer is then written with a second data pattern. The DMA device is instructed to transfer its data to the buffer, causing a write-hit and updating the cache with the first data pattern. Writes are then disabled, and a third data pattern is written to the buffer. With reads enabled, the test verifies that the first data pattern is in the cache.

8.11.1.10 TEST 10 - CACHE DMA WRITE-MISS REPLACE TEST

This test checks that data is properly cached when a DMA write-miss occurs. A write-miss occurs when a data operand is active on the bus with an address which has not previously been cached, and so the cache should not be updated. A DMA device is initialized with a data pattern. With cache reads and writes enabled, the DMA device is commanded to transfer its data to a data buffer. Cache reads and writes are then disabled, and a second data pattern is written to the buffer. The buffer should now contain the second pattern, but the cache should not contain either the first or second patterns. Reads are then enabled, and the cache is checked to make sure that it did not update during the write-miss. If the first pattern is found in the cache, the error is reported.

8.11.1.11 TEST 11 - CACHE DMA-MATCH TEST

This test verifies that eh CMB Cache does not update when a Match-Read occurs. A match-read occurs when a DMA device is written with data addressed at the same address as data in the cache. With cache writes enabled, the test writes a 1 KB data pattern to the data buffer and so to the cache. Reads and writes are then disabled, and a second pattern is written to the buffer. Reads and writes are re-enabled, and the DMA device is written to from the data buffer. Cache reads and writes are again disabled, and the DMA device writes to a second data buffer. The second buffer is checked to contain the second data pattern. If it contains the first, an error is reported.

8.11.1.12 TEST 12 - RETRY CACHE DISABLED TEST

This test verifies the CMB Cache does a retry when it encounters a parity error. A 1K long word data pattern is written to a buffer with the TAGTEST bit on, thus caching the first data pattern with parity errors. Reads and writes are then disabled and a second data pattern is written to the data buffer. Reads and Writes are then re-enabled and the data buffer is read. The cache gets a read-hit, but because the TAGTEST forces the parity bit on, the cache execute the parity error routine. Level 3 interrupts are disabled for this test, so the cache should perform a retry, causing the data to be read from memory rather than cache. If the data is the first data pattern, i.e., from the cache, an error is reported.

8.11.1.13 TEST 13 - CACHE FLUSH TEST

This routine verifies that the CMB cache is cleared when it is flushed. A data pattern is written to a buffer with cache reads and writes enabled, so the cache is written. Reads and writes are then disabled, and a second data pattern is written to the data buffer. The cache is flushed, and then reads and writes are re-enabled. The data buffer is then read, and the second data pattern is verified. If the first data pattern was read, then the cache was not flushed and the error is reported.

8.11.2 Cache Test Error Reporting

Errors reported by the Cache Test Module are of two types. Type 1 errors occur during the test setup phase, making it impossible for the test to proceed. Type 2 errors are data compare errors, indicating that the test ran but the cache function failed the test.

When an error occurs, CACHE always outputs the error number to the LEDs as a two hex digit code. The first digit is a 6, indicating that the CACHE module is running. The second digit is the error code itself. When the sense switches are set for running a single CACHE test, this is the only output from the system; otherwise, output is also sent to the VDT.

- Error 61

  This error occurs in test 1, Cache Data Test. The VDT displays one of the following messages:

  Cache Data: Data not being cached/loaded into mem correctly

  This indicates a type 1 error, that the cache data buffer is not reading or writing the data pattern correctly.

  Cache Data: Exp: 1A1A1A1A Rec: xxxxxxxx Word:XXX

  This indicates a type 2 error, that the cache either did not cache the data pattern correctly or did not get a correct read-hit.

- Error 62

  This error occurs in test 2, Cache Instruction Test, and is a type 2 error. The VDT displays the following message:

Cache Instruction:	Reg	Expected	Received
	DX	FFFFFFFF	XXXXXXXX

  This error occurs if the cache either did not cache the subroutine correctly or did not get correct read-hits on the instruction fetches.

- Error 63

  This error occurs in test 3, Dual Set Cache Test, and is a type 2 error. The VDT displays the following message:

  ```
  DSC    DSC1: Exp:1A1A1A1A   Rec:XXXXXXXX
         DSC2: Exp:2A2A2A2A   Rec:YYYYYYYY Wrd:ZZZ
  ```

  This error occurs if the cache did not cache the correct data value into the correct set. The message indicates the set in which the error occurred.

- Error 64

  This error occurs in test 4, Least Recently Used Test, and is a type 2 error. The VDT displays the following message:

  ```
  Cache LRU: SSS: Exp: XXXXXXXX        Rec:YYYYYYYY Word: ZZZ
  ```

  This message indicates that an error occurred in the least recently used logic. The message indicates in which set and data buffer the error occurred.

- Error 65

  This error occurs in test 5, Read/Write Enable Test, and is a type 2 error. The VDT displays the following message:

  ```
  Cache RWE: SSS: Exp: XXXXXXXX        Rec: YYYYYYYY        Word: ZZZ
  ```

  This message indicates an error during a write to or a read from the cache. The message indicates the buffer, the expected data and the data actually received.

- Error 66

  This error occurs in test 6, Inject Tag Word Parity Error Test, and is a type 2 error. The VDT displays one of the following messages:

  ```
  No interrupt serv on forced Tag Parity error: Word: XXX
  ```

  This message indicates that the cache did not interrupt the CPU as required when a parity error occurred in the Tag Word.

  ```
  PEX bit not set on forced Tag Parity error.  Word: XXX
  ```

  This message indicates that the proper bits were not set signaling the tag word in error. The message indicates which tag word parity bit is in error.

- Error 67

  This error occurs in test 7, Inject Data Word Parity Error Test, and is a type 2 error. The VDT displays one of the following messages:

  No int serv on forced Data Parity error. Word: XXX

  This message indicates that the cache did not interrupt the CPU as required when a parity error occurred in the data word.

  DCPEX bit not set. Word: XXX

  This message indicates that a Cache Data Word parity error interrupt occurred but failed to set the proper bits signaling the data word in error. The message indicates the byte of the data word for which the parity bit is in error.

- Error 68

  This error occurs in test 8, Interrupt Disable Test, and is a type 2 error. The VDT displays one of the following messages:

  Cache forced level 3 interrupt with interrupts inhibited. Word: XXX

  This message indicates that the CPU received a level 3 interrupt on a forced parity error, even though interrupts were inhibited.

  PEX bit not set: Word: XXX

  This message indicates that the proper Tag Word parity bits were not set when a parity error was forced in the cache with interrupts inhibited.

  DCPEO bit not set: Word XXX

  This message indicates that the proper Data Word parity bits were not set when parity error were forced into the cache with interrupts inhibited.

- Error 69

  This error occurs in test 9, Cache DMA Write-Hit Replace Test, and is a type 2 error. The VDT displays the following message:

  Cache DMA: write-hit: Word: XXX Exp: 1B1B1B1B Rec: XXXXXXXX

  This message indicates that the cache did not update correctly when the DMA was loading data into a data buffer which has been previously cached.

- Error 6A

  This error occurs in test 10, Cache DMA Write-Miss Replace Test, and is a type 2 error. The VDT displays the following message:

  Cache DMA: Write-miss: Word: XXX Exp: 1A1A1A1A Rec: XXXXXXXX

  This message indicates that the data cached when the DMA device loaded data into a data buffer which has not been previously cached.

