VOLUME 2

Insert the following items into binders in the order that they
appear below:

20. Tab marked SYSTEM DIAGNOSTICS

21. Tab marked ADCC DIAGNOSTIC

22. Manual (P/N 30070-90037) - Asynchronous Data Communication
    Diagnostic Manual

23. Tab marked GIC DIAGNOSTIC

24. Manual (P/N 30070-900039) - General I/O Channel Diagnostic
    Manual

25. Tab marked SERIES 30/33 MEMORY DIAGNOSTIC

26. Manual (P/N 30070-90038) - Error Correcting Memory Diagnostic
    Manual

27. Tab marked SERIES 40/44 MEMORY DIAGNOSTIC MANUAL

28. Manual (P/N 30092-90001) - HP 3000 Series 40/44 Pronto Memory
    Diagnostic Manual

29. Tab marked SERIES 40/44 CMP MAINTENANCE MODE MANUAL

30. Manual (P/N 30090-90007) - HP Series 40/44 CMP Maintenance
    Mode Manual

31. Tab marked SERIES 40/44 CMP REMOTE MAINTENANCE MANUAL

32. Manual (P/N 30090-90006) - HP Series 40/44 CMP Remote Main-
    tenance Manual

33. Tab marked PERIPHERAL DIAGNOSTICS

34. Tab marked 7902/9895 DIAGNOSTIC

35. Manual (P/N 30070-90040) - 7902/9895 Flexible Disc Diagnostic
    Manual

36. Tab marked 7970 DIAGNOSTIC

37. Manual (P/N 30070-90015) - 7970 Magnetic Tape Diagnostic Man-
    ual

MAR 1982

NOTICE

The information contained in this document is subject to change without notice.

HEWLETT-PACKARD MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATER-
IAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY
AND FITNESS FOR A PARTICULAR PURPOSE. Hewlett-Packard shall not be liable for errors
contained herein or for incidental or consequential damages in connection with the furnishing, perfor-
mance or use of this material.

This document contains proprietary information which is protected by copyright. All rights are reserved.
No part of this document may be photocopied or reproduced without the prior written consent of
Hewlett-Packard Company.

LIST OF EFFECTIVE PAGES

The List of Effective Pages gives the date of the current edition and of any pages changed in updates to that edition. Within the manual, any page changed since the last edition is indicated by printing the date the changes were made on the bottom of the page. Changes are marked with a vertical bar in the margin. If an update is incorporated when an edition is reprinted, these bars are removed but the dates remain.

PRINTING HISTORY

New editions are complete revisions of the manual. Update packages, which are issued between editions, contain additional and replacement pages to be merged into the manual by the customer. The date of the title page of the manual changes only when a new edition is published. When an edition is reprinted, all the prior updates to the edition are incorporated.

First Edition. . . . . Sep 1978
Update No. 1 . . . . May 1980
Update No. 2 . . . . Mar 1981

CONTENTS

CONTENTS (continued)

ADCC Diagnostic

Asynchronous Data Communications Channel — Main
and
Asynchronous Data Communications Channel — Extended
Diagnostic Manual

HEWLETT PACKARD

LIST OF EFFECTIVE PAGES

The List of Effective Pages gives the date of the current edition and of any pages changed in updates to that edition. Within the manual, any page changed since the last edition is indicated by printing the date the changes were made on the bottom of the page. Changes are marked with a vertical bar in the margin. If an update is incorporated when an edition is reprinted, these bars are removed but the dates remain.

PRINTING HISTORY

New editions are complete revisions of the manual. Update packages, which are issued between editions, contain additional and replacement pages to be merged into the manual by the customer. The date of the title page of the manual changes only when a new edition is published. When an edition is reprinted, all the prior updates to the edition are incorporated.

CONTENTS

CONTENTS (continued)

1.0 INTRODUCTION

This diagnostic test program confirms correct operation of the HP
Main Asynchronous Data Communications Channel and the HP Extended
Asynchronous Data Communications Channel. It operates on HP 3000
HP-IB version computer systems.   Communication with the operator
is via the system console.

Hereafter in this document the Asynchronous Data Communications
Channel will be referred to as the ADCC.

The test method consists of executing one or more instructions to
provide stimulus to the circuitry under test, then comparing the
result to predetermined or calculated values or tables residing
in memory. This involves the use of many CPU base set instruc-
tions as well as Channel Program Processing instructions.