- Error 6B

  This error occurs in test 11, Cache DMA-Match Read Test, and is a type 2 error. The VDT displays the following message:

  Cache DMA: Read hit: Word: XXX Exp: 1B1B1B1B Rec: XXXXXXXX

  This message indicates that the data cached when the DMA device was being loaded with data.

- Error 6C

  This error occurs in test 12, Retry-Cache Disabled Test, and is a type 2 error. The VDT displays one of the following message:

  Cache Retry: XXXX-Retry not performed. Cache not disabled: Word: YYY

  This error indicates that the wrong data value was returned when a retry was performed. The first parameter specifies whether the retry was performed on a Tag or Data word forced parity error.

  Cache Retry: Cache no Op after Clear: Word: XXX

  This message indicates that the cache would not run properly after being cleared following a forced parity error.

- Error 6D

  This error occurs in test 13, Cache Flush Test, and is a type 2 error. The VDT displays the following message:

  Cache Flush: Received: XXXXXXXX

  This message indicates that the cache would not run properly after being flushed. The received value is the value in the cache after being flushed.

- Error 6F

  This error may occur while initializing any CACHE test, and is a type 1 error. The VDT displays the following message:

  Data not being cached/loaded into mem correctly - Rerun Data Caching Test

  This message indicates that the cache is not performing the basic data caching required to set up the test conditions. If this error occurs, run the Cache Data Test, test 1.

8.12 CONTROLLERS TEST MODULE (<ctlr>)

The Controllers Test Module tests the 1/2 inch Tape Streamer (TS), 1/4 inch Cartridge Streamer (CS), Local Area Network (LAN), Four-way (FW) and Eight-way (EW) controllers. All of these tests are executed during the power-on self tests for the controllers that are installed, or can be individually selected by command or sense switch settings. If more than one board of a single type is present, all of the boards of that type are tested.

Refer to Section 8.2.2 for instructions on starting CTLR by sense switch setting. To begin CTLR from the <mds> prompt or another test prompt, enter:

```
ctlr
```

The test module then starts, and the <ctlr> prompt is displayed.

8.12.1 CTLR Test Descriptions

8.12.1.1 TEST 1 - CS SELF-TEST REPORT

This test executes a reset to the CS controller, which causes the self-test in the CS firmware to run and returns the results to a register. An error 76 is reported if an error occurs. The CS self-test status register is a word in length, but only bits 15-8 are used as follows:

00	No error
10	RAM
20	ROM (check sum)
30	CTC
40	FIFO
50	Control/Status Register

8.12.1.2 TEST 2 - TS DATA REG

This test writes, reads and compares all byte values 00 to FF to the TS controller data register. If a mismatch occurs, an error 7D is reported. After all 256 bytes have been written and read, the controller parity error register is checked to verify that a parity error did occur, and if not an error 7E is reported.

8.12.1.3 TEST 3 - TS VECTOR REG

This test writes, reads and compares all byte values 00 to FF to the TS controller vector register. If a mismatch occurs, an error 7C is reported.

8.12.1.4 TEST 4 - TS RESET

This test resets the TS controller, and then verifies that the status register of the board is zero. If a different status is returned, an error 7F is reported

8.12.1.5 TEST 5 - 4-WAY SELF-TEST REPORT

This test issues a reset command to all 4-way boards, causing each to perform a self-test and return the results. If any errors are reported, they are reported.

Errors returned are as follow:

77	Timeout waiting for self-test to complete
79	RAM test failed
7A	Fatal error
7B	Internal loopback test failure

8.12.1.6 TEST 6 - 8-WAY SELF-TEST REPORT

This test issues a reset command to all 8-way boards, causing each to perform a self-test and return the results. If any errors are reported, they are reported.

Errors returned are as follow:

77	Timeout waiting for self-test to complete
78	ROM checksum error
79	RAM test failed
7A	Fatal error
7B	Internal loopback test failure

8.12.1.7 TEST 7 - LAN TEST

This test first writes the vector number to the LAN vector register, then reads and compares the value. If a mismatch occurs, and error 72 is reported. Next, a zero is written to the LAN control register, then it is read and compared. If a mismatch occurs, an error 71 is reported. A 99 hex is the written to the LAN control register, read and compared. If a mismatch occurs, an error 73 is reported. A WHO command is then issued to the LAN. An error 74 is reported if the board fails to interrupt, or an error 75 if the node value DMA'ed by the board doesn't match the value in its nar register.

Following a successful completion of these tests, the led on the controller will light, indicating that the board passed. These tests are performed on all LAN controllers present.

8.12.2 CTLR Test Error Reporting

When an error occurs, CTLR always outputs the error number to the LEDs as a two hex digit code. The first digit is a 7, indicating that the CTRL module is running. The second digit is the error code itself. When the sense switches are set for running a single CTRL test, this is the only output from the system; otherwise, output is also sent to the VDT.

The error codes are grouped as follows:

71-75	LAN
76	CS
77-7B	4-way or 8-way
7C-7F	TS

- Error 71

  This error is reported by test 7. The VDT display is:

  LAN Writing 0 to the Control Reg Failed to Clear It

  The module number of 0 or 1 is also displayed, indicating which controller failed.

- Error 72

  This error is reported by test 7. The VDT display is:

  LAN Interrupt Reg Write/Read Err

  This error indicates that the data read in the LAN vector register is not what was written. The module number of 0 or 1 is also displayed, indicating which controller failed.

- Error 73

  This error is reported by test 7. The VDT display is:

  LAN ctl Reg wr/rd Compare Err

  This error indicates that a 99 hex was written to the LAN control register, but was not read. The module number of 0 or 1 is also displayed, indicating which controller failed.

- Error 74

  This error is reported by test 7. The VDT display is:

  Execution of a LAN "Who" CMD Failed to Interrupt

  This error indicates that an interrupt was not received from the LAN controller after sending it a WHO command. The module number of 0 or 1 is also displayed, indicating which controller failed.

- Error 75

  This error is reported by test 7. The VDT display is:

  A 'WHO' CMD Gave a Nod Addr Which is Different From Reading the Reg Directly

  The message indicates that the test has received an interrupt from the WHO command, but the node address that was DMA'ed is incorrect. The module number of 0 or 1 is also displayed, indicating which controller failed.

- Error 76

  This error is reported by test 1, when the CS controller reports an error after the board has been reset. The VDT display is:

  CS Selftest Failure Code: X

  Refer to the test description for the failure codes.

- Error 77

  This error is reported by test 5 when the self-test bit of the 4-way status register remains ON and the fatal error bit is OFF. It is also reported by test 6 when either the busy bit of the 8-way status register remains ON, or the self-test bit remains OFF (active low). The VDT message is:

  Timeout Waiting for Module n to Finish the Selftest, Status=xx

  Refer to the test descriptions.

- Error 78

  This error is reported by test 6 when the 8-way self-test reports a ROM checksum error. The VDT message is:

  Module n Failed the ROM Checksum Test

  Refer to the test description.

- Error 79

  This error is reported by tests 5 and six when a 4-way or 8-way board has failed its RAM test. The VDT message is:

  Module n Failed the RAM Test

  The module number indicates the board that failed.

- Error 7A

  This error is reported by tests 5 and 6 when a 4-way or 8-way board has reported a fatal error. The VDT message is:

  Module n Received a Fatal Error, Status Reg=nn

  The module number indicates the board that failed.

- Error 7B

  This error is reported by test 5 and 6 when a 4-way or 8-way board reports that its internal loopback test has failed. The VDT message is:

  Module n Looopback Test Failure Code: nn

  The module number indicates the board that failed. The least significant 4 bits of the code indicate the port that failed.

- Error 7C

  This error is reported by test 3 when the byte that was written to the TS vector register was not the same as was read. The VDT message is:

  TS Vector Reg Miscompare. Wrote: nn Read: mm

  The hexadecimal values written and read are shown.

- Error 7D

  This error is reported by test 2 when the byte that was written to the TS data register was not the same as was read. The VDT message is:

  TS Data Reg Miscompare. Wrote: nn Read: mm

  The hexadecimal values written and read are shown.

- Error 7E

  This error is reported by test two when the TS controller parity error register did not indicate that a parity error occurred after the data register write, read, compare portion of the test completed. The VDT message is:

  TS Parity Err Expected but Did Not Occur

  Refer to the test description.

- Error 7F

  This error is reported by test 4 when the controller status register is not zero after the controller has been reset. The VDT message is:

  TS Status Reg Not Cleared After Reset

SECTION 9
DISK UTILITY

9.1 OVERVIEW

The Disk Utility diagnostic, DUTIL, is primarily used for preparing a fixed disk for data storage. This procedure includes performing a disk surface analysis, mapping out bad blocks, formatting the disk, and related procedures.

The procedure for using DUTIL to prepare a fixed disk is described in the service documentation for the MAI 2000/2500/3000/4000 systems and in the BOSS/IX Technical Reference Manual (M6227). This section is intended only to supplement those procedural descriptions by providing descriptions of the full set of commands available through DUTIL.

DUTIL operates under the control of the bootable diagnostic Executive, as described in Section 4, just like SIT, the Logic tests and the Function Select tests. DUTIL is primarily a command driven program, and so is a Function Select type test, enhanced with a menu driven mode of operation.

9.2 STARTING DUTIL

9.2.1 Load Diagnostic Executive

DUTIL runs under control of the diagnostic Executive. Refer to Section 4 for instructions on performing an alternate load to the Diagnostic Executive.

9.2.2 Loading DUTIL

Once the diagnostics executive has been loaded, you are ready to load the DUTIL program. At the diagnostics executive prompt type:

LOAD DUTIL

in either upper or lower case, and then press **ENTER**.

The Executive then loads and initializes the program. DUTIL is now ready to accept commands.

9.3 COMMAND DESCRIPTIONS

The DUTIL command descriptions are broken up into several sets, as follows:

- Preliminary Commands
- Buffer Commands
- Parameter Commands
- Initialization Commands
- Control Commands
- Immediate Commands
- Execute Commands
- Multi-parameter Execute Commands
- High Level Commands
- Conversion Commands
- Superblock Commands
- Support and Inspection Commands
- Block Commands

9.3.1 Preliminary Commands

REVISION

This command displays the program title and revision level.

DUTIL

This command causes DUTIL to enter menu mode of operation. The controller must be selected using the WDC command prior to entering menu mode.

WDC {*n*}

This command selects the disk controller to use in subsequent operations. If no argument is given, the number of the current drive is displayed. Legal controller parameters are 0-3.

LUNIT {*n*}

This command selects the logical unit to test. The legal logical unit parameters are 0-7. The program selects the correct physical controller and drive corresponding to the logical unit.

INQUIRE

This command sizes and displays the disk configuration of the system. The display is the same as that displayed when the diagnostic is first loaded. The self-test is run for each controller found. If the self-test passes, "OK" is displayed for that controller. If a controller does not pass its self-test, "ERROR" is displayed for that controller, in which case the controller should be replaced.

ADDRESS {*n*}

This command specifies the controller address. There is seldom a reason to do this. The default controller address is CC0000. If the command is entered without an argument, the controller address is displayed.

9.3.2 Buffer Commands

CREATE {*n*}

This command creates a block of data to be used for the next write or compare operation. The parameter n specifies the data type. The data types are the same as those shown in Table 7.2 for the DISKFS Function Select test. If no parameter is specified, a block of data is created using the last data type, and is then incremented to the next type for each subsequent **CREATE** command.

WDISPLAY {*n*}

This command displays the contents of the write buffer. The argument, if present, specifies the number of words to display. If no argument is given, a page of the write buffer is displayed (256 bytes).

RDISPLAY {*n*}

This command displays the contents of the read buffer. The argument, if present, specifies the number of words to display. If no argument is given, a page of the read buffer is displayed (256 bytes).

COMPARE {*n*}

This command compares the read buffer with the write buffer. The optional argument specifies the number of words (in hex) in the buffers to compare. If the argument is greater than the amount of data indicated by the **#SECTORS** command, the lesser amount is compared. If no argument is given, the amount specified by **#SECTORS** is used.

SWAPBUFF

This command swaps the pointers to the write and read buffers. This allows reading the data into the read buffer, swapping pointers, and writing the same data back to the same or another drive.

9.3.3 Parameter Commands

Parameter commands do not cause any action to occur, but set default values for other commands.

HEAD {*n*}

This command selects the drive head (0-FF) to be used for the next Execute command. This is the logical head for starting the next transfer, and remains in effect until another **HEAD** command or the **FORHEAD** command is encountered. HEAD also cancels any FORHEAD loop by zeroing the head count. If an argument is not given, the currently selected head is displayed.

SECTOR {*n*}

This command selects the logical sector (0-FF) to use to start the next Execute command. If an argument is not given, the currently selected sector is displayed.

#SECTORS {*n*}

This command specifies the number of sectors (0-FF) to transfer. The default value is 1. If an argument is not given, the currently selected number of sectors is displayed.

GRCYL {*x*}

This command generates a random cylinder number to be used for the next seek operation. The optional argument specifies the number of times the seek is to be performed, generating a new cylinder number for each seek. The range of cylinder available for the seek is 0 to the value set by the **MCYL** command.

UNIT *n*

This command specifies the drive unit for the next operation. The unit number is either 0 or 1.

PARAMETER

This command displays the currently active parameters.

DPATTERN {*n*}

This command specifies the one byte data pattern to be used for the next FORMAT command. The default value is E5 hex, and the legal range is 0-FF hex. If the parameter is not given, the current data pattern is displayed.

PRECOMP {*n*}

This command specifies the write precomp cylinder (0-FFFF hex) to be used in the next formatting operation. If the argument is not given, the current value is displayed.

INTERLV {*n*}

This command specifies the sector interleave factor used in formatting the disk. An argument value of 0 causes the Adaptec controller to default to a factor of 2. A factor of 1 is the usual default. A factor of 2 or more causes the controller to provide an extra sector per track. Changing the interleave factor from the default may cause degradation of performance. Legal values are 0-FFFF hex. If no argument is given, the current value is displayed.

FORHEAD {*b*{,*e*}}

This command specifies a range of heads to use for the following multiple head operation. If no arguments are given, a range from 0 to the maximum head (**MHEAD**). If both parameters are specified, b is the beginning head, e is the ending head. The ending head must be less than the maximum head and greater than the beginning head.

FORCYL {*b*{,*e*}}

This command specifies a range of cylinders to use for the following multiple cylinder operation. If no arguments are given, a range from 0 to the maximum cylinder (**MCYL**). If both parameters are specified, b is the beginning cylinder, e is the ending cylinder. The ending cylinder must be less than the maximum cylinder and greater than the beginning cylinder.