1.1 REQUIRED HARDWARE

The following hardware is required:

a. ADCC test connector "ADCC TEST HOOD" HP 5060-5563 (see Figure
   5.1)

b. For HP 3000 HP-IB version systems: a known-good ADCC Main for
   console communication.

Figure 5.2 shows the signal name/mnemonic to pin number cross
reference of J2.

1.2 REQUIRED SOFTWARE

Diagnostic/Utility Disc or Tape

1.3 PROGRAM ORGANIZATION

The ADCC hardware consists of two boards, one of which is
optional.   The first board (hereafter called the ADCC Main) pro-
vides four RS-232-C ports and modem controls.   The second board
(hereafter called the ADCC Extended) may be added to provide an
additional four RS-232-C ports plus additional modem controls for `

The diagnostic program performs repetitive tests on each device present on the ADCC in ascending order, so that the IMB control and data lines which are common to device numbers on the ADCC are tested several times for each step.

With the exception of Section 13, each device is tested individually, so that only one device is active at any one time.

1.4 MESSAGES AND PROMPTS

Three types of messages are output by the diagnostic: error, information, and prompt messages.

Error messages are used to inform the operator when the ADCC responded unexpectedly to a given stimulus. Error messages will begin "Error in Step XX" where XX is the step number executing when the error condition was detected, followed by the text of the error message. Unless specified otherwise, a pause occurs after each error. (See Section 3 -- Operating Instructions).

Information messages (e.g., title and end-of-pass) will be displayed with no program pause.

Prompt messages (e.g., which channel number to test, etc.) require the operator to input the requested information.

1.5 PHILOSOPHY

The ADCC Diagnostic begins by testing for those elements of the ADCC which must work in order for any further testing to be meaningful (e.g., IMB handshake logic and channel address logic). When it has been established this rather significant subset of the ADCC logic works correctly, a larger subset of the logic is tested. This type of testing continues until all of the logic on the ADCC has been found to be operating correctly.

The tests which are directed at the logic peculiar to a device number are executed such that only one device number at any one time is active (except in Test Section 13 - ADCC Channel Program). These tests are executed on device number 0 first, then 1, ... until all device numbers present on the ADCC have been tested.

With the exception of Test Sections 10, 11, 12, and 13, all tests are executed using Direct I/O software instructions. This method is used to minimize the amount of logic under test at a given time. When it has been determined that a large enough subset of the ADCC logic is functional, channel program instructions are used to verify that this logic functions correctly at machine speed (channel programs exercise the board much faster). Also, at this time all devices present are active simultaneously, so that any interaction or contention between devices can be tested.

1.6 TEST LIMITATIONS

SET-UP

The ADCC requires a set-up procedure in order to transmit and
receive data. The following describes the minimum required ADCC
initialization procedure:

a. WRITE MODIFIER 1 - UART Control A byte written with this
   modifier establishes UART control functions necessary for the
   UART to transmit and receive intelligible data.

b. WRITE MODIFIER 6 - Baud Rate Select Bytes written with this
   modifier are used for the selection of transmit and receive
   baud rates.

If, for any reason, the above writes cannot be completed or the
control circuitry does not respond correctly to the data byte(s)
written with one or both of the above mentioned modifiers, diag-
nostic program will be unable to determine whether an unsuccess-
ful data transfer was due to a UART malfunction, a UART control
circuitry malfunction or a baud rate control circuitry malfunc-
tion.

CHANNEL ADDRESS SWITCH POSITIONS

Since it would mean reloading this diagnostic program to test all
15 positions of the Channel Address Switch, only 14 positions are
tested in Section 16. The positions not tested are N(0), because
there is no channel 0, and the position which corresponds to the
channel number being used for the system console. If it is
desired to test the remaining position (being used to operate the
system console), the operator must change the channel address of
this channel then reload and restart the diagnostic.

IDENTIFY CIRCUITRY ON ADCC MAIN

The ADCC is specified to assert Data-Not-Valid on the IMB if an
attempt is made to issue an IDENTIFY command to any device num-
bered 4-7 while the Extended ADCC is not present.

The test which verifies the correct operation of this specifi-
cation is performed automatically if the Extended ADCC is not
present, but cannot be performed if the Extended ADCC is present.

If it is desired to execute the test for this specification, but
the Extended ADCC is present, the operator may disconnect the
Extended ADCC by removing the frontplane interconnecting cable on
connector J1, and restarting the test.