DISK

This command specifies that the next command will cause a range of cylinders and heads to take effect. The range is from head 1 and cylinder 0 to the maximum head (**MHEAD**) and maximum cylinder (**MCYL**).

9.3.4 Initialization Commands

Initialization commands use a single parameter, *n*. If the parameter is not given, the current parameter value is displayed.

SIZE *n*

This command selects and displays the sector byte count used for formatting the disk. The default is 200 hex. The legal range is 0-FFFF hex.

MICROPOL {*n*}

This command causes the program to select Micropolis drive defaults. If the argument is omitted, the program assumes the 50 MB defaults (6 heads, 32E hex cylinders and reduced write current cylinder of 190 hex).

If *n* = 50, select 50 MB defaults.
If *n* = 85, select 85 MB defaults.

MAXTOR {*n*}

This command causes the program to select Maxtor drive defaults. If the argument is omitted, the program assumes the 140 MB defaults.

If *n* = 105, select 105 MB defaults.
If *n* = 140, select 140 MB defaults.
If *n* = 190, select 190 MB defaults.

RODIME {*n*}

This command causes the program to select Rodime drive defaults. If the argument is omitted, the program assumes the 20 MB defaults.

If *n* = 20, select 20 MB defaults.
If *n* = 40, select 40 MB defaults.
If *n* = 53, select 53 MB defaults.

9.3.5 Control Commands

Control commands cause no operation to be performed. These commands control the operation of the diagnostic and the controller operation. They are also used to check the results of the last operation.

RETRY *d*

This command sets the retry count for the **SURFACE** and **WRCOMPARE** commands. The count must be a decimal number in the range 0-255.

READONLY

This command prevents most writes to the disk by diagnostic commands. This is the default condition for this diagnostic.

WRITEOK

This command disables write protection.

9.3.6 Immediate Commands

Immediate commands cause an immediate operation on the controller or the last drive selected by the UNIT command. Many of these commands are low level commands, so the operator must understand the overall operation of the controller.

CCONTROL

This command clears the entire controller board by a SASI reset.

RCAL {*u*}

This command recalibrates drive *u*. If an argument is present, it specifies the unit which then remains selected for subsequent operations. If no unit is specified, the last unit selected is recalibrated.

WDATA *n*

This command writes the byte argument to the output register.

RDATA

This command reads a byte from the input register and displays the contents.

STATUS {*n*}

This command, without an argument, reads the adapter status and displays the result. With an argument, the command reads the status and waits for the status to match the argument. The command will wait indefinitely, but may be interrupted by **CTRL + C**.

ECC

This command forces the controller to do ECC checking of data, and enables controller retries. This is the default for this diagnostic.

NOECC

This command disables error correction of data and retries by the controller. This command should be entered prior to doing a disk surface analysis.

SELECT

This command selects the SASI controller. After selection, the program waits until the controller is ready for the DCB bytes, indicated by the controller returning a C2 hex status. The select command must precede and controller command while in Immediate mode.

9.3.7 Execute Commands

USTATUS {*u*}

This command issues the Test Unit Ready command and checks for errors. The argument specifies the test unit, and remains in effect for subsequent commands. If no argument is given, the last unit selected is used.

SEEK {*n*}

This command causes the drive to seek to the specified cylinder. If no cylinder is specified, the seek is to the cylinder specified by the last **SEEK**, **CYLINDER** or **GRCYL** command.

SPECIFY

This command issues the Mode Select command to specify the attached drive characteristics. The required parameters must have been specified previously. The diagnostic assumes drive definition default values if the user has not specified then otherwise. The required parameters are: sector size (**SIZE**), physical number of cylinders and the reduced write current cylinder, the number of heads per cylinder, the number of sectors per track, and the write precomp cylinder (**PRECOMP**).

SENSE {*u*}

This command issues the Mode Sense command. If there is an error indicated by the check bit in the sense byte, it is displayed. If the valid bit is set, the cylinder, head and sector where the error occurred is also displayed. If the unit argument is specified, the sense command is issued to that unit and the unit remains in effect. If the check bit is not set, an error message is displayed with error code 0 (no error).

MODESENS

This command issues the Mode Sense command and displays the data received.

LOGOUT

This command displays the current list of error detected by the controller.

CLRLOG

This command clears the error log.

9.3.8 Multiple Parameter Execute Commands

The following Execute commands require that parameters have been set using the Parameter commands. Which parameters must be set depends on the command.

The unit argument *u* specifies the test unit. If the unit is not specified, the last unit selected is used.

READ {*u*}

This command reads data from the disk into the read buffer.

WRITE {*u*}

This command writes data to the disk from the write buffer.

VERIFY {*u*}

This command issues the Verify Data command, to verify the ECC of the last data written to the disk.

TRANSLAT {*u*}

This command issues the Translate command and displays how the controller translates the logical sector. The value returned specifies the physical location of the logical sector.

9.3.9 High Level Commands

FORMAT {*u*}

This command prompts for entry of disk defaults and then formats the disk. The argument specifies the disk unit (0 or 1), which remains the current unit following the format. If the argument is not specified, the current unit is formatted. Before prompting for defects, the command displays the results of the previous analysis, if any.

The format can be done first with no defects entered, with an empty defect table. This can be followed by a disk surface analysis, and then a reformat entering any defects found. Defects are also frequently included in a list provided by the manufacturer.

DEFORMAT {*u*}

This command performs a disk format using a defect track list. This requires that the operator enter the specific defect list into the write buffer prior to doing the **DEFORMAT** command.

DEFECTS

This command first displays the results from the last disk surface analysis. It then prompts for entry of disk defects. All defects entered are then sorted with the previous defect table and placed in the write buffer for the next format procedure. This command prepares a defect list for the DEFORMAT command.

SURFACE {*u*}

This command performs a surface analysis of the disk unit specified. The procedure overwrites all data previously on the disk. The disk is checked from cylinder 0 to MCYL.

Failing tracks are stored in an error track table. Upon completion, the failing tracks are displayed in a table format. If more than 128 bad tracks are located, the surface analysis is aborted and the errors are displayed.

The retry count set by the **RETRY** command is used by this command. If an error is detected and the retry count is not exhausted, the operation is retried. Only when the retry count is exhausted is the failing track put into the error table.

COPYDISK {*n*}

This command copies one entire disk to another. If no argument is specified, the copy is from unit 0 to unit 1. If an argument is specified, the specified disk is the source disk, and the other unit is the destination disk.

WRCOMPAR

This command writes data from the write buffer to the disk, reads the data back into the read buffer and compares the buffers. The **RETRY** count is used in case of miscompares. The hex pattern BAD1 is written to the read buffer prior to performing the read operation.

REF_SURF

This command reformats the disk using surface analysis results. Note that the original defects used to format the disk are not included. The purpose of this command is to format a disk when the manufacturing defect list is unknown or when there are no manufacturing defects.

DEFAULTS

This command initializes the controller by issuing the Clear Controller command, and then loading the vector register and the DMA register. It then restores all the default parameters (Rodime 20) and selects unit 0 and Auto mode.

RESTART

This command initializes the diagnostic by issuing the recalibrate command and resetting necessary flags.

INIT

This command initializes the diagnostic.

9.3.10 Conversion Commands

HEX *n*

This command displays the decimal equivalent of the hexadecimal number *n*.

DEC *n*

This command displays the hexadecimal equivalent of the decimal number *n*.

9.3.11 Superblock Commands

The following commands should only be used to repair or display the superblock. The superblock is normally written automatically following the formatting process.

SBST

This command displays the superblock structure as it is contained in memory. The memory address where the structure is contained in memory is displayed so patches can be made in memory.

DVBLKS

This command calculates and displays the capacity of the disk based on current disk parameters. The device capacity is saved in the disk superblock.

WSB

This command writes the superblock information in memory to block 4 on disk. This command is potentially destructive to disk data, if the data written is incorrect.

RSB

This command reads the superblock into the read buffer. If the data is valid, it is transferred into the superblock structure in memory.

DSB

This command reads and displays the contents of logical sector 4, where the superblock resides.

9.3.12 Support and Inspection Commands

DVERIFY

This command verifies the entire disk. The VERIFY command is issued for every track on the disk, checking the ECC codes for data fields. All errors encountered are displayed at the end of the procedure.

DREAD

This command reads the entire contents of the disk. Every track on the disk is read to check the ECC codes for data fields. All errors encountered are displayed at the end of the procedure.

TABLE

This command displays the error table created by the other commands.

CONVERT

This command converts the contents of the error table from the logical locations to physical locations. The cylinder and head values are changed from hexadecimal to decimal. The sector entry is converted to a "bytes from index" value.

9.3.13 Block Commands

These commands can be used to verify or recreated errors detected by the O.S. The default argument entry for these commands is decimal. To enter the argument in hexadecimal, precede it with "h" or "'".

BLOCK *n*

This command displays the contents of a logical block on disk, specified by *n* (decimal). Once a block has been displayed, the +BLOCK and -BLOCK commands can be used.

+BLOCK

This command displays the next logical block on disk.

-BLOCK

This command displays the previous logical block on disk.

SETBLOCK {*n*}

This command specifies a logical block number for the next command. The block is specified in decimal. If the argument is not given, the command displays the block number for the next disk operation.

GET {*n*}

This command reads the logical block on disk specified by *n* (in decimal) and puts the data into the write buffer. To display the data that was read, use the **WDISPLAY** command.

PUT {*n*}

This command writes data to a logical block on disk. The optional argument n gives the block number to write. If the argument is not specified, the data is written to the block used in the last **GET** command. If the block number is specified, that block is written.

MBLOCK {*n*}

This command specifies the maximum block to **GET** or **PUT** prior to the **IBLOCK** command. If the block is not specified, the highest sector available is used, which is the entire disk including the diagnostic cylinder.

IBLOCK {*n*}

This command reads and writes the disk over a selected range. The optional argument specifies the number of blocks to read and increment after each read. The command causes the range of blocks specified by the **GET** and **MBLOCK** commands to be read and written if the **PUT** command was entered before the **IBLOCK** command.

To specify the range of blocks, first use the **MBLOCK** command to specify the last block to read, then the **GET** command to specify the sting block number. Then use the **PUT** command (without argument) to specify a **GET** or **PUT** function. Finally, issue the **IBLOCK** command and specify the amount to read each pass. The loop is automatically executed over the range of disk.

EXTRAP *n*

This command translates a decimal block number, and returns the cylinder, head and bytes-from-index (BFI) for the logical blocks 3 before and 3 after the specified block. The command can be used to obtain the physical locations of a block especially when the preceding and following blocks are not physically adjacent to the defective block.

9.4 ERROR REPORTING

9.4.1 Unexpected Status Errors

9.4.1.1 BAD STATUS BEFORE COMMAND ISSUE

- Adapter not ready prior to issuing command

  Either the controller was busy prior to issuing the command, or the command caused the controller to stay busy after the command was issued. The operation flag displayed with this error is the DCB opcode.

- Unit write protected

  The write protect flag in DUTIL memory is set. The write operation requested cannot be completed.

- Bad status after SASI selected

  The adapter status should indicate selected and busy condition and the output register free. After the SELECT command is issued, the adapter status of C2 is expected, and a timeout occurred while waiting for this status.

- DCB byte transfer timeout

  The bytes which make up the DCB are sent to the controller in simple output mode. Correct status is checked prior to sending each byte. A timeout occurred waiting for correct status of C2 to transfer the next DCB byte.

- Data transfer timeout

  If the operation which transfers disk data in non-DMA mode, or if the command itself requires sending data associated with the command in simple I/O, status must be correct before the data is transferred. The correct status was not obtained within the allotted time.

- Bad status found before command completion

  After the command is issued, the program waits for an interrupt and check status, or waits for correct ending status from the adapter. The expected status of CC was not obtained, indicating an error condition of some sort.

9.4.1.2 BAD STATUS AFTER COMMAND ISSUE

- Expected bus error status not found

  The program attempts to get an interrupt from the bus error status. After trying to force the bus error, the status bit did not set.

- Unexpected Adapter detected address bus error

  While the adapter is doing a DMA transfer, the adapter detected a bus error.

- Adapter detected bus error didn't generate interrupt

  The adapter has the bus error bit set in status, but it did not generate an interrupt as expected.

- Can't obtain the ending status

  Because the command completion status was not obtained, the ending status byte was not read from the SASI controller for ending status error checking.

- Message phase timeout

  After the ending status byte is read from the controller, it should indicate message phase with a status of E8.

- Message byte not zero

  The actual message byte read from the controller while in message phase should be 0.

- Error processing: SENSE command failure

  The previous operation had the check bit set in the ending status byte indicating that a SENSE command was necessary. The SENSE command is automatically issued in this state. This error indicates that the SENSE command failed to complete correctly.

- Position error

  After the user has selected seek verification by the SVERIFY command, the translate command issued by the program indicated that there is a difference between the physical position requested and the actual position that the SASI controller indicates.

- With interrupts enabled, no interrupt occurred.

  After the command was issued, the controller did not generate the expected interrupt within the allotted time.

- With interrupts disabled, an interrupt occurred.

  The program detected an interrupt even though the controller was not expected to generate one.

- Non-zero status after reset

  The controller status should be equal to 0 when the clear controller command is issued.

- Ending status non-zero

  The command completion status byte read from the controller after the command completes is not 0. The ending status obtained from the controller is displayed along with the command code. The command code is the DCB opcode byte.

- Command aborted before completion

  The program found operation status bytes in error while trying to check the results of the operation. This normally occurs when commands are entered out of sequence, or if the WAIT command is not entered while in IMMEDIATE mode after a SASI controller command is issued. It also occurs if the operator hits ESCAPE during command processing.

- Unexpected sequence or program error

  Normally caused when DISKFS is not exercising the expected control over the result of the operation. The program should be reloaded in this case, since it is not operating as expected.

- Bad unit x status

  Bad unit status existed after the command. The unit, error code from the SENSE operation, logical sector address, and the cylinder, head and sector where the error occurred is displayed, as applicable.

- Cylinder exceeds maximum

  The cylinder requested is too large, or larger than the cylinder specified in the MCYL command arguments. This prevents the user from trying to seek to an invalid cylinder. The cylinder requested and the program maximum are displayed.

- Buffers don't compare

  The write buffer data is not the same as the read buffer data. The expected words and the received words are displayed, as well as the buffer addresses. This is normally caused by the write buffer being changed after a read operation and then doing the COMPARE command, or that the user did not seek to the cylinder that the CYLINDER parameter specifies.

  If the data in the read buffer is BAD1 hex, then the read operation probably did not occur. The read buffer is configured to the BAD1 pattern by the WRCOMPARE command prior to the read operation.

- Adapter address bus error

  The program received a bus error from a legal controller address.