ADCC Diagnostic

2.0 INTRODUCTION

Before running this diagnostic be sure that the physical config-
uration (switches on the frontplane) matches the logical config-
uration required by the operating system.

If the diagnostic is being used to test the console ADCC (CHAN
ADDR=1) on an HP 3000 HP-IB version system, the following prelim-
inary procedure must be followed:

(1) Power down the system.

(2) If the ADCC under test is the only one in the system, insert
a spare ADCC Main.

(3) Remove the ADCC cable (5061-2502) from the ADCC under test
and connect it to a spare ADCC Main.

(4) Set the CHAN ADDR of the spare ADCC to 1.

(5) Set the CHAN ADDR of the ADCC under test to an unused CHAN
ADDR.

(6) Double check that there are no duplicate CHAN ADDR settings
by looking at all CHAN PCAs in the system. (Damage to
channels can result if the channels are at the same address.)

There are two modes of operation possible for this diagnostic
program, the standard (default) mode and the extended mode.

To operate in either mode, the following steps must be executed:

(1) Bring up the Diagnostic/Utility System (DUS)

(2) The DUS prompt character (:) is displayed.

(3) Respond ´ADCC DIAG´, to load the Diagnostic.

(4) The ADCC Diagnostic Program displays its title message and
prompt character ´>´

The next step (step 5) depends on whether you want the Standard
Mode or not.

2.1 STANDARD MODE

(5) If the standard mode is to be executed, the operator responds
´GO´ and diagnostic execution begins.

ADCC Diagnostic

The diagnostic will continue execution until an error condition
is detected or all standard sections have been executed.

The standard mode is defined as follows:

a. Execute all Sections 1 through 13 (see Figure 3.1)

b. Display error, information, and prompt messages

c. Pause on errors and prompts.

2.2 EXTENDED MODE

(5) If the extended mode is to be executed, the operator may
    input one or more of the commands shown in the table on the
    next page.

After all desired options have been entered, the operator enters
´GO´, and the diagnostic will begin execution.

```
Output and Pauses

   *EEPR   - enable error messages.
    SEPR   - suppress error messages

   *ENPR   - enable non-error messages.
    SNPR   - suppress non-error messages.

   *EEPS   - enable pauses after error messages
    SEPS   -  suppress pauses after error messages

    RST    - reset print and pause commands to 'enable'.
               (Equivalent of EEPR + EEPS)

   *Default Value


Test Selection

    TEST     - change from the default set of section execution

               'TEST 1,5,8' -- execute sections 1,5, and 8.
               'TEST 1/3,8' -- execute sections 1,2,3, and 8.
               'TEST  +3,6' -- add sections 3 and 6.
               'TEST  -3,6' -- remove sections 3 and 6.


Program Control

    GO       - continue Diagnostic execution from a pause.
    GO,1     - suppress the initial pause and message
               (instructs the operator to attach the test
               connector(s) on J2.)

    EXIT     - stop diagnostic execution and return to the DUS

    RUN      - restart execution of diagnostic at the beginning.

    LOOP     - loop on the selected sections.
    LOOPOFF  - supercedes the LOOP command.

    DEV(n)   - perform selected test sections only on
               the device number specified by (n).
               (Default: test all)
```

Figure 2.1 - ADCC Diagnostic Commands

ADCC Diagnostic

EXECUTION TIMES

Std	Test Sect.	Test Steps	Name	Approximate run time/pass	Notes
*	1	1-4	ROCL - Read Reg 2,14	milliseconds	
*	2	5-11	OBII,OBSI,Interrupts	milliseconds	
*	3	12	Chl Address Recognition	milliseconds	
*	4	13-14	Register 1-15 Integrity	milliseconds	
*	5	15	Baud Rate Clocks / UARTs	10 sec/device	1
*	6	16-17	Special Character Detect	2 sec/device	1
*	7	18-26	Device Specified Jump	milliseconds	1
*	8	27-36	Modem Status and DSJ	2 seconds	1
*	9	37	DSJ Priority Structure	milliseconds	1
*	10	38	Chl Program Read Reg 2	milliseconds	2
*	11	39-41	Identify	milliseconds	2
*	12	42-48	Inhibit CSRQ Assertion	5 seconds	
*	13	49	ADCC Channel Program	3 seconds	1,2
	14	50-51	Extended Spec Char RAM	3 min/board	1
	15	52	Extended Baud Rate/UARTs	2 min/device	1
	16	53-55	Channel Address Switch	operations	
	17	56-57	RS-232-C Cable Test	milliseconds	

total run time per pass:

          standard set = 45 sec (4 devices - Main only)
                         1-1/2 min (8 devices - Main and Extended)

          complete set= 12 min (4 devices - Main only)
                        24 min (8 devices - Main and Extended)

Notes:

      *   part of standard set of Sections
      1   loopback connector utilized
      2   channel programs utilized

Figure 3.1 - Table of Sections

ADCC Diagnostic

4.0 INTRODUCTION

This diagnostic program consists of a control/initialization sec-
tion and 17 test sections.