- Controller detected error

  At the completion of a controller command, the ending status byte indicated an error. The results of the SENSE command are displayed with this error in the format: error code, command causing the error, the unit in error.

- Bus error status won't reset

  The command to reset the bus error bit in status did not cause the status to clear.

- Reset line status not set

  After issuing the SASI reset command, the status did not indicate SASI reset active.

- Device busy timeout after multiple retries

  If the controller indicates that it is busy by the ending status, the operation is retried automatically by the diagnostic. This error is displayed when the controller continually indicates busy status.

- WARNING: BAD SUPERBLOCK!

  The superblock does not contain valid data in order to do an auto-size.

9.4.2 Interpretive Messages

The following messages may accompany error messages to give additional error information.

- SASI reset active

  During error processing, the reset line status bit in status was found ON.

- Lost BUSY status

  During the processing of the command after the SASI controller selected, the busy status bit was found OFF.

- Command complete timeout

  The status of CC hex was not read by the program in the time allotted.

- Interrupt service routine hung

  When the RTE instruction is executed in the interrupt service routine, control should be returned to the diagnostic test flow. If the interrupt service routine is called 128 times before the diagnostic can reset the counter, this error is displayed.

- Invalid emulation instruction

  The diagnostic uses the emulation vector for certain operations. This error is displayed before a trap 15, when the emulation code is invalid. This normally means the program was corrupted in some, or there is a program error.

SECTION 10

SCSI DISK UTILITY

10.1 OVERVIEW

The SCSI Disk Formatter, FAVI, is primarily used for preparing a SCSI disk for data storage. This procedure includes performing a disk surface analysis, mapping out bad blocks, formatting the disk, and related procedures.

The procedure for using FAVI to prepare a fixed disk is described in the BOSS/IX Technical Reference Manual, M6227. This section is intended only to supplement that procedural description by providing a description of the full set of commands available in FAVI.

FAVI operates under the control of the bootable diagnostic Executive, as described in Section 4, just like SIT, the Logic tests and the Function Select tests. FAVI is primarily a command driven program, and so is a Function Select type test, enhanced with a menu driven mode of operation.

The FAVI commands are very similar to those for the DSCFS function select test.

10.2 STARTING FAVI

10.2.1 Load Diagnostic Executive

FAVI runs under control of the diagnostic Executive. Refer to Section 4 for instructions on performing an alternate load to the Diagnostic Executive.

10.2.2 Loading FAVI

Once the diagnostics executive has been loaded, you are ready to load the FAVI program. At the diagnostics executive prompt type:

LOAD FAVI

in either upper or lower case, and then press **ENTER**.

The Executive then loads and initializes the program. FAVI is now ready to accept commands.

10.3 FAVI OPERATIONS

FAVI starts running in menu mode. This provides an interactive interface for running several standard disk maintenance procedures. The disk formatting procedure is described in the BOSS/IX Technical Reference Manual, M6227. Three other disk testing procedures are also supported in menu mode: Analysis (10.3.1), Verification (10.3.2) and Inspection (10.3.3). Help text is available at most points in the menu system, and may be accessed by entering "?".

Menu mode operation begins with a login screen. Entering a login name is optional, and simply displays the name at the top of the screen while FAVI is running. The next screen asks you to select a SCSI unit to test/format. The drives available are identified by Dual SCSI Controller number, port (A or B), the device address, and a product identification. After selecting the device, you are prompted for the procedure to run.

FAVI also supports a command level operation, using the commands described in section 10.4. To access command level (terminate menu mode), press ESCAPE.

10.3.1 Analysis Test

The surface analysis test is used to test the disk surface and locate blocks on the disk which will produce errors under normal system usage. This is not necessary under normal conditions, since the testing performed by the manufacturer is more precise. It may be necessary, however, if a disk begins generating errors during use.

The surface analysis proceeds by writing one or more data patterns on the entire disk, and then reading back and verifying the data. During the process, all data on the disk is destroyed. You are asked to select the number of data patterns to use (1-23), selected from a list of data patterns. If you select to use only one pattern, you are also asked which pattern to use. For two or more patterns, the test selects tests in reverse order, starting with the worst case patterns.

Use the help text for additional information, and to display the data patterns.

10.3.2 Verification Test

The verification test is a non-destructive disk surface analyzer, and is usually run as part of the formatting process. Every cylinder of the disk is read and verified for data corruption. Any errors that are found are added to an error table. The table can be displayed at the end of the test, and you are given the option of reassigning the blocks in error to alternate blocks.

The reassignment process is performed by the disk itself, and includes moving the data to the alternate block, provided the error block can be accessed. The process normally completes with no data loss, unless there are disk access problems.

Use the help text for additional information.

10.3.3 Inspection Test

The inspection test is used to do non-destructive tests, such as reading a range of blocks, displaying actual disk data, and reading blocks of data and writing the same data back. A separate menu is displayed for selecting these functions.

Use the help text for additional information.

10.4 COMMAND DESCRIPTIONS

The FAVI command descriptions are broken up into several sets, as follows:

- Unit Selection Commands
- DSC Register Commands
- Write/Read SBIC Commands
- Program Control Commands
- Block Selection Commands
- Buffer Control Commands
- Superblock Commands
- SCSI Commands
- Special Function Commands

10.4.1 Unit Selection Commands

Before any other FAVI commands are executed, the DSC controller and port must be selected using the Unit Selection commands. These selections remain in effect until the commands are issued again.

DSC {*d*}

This command selects the DSC to test (d = 1-3). If no argument is given, the number of the currently selected DSC is displayed.

LPORT {*n*}

This command selects the logical port (0-F hex), and so performs the same function as the DSC and PORT commands together. The bits are as follows:

Bit	Meaning
0	0 = port a
	1 = port b
1,2	DSC boards 0-3
3	0 = main unit
	1 = expansion unit

PORT {*args*}

This command selects which port(s) to test on the DSC. The ports are designated as a and b, and both may be specified:

port a b

If no argument is specified, the currently selected ports are displayed.

SIZE

This command causes the system to execute the sizing operation, as it did when the diagnostic was loaded. The results are shown on the screen.

TARGET {*n*}

This command sets the destination ID register in the SBIC. If no argument is given, the current destination ID is displayed.

XDSC {*d*}

This command selects the expansion unit DSC to test (d = 1-3). If no argument is given, the number of the currently selected DSC is displayed. If there is no expansion unit, the currently selected main unit DSC is displayed.

10.4.2 DSC Register Commands

BERRC

This command strobes the Clear Bus Error/DMA Reset register of the currently selected DSC and Port.

BUSRESET

This command issues a SCSI bus reset to the DSC control register. The control register is then cleared.

CCHANNEL

This command first resets the SCSI bus and then resets the active DSC port.

CCONTROL

This command issues a DSC port reset to the active port.

REGS

This command displays the bit definitions of the DSC status and control registers. The contents of the status, control and vector registers are then displayed for the current DSC and Port.

RESET

This command issues a reset to the entire system. Following the reset, the system console device is reprogrammed.

STATUS {*n*}

With no argument, this command reads and displays the DSC Status register. The command may be looped, if no argument is given. If an argument is given (00-FF hex), the command causes the program to wait until this status is received, or until CTRL + C is entered.

10.4.3 Write/Read SBIC Commands

AUXSTAT

This command reads and displays the contents of the SBIC Status register.

OWNID {*n*}

This command loads or reads the SBIC OWNID register. If an argument is specified, this value is written to the register. If no argument is specified, the current value is read and displayed.

REGISTER *n* {*b*}

This command reads the specified SBIC register (1-19 hex). If the second argument is also specified, that data is written to the specified register address (in hex). This command can be looped.

SCSISTAT

This command is like REG 17, except that it interprets the status for easier identification.

10.4.4 Program Control Commands

KEYS

This command loads the terminal function keys with a set of commands defined for DSCFS. As the keys are programmed, the functions programmed into them are displayed.

LOOP {*n*}

This command causes loopable commands to repeat the specified number of times (0-7FFF hex). A value of 0 disables looping.

NOPROTECT

This command removes write protection the device during testing, thus allowing write operations. (See PROTECT.)

PARAMETER

This command displays the current program parameters and control selections.

PROTECT

This command write protects the device during testing. (See NOPROTECT.)

STEP (0|1)

This command causes the driver to use the SBIC step mode, if the argument is 1. When the argument is 0, combination mode is used. Combination mode is valid only when DMA data transfers are selected (see the DMA command).

10.4.5 Block Selection Commands

BLOCK {*d*}

This command selects the starting block for the next transfer. The block is specified in decimal, but may be specified in hexadecimal by preceding the number with "h".

#BLOCKS {*n*}

This command specifies the number of blocks to transfer. The block size is used to calculate the actual byte count for the transfer.

+BLOCK

This command increments the current block number by 1. When the maximum block number is exceeded, the block number will wrap around to the minimum block number. The new block number can then be used by the next command.

-BLOCK

This command decrements the current block number by 1. When the minimum block number is exceeded, the block number will wrap around to the maximum block number. The new block number can then be used by the next command.

GRBLOCK

This command generates a random block number within the range specified by MINBLOCK and MAXBLOCK. The block number can then be used by the next SCSI command.

MAXBLOCK {*n*}

This command specifies the maximum block for a range of blocks. The GRBLOCK and BLOCK commands use this value to determine the blocks to use in data transfers. (See the #BLOCKS command to specify the size of the range.)

MINBLOCK {*n*}

This command specifies the minimum block for a range of blocks. The GRBLOCK and BLOCK commands use this value to determine the blocks to use in data transfers.

10.4.6 Buffer Control Commands

COMPARE {*n*}

This command compares the write and read buffers following a SCSI operation. If no argument is given, the entire buffers are compared. If an argument, only that many bytes are compared, up to the buffer size.

CREATE {*n*}

This command selects a data pattern from the patterns listed in Table 7-7. The optional argument is in the range 1-15 hex. If no argument is given, the next pattern is selected.

PATTERN {*n*}

If an argument is given (1-15 hex), this command selects a data pattern from the patterns listed in Table 7-7, as in the CREATE command. If no argument is given, the available data patterns are shown, followed by the current data pattern.

RDISPLAY {*n*}

This command displays the data in the read buffer. If an argument is given (in hex), this number of bytes is displayed. If no argument is given, a page of 256 bytes is displayed. The display includes the address of the data and the ASCII interpretation of the data.

SWAPBUFF

This command swaps the read and write buffers by exchanging the start addresses.

WDISPLAY {*n*}

This command displays the data in the write buffer. If an argument is given (in hex), this number of bytes is displayed. If no argument is given, a page of 256 bytes is displayed. The display includes the address of the data and the ASCII interpretation of the data.

ZAPREADB {*n*}

This command fills the read buffer with a known data pattern. This is useful to assure that the next compare command is not comparing data from the previous read command. If no argument is given, the buffer is filled with the pattern 0BAD hex.

ZAPWRITE {*n*}

This command fills the write buffer with a known data pattern. If no argument is given, the buffer is filled with the pattern 0BAD hex.

10.4.7 Superblock Commands

DSC

This command reads and displays the superblock data from the last disk selected by the DSC and PORT commands.

RSB

This command reads and checks the superblock data from the last disk selected by the DSC and PORT commands. Certain field values are used to check command syntax, such as a valid block number or cylinder.

SUPER

This command initializes the superblock data in memory. This data is written by the next WSB command.

WSB

This command writes a superblock to the disk last selected. The data written is taken from the superblock table in memory. WSB should be used only after SUPER has run without error.

10.4.8 SCSI Commands

CAPACITY (0|1)

This command issues the SCSI Read Capacity command (SCSI code 25), which reads and displays the device capacity. The optional argument specifies the PMI (partial media) bit. Setting the PMI to 0 returns the last addressable logical block and the block length in bytes. Setting the PMI to 1 returns the actual last logical block and the block length.

DISK

This command selects the range of blocks from the first block to the maximum addressable block for the next disk operation. Use the RANGE command to select a smaller range.

DREAD

This command performs multiple reads of the entire disk. Any errors detected are stored in a table and displayed at the end of the operation. The superblock is read to determine the size of the disk. The entire disk is tested, except for the diagnostic cylinder.

DVERIFY

This command is similar to DREAD, except that the SCSI Verify command is used to force an ECC check on every block and no data is transferred. All errors detected are stored in a table and displayed at the end of the operation.

GLIST

This command issues the SCSI Display Defect Data command, to read and display the drive's Glist (SCSI code 37).

INQUIRE

This command issues the SCSI Inquire command (SCSI 12). The data returned in placed in the Read buffer and displayed.

LREAD

This command issues a SCSI Long Read command (SCSI code 28), and places the data in the read buffer. This command can be looped. 1 to 65535 blocks (decimal) can be read by this command.

LSEEK

This command seeks to the block specified by the last BLOCK command. It has a greater range than the SEEK command.

LWRITE

This command performs a SCSI Long Write command (SCSI code 2A), writing data from the write buffer to the peripheral. This command can be looped, and can be set to execute over a range of blocks. The transfer can be from 1 to 65535 blocks (decimal).

MODESENSE (1|2|3)

This command issues the modesense command and displays the data in ASCII form. The data for the specified page is displayed for verification.

PGLIST

This command issues the SCSI Display Defect Data command, to read both the Plist and the Glist and display the combined data (SCSI code 37).

PLIST

This command issues the SCSI Display Defect Data command, to read and display the drive's Plist (SCSI code 37).

RANGE start stop {n}

This command specifies a range of blocks for the next disk operation. The start and stop parameters specify the inclusive range for disk access. The optional parameter specifies the block size for transfers (1-FF hex). Use the LPORT command to specify the maximum block number that can be selected by RANGE. Use the DISK command to select the entire disk.

READ

This command issues a SCSI Read command (SCSI code 08), and places the data in the read buffer. This command can be looped. 1 to 255 blocks can be read by this command (see LREAD for larger reads).

RECALIBR

This command issues the SCSI Recalibrate command (SCSI code 01). The device recalibrates the heads to cylinder 0 and returns the block value to 0.

RCOMPARE

This command reads the SCSI peripheral and, if the command succeeds, compares the data in the write buffer with the data in the read buffer.

REMOVE

This command issues the SCSI Reassign Logical Block command (SCSI code 07). The logical block specified by BLOCK is reassigned to one of the spare sectors deallocated for this purpose. The block is added to the defect growth list for use with a later reformatting operation.

SEEK {*n*}

This command issues the SCSI Seek command (SCSI code 0B) to the specified cylinder, or to the last block selected if no cylinder is specified. The command cannot seek to the preserved diagnostic cylinders.

SENDDIAG

This command issues the SCSI Send Diagnostic command (SCSI code 1D), with the self-test bit set to 1.

SENSE

This command issues and interprets the SCSI Sense command to display the data bytes received (SCSI code 03).