The control/initialization section displays the diagnostic title
message, instructs the operator to replace the RS-232-C cable
with the test connector HP 5060-5563, ask the operator to input
the Channel Address of the ADCC under test, and asks whether the
Extended ADCC is present.

"Asynchronous Data Communication Channel Diagnostic Rev 00.00",
"Type GO to continue (LC for List Commands)"

NOTE: If TEST 16 is input the next three messages don't
appear.

"Replace ADCC cable(s) with test connector(s), respond GO"

"Enter Channel Number of ADCC under test"

"Is Extended ADCC present?(Y/N)"

4.1 TEST SECTION 1 — Roll Call and Register 2, 14 Reads

This section places the command decode logic, the channel address
recognition logic, the global response logic, the register decod-
ing logic and the handshake control logic under test.

Step 1 - Roll Call (ROCL)

The ability of the ADCC under test to respond to the
global I/O instruction ROCL is tested. A ROCL is
issued, and the resulting returned word is tested to
verify that the bit which corresponds to the ADCC chan-
nel number has been set to a one.

"ADCC under test did not respond to ROCL"

Step 2 - Read Register 2

The ability of the ADCC under test to return the cor-
rect value in response to a Register 2 read is veri-
fied. A register 2 read is issued, and the value of
the resulting returned word is verified to be !800A,
the expected value of register 2 at this time. A fur-
ther test of this register is done in Test Section 11.

"Register 2 read is !XXXX expected !800A"

XXXX is the value of the returned word.

Figure 4.1 shows the expected value for a register 2 read.

Step 3 – Channel Configuration Register – 'CPP' testing the Channel Program Processor bit specification.

The contents of Register 14 are read and the sense of bit 2 is tested. If the sense of bit 2 is one, indicating that the ADCC is using the CPP, the operator is asked to set the processor select switch to the CPU position. At the end of the diagnostic, a message is displayed as a reminder to reset this switch.

Step 4 – Channel Configuration Register – 'CPU' testing the Central Processing Unit bit specification.

The value of Register 14 is read, and the returned word is verified to the format described in Figure 4.2.

"Register 14 read is !XXXX expected !80N1"

XXXX is the value of the returned word, and the value of N is determined by the sense of bit 11.

```
    Register 2

     0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
    |---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
    | 1   0   0   0 | 0   0   0   0 | 0   0   *   0 | 1   0   1   0 |
    |---------------|---------------|---------------|---------------|
    |       8       |       0       |   2=CSRQ      |       A       |
    |               |               |   0=no CSRQ   |               |
     ---------------------------------------------------------------
```

         Bit 0   - the ´OR´ of the other bits in this word.

         Bits 1-9   - have no meaning on an ADCC, but are defined as all zeroes for use by the channel program microcode

      * Bit 10   - Poll response. This bit will be set to a one whenever the ADCC is asserting Channel Service Request on the IMB.

         Bits 11-15   - have no meaning on an ADCC, but are defined as !A for use by the channel program microcode

Figure 4.1 - Register 2 Bit Significance

```
    Register 14
    0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
   |---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
   | 1   0   *   0 | 0   0   0   0 | 0   0   0   ‡ | 0   0   0   1 |
   |---------------|---------------|---------------|---------------|
   |    A=CPP      |       0       |    1=EXT      |       1       |
   |    8=CPU      |               |    0=no EXT   |               |
    -------------------------------------------------------------
```

	Bit 0		—	always a one on an ADCC
	Bit 1	HYBRID	—	always a zero on an ADCC
*	Bit 2	CPU/CPP	—	zero if the PROCESSOR switch is set to 'CPU' position (In)
				one if it is set to the 'CPP' position (Out)
	Bits 3-10		—	always zeros on an ADCC
‡	Bit 11	MAIN/EXTENDED	—	zero if the Extended ADCC is not present
				one if it is present
	Bits 12-15	CHANNEL TYPE	—	indicates channel is an ADCC. (a General Interface Channel returns a zero in this nibble.)

Figure 4.2 - Register 14 Bit Significance

4.2 TEST SECTION 2 — OBII, OBSI and Interrupts

Test Section 2 places the interrupt queue, priority encoding, interrupt mask and the global response logic function blocks under test.

Step 5 - Obtain Interrupt Information (OBII)

OBII is a command issued by the interrupt processing micro-code. Step 5 issues a register 0 read with a special command code to emulate OBII. The resulting returned word, bits 9 thru 12 are verified to contain the value of the channel address of the ADCC under test.

"OBII returned YY expected ZZ"

YY is the value of bits 9 thru 12 in the returned word and ZZ is the value of channel address of the ADCC under test (the expected value).

Step 6 - Obtain Service Information (OBSI)

OBSI is a command issued by the channel service process micro-code. Step 6 issues a register 0 read with a special command code to emulate OBSI. The resulting returned word, bits 9 thru 12 are verified to contain the value of the channel address of the ADCC under test.

"OBSI returned YY expected ZZ"

YY is the value of bits 9 thru 12 in the returned word and ZZ is the value of channel address of the ADCC under test (the expected value).

Step 7 - Interrupt Queue - cleared by INIT

Because the interrupt queue may be in a random state initially, an INIT is issued followed by a software equivalent OBII. The resulting returned word is then verified to be the value of the channel address in bits 9 thru 12, and a seven (the default case when no interrupts are queued) in bits 13,14,15.

"Interrupt queue not cleared by INIT"

Step 8 - Interrupt Queue

The interrupt queue for all eight device numbers is present on the ADCC-Main, so all eight are tested regardless of the absence of the ADCC-Extended. An interrupt is queued by a write to register 12 with bit 12 set to a one, and the device number of the device to be queued in bits 13,14,15. The test sequence is: all

devices are queued, then OBII is issued to check bits
13,14,15 of the returned word one at a time 0 thru 7.

"Interrupt queue is X expected Y"

X is the value of bits 13,14,15 of the returned word,
and Y is the value of the device number which was
queued.

Step 9 - Interrupt Queue cleared by INIT

Assured by Step 8 that the interrupt queue is not in a
random state, it is now necessary to insure that INIT
will indeed set the Interrupt Queue to the default
condition (all zeroes in the queue, but inverted when
read by the priority encoder). An interrupt for device
0 is queued, then an INIT is issued and OBII used to
verify bits 13,14,15 are reset to all ones.

"Interrupt queue not cleared by INIT"

Step 10 - Queued Interrupts and IRQ

Being assured by Step 8 that interrupts can be queued,
Step 10 sets the interrupt mask bit for the ADCC under
test and allows external interrupts with the execution
of an ION instruction. A short wait loop is employed
to allow the CPU to process the expected IRQ, then the
program verifies that the interrupt was requested by
the ADCC under test.

Two error conditions are possible as a result of Step
10, no interrupt occurred, or an unexpected device was
the cause of the interrupt.

"Queued device D did not cause interrupt"

D is the value of the device number which was queued to
interrupt.

"Unexpected interrupt from device D"

D is the value of the device number causing the
unexpected interrupt.

Step 11 - Priority Encoder

The priority encoder is tested by queuing interrupts
for all devices, then allowing external interrupts. As
each interrupt is processed, OBII is used to insure the
correct device caused the interrupt, holding off
interrupts from lower priority devices. The highest
priority device is then removed from the interrupt
queue, and the next highest priority device is tested.

The step is completed when all eight device numbers are tested.

"Device X did not hold off interrupt from device Y"

X is the value of the device number under test and Y is the value of the device number causing the unexpected interrupt.

4.3 TEST SECTION 3 — Channel Address Recognition

Section 3 gives a more extensive test of the Channel Address recognition logic than Section 1, and places a subset of the interrupt queue, device address latches, priority encoding and interrupt mask under test. This test section is directed at the case of one channel responding to a Roll Call, Interrupt Poll or Service Poll as more than one channel.

Step 12 - Channel Address Recognition

A test is made by queueing an interrupt and enabling the mask for only the channel number under test, keeping the external interrupt system off. An IPOLL at this time should return only the channel number of the ADCC under test.

"IPOLL returns !XXXX expected !YYYY"

The value of XXXX is determined by the word returned by IPOLL (e.g., if channel 2 responds to the IPOLL, !2000 will be returned) and the value of YYYY is determined by the expected IPOLL Return