SPECIFY (1|3|4)

This command issues the mode select command to the disk using tabular data. The program displays each data item and waits for your input. When all entries have been made, the mode select command is issued. This command is provided to verify that the disk will accept the mode select command without error. Care should be exercised, since the disk normally expects the format command to be issued following the mode select.

TABLE

This command displays the last error log table built by a RCOMPARE, DREAD or DVERIFY command.

USTATUS

This command issues the SCSI Test Unit Ready command (SCSI code 00). If no error is displayed, the unit is ready.

VERIFY

This command issues the SCSI Verify command (SCSI code 2F). The data at the specified blocks is verified for correct ECC, and can be run on a range of blocks. This command can be looped.

WRITE

This command performs a SCSI Write command (SCSI code 0A), writing data from the write buffer to the peripheral. This command can be looped, and can be set to execute over a range of blocks. The transfer can be from 1 to 255 blocks (decimal).

10.4.9 Special Function Commands

BOOTDISK

This command causes the system to boot from the SCSI disk specified by the DSC and PORT commands.

FAVI

This command returns FAVI to menu operation mode.

APPENDIX A

INSTALLING DIAGNOSTICS

A.1 OVERVIEW

Some of the diagnostic programs described in this manual may optionally be installed on the fixed disk. Installing the programs simplifies the process of loading the diagnostics, since removable media are not then required.

This section describes two installation procedures. The first procedure creates a special diagnostics partition, and installs the off-line diagnostic programs (the Executive, SIT, Logic Tests and Function Selects) on the partition. This procedure must be performed before installing the operating system software and applications. This feature is not supported on the MAI 2000.

The second procedure installs the BASS system exerciser programs. This installation is performed only after the operating system software has been installed. Instructions for installing the operating system are given in the BOSS/IX Technical Reference Manual, M6227. This feature is supported on all BOSS/IX systems.

A.2 INSTALLING OFF-LINE DIAGNOSTICS

The off-line diagnostics programs can be installed on the fixed disk Before installing these programs, you must create a special partition. After the partition is created, the diagnostics can be installed.

The procedure to create the diagnostics partition must be performed the first time diagnostics are to be installed, and any time the partition size is being changed. It is not required to update the diagnostics, when only the installation procedures needs to be performed.

A.2.1 Create Diagnostics Partition

This procedure can only be performed on a formatted disk. Since it reserves disk for diagnostics use, the procedure must be performed before installing the operating system.

NOTE: Creating or changing the size of the diagnostics partition initializes the disk super block. Therefore, any modifications to the diagnostics partition must be followed by a full installation of the operating system. This includes creating default disk partitions.

1. If a file system is already created on the disk and files installed, perform a complete file-by-file backup of all your files. Since the disk structure will be changed, an image backup cannot be used. Operating system files will be restored from the installation tape, and so do not need to be backed up, but you must back up the system configuration files, as described in the BOSS/IX Technical Reference Manual, M6227.

   NOTE: The existing partitions are destroyed by this procedure and will need to be created again. This process erases all files from the disk. The changed root partition size prevents you from restoring an image backup, so the backup must be file-by-file.

2. Perform an alternate load to the diagnostic cartridge or reel. When the system is booted, the diagnostic Executive prompt is displayed

3. Enable diagnostics service mode. Enter:

   service

   You are then prompted for the service password:

   Enter the password to enable Function Select service mode:

   Enter this password:

   b4bus

4. Prepare to update the superblock. To modify the superblock, the current volume ID must be destroyed. Enter:

   sbinit

   to destroy the current disk partitions and initialize the superblock for the diagnostic partition.

5. Destroy: "volume ID" (y/n)?

   To modify the superblock, type **Y**. Any other response terminates the procedure and repeats the <exec> prompt.

6. Diagnostic Partition size (in sectors)?

   This prompt asks for the number of sectors to allocate for the diagnostics partition. To provide space for the Executive, SIT, Logic Tests and Function Selects, enter:

   2000

7. Destroy: "partition name" (y/n)?

   Type **Y** to initialize the disk directory for the subsequent install. You are then prompted for the new name. Type the new name, up to 12 characters, and press **ENTER**.

   Type **N** to leave the name unchanged.

This completes the procedure for defining the diagnostics partition. The diagnostics can now be installed using the following procedure. Because the diagnostic executive is already running and service mode is enabled, you can continue with installing the diagnostics.

A.2.2 Install Diagnostic Programs

This part of the procedure copies the bootable diagnostic Executive program, SIT, Logic Tests and Function Selects to the diagnostic partition. If you are installing the programs immediately after creating the diagnostic partition, you are already in service mode, and may proceed with step 3.

1. Perform an alternate load to the diagnostic cartridge or reel. When the system is booted, the diagnostic Executive prompt is displayed

2. Enable diagnostics service mode. Enter:

   service

   You are then prompted for the service password:

   Enter the password to enable Function Select service mode:

   Enter this password:

   b4bus

3. Install all diagnostic programs. Enter:

   install

   The full set of installable diagnostics fits in a 2000 block partition.

4. Destroy: "partition name" (y/n)?

   Press **N** to begin the installation. (The partition name was entered in the previous procedure.)

   As each diagnostic is installed, its name and size (in sectors) is displayed. When all the files have been installed, the <exec> prompt is displayed again.

5. Following the installation, list the diagnostics installed. Enter:

   ldisk

This ends the procedure for installing the off-line diagnostic programs. You may now install the operating system, as described in the BOSS/IX Technical Reference Manual, M6227. Then restore your application program and data files.

To test the booting capability of the diagnostics, shutdown the system and reboot. The boot and load procedure is described in Section 4.

A.3 INSTALLING BASS

The BASS system exerciser programs are installed into the root filesystem, in the /sys directory. The procedure for installing them is the same as for installing the utility programs and BASIC.

This procedure should only be run in single user mode. The system should be booted normally and rooted on the root partition.

The BASS programs are contained on either a series of floppy diskettes (MAI 2000 only) or on a tape cartridge or reel labeled EBA.

A.3.1 Installation From Tape

If you are installing from tape, do the following steps:

1. Bring the system to single-user mode.

2. Mount the Diagnostics medium (CS or TS), and make the device ready.

3. At the system administrator prompt, enter the command line appropriate for the installation media:

   install cs EBA

   or

   install ts EBA

   The program looks for EBA on the tape, displays its saveset information, and begins copying files. The name of each file is displayed as it is copied.

This completes the installation procedure.

A.3.2 Installation From Floppy

The floppy installation procedure applies only to the MAI 2000. If you are installing from floppies, Utilities and BASIC must be installed separately.

1. Bring the system to single-user mode.

2. Insert the diskette labeled EBA #1 into the diskette drive and press the lock button.

3. At the ADMIN> prompt, enter:

   install fd0 EBA

4. The program displays:

   Insert diskette number #, <RETURN> when ready:

   Insert the indicated floppy and press **ENTER**. The diskette label is then displayed, and you are asked:

   Is this the correct RELEASE diskette (y/n)?

   If the diskette is correct, type **Y**; otherwise type **N** and insert the correct diskette.

   The program then begins copying files. The name of each file is displayed as it is copied.

5. When all the files on the first floppy have been copied, you are prompted to insert the next floppy in the set. Change floppies and press **ENTER**.

   Continue changing floppies as promtped until all the files have been copied. When the program has finished copying the files from the last floppy, the program ends.

This completes the installation procedure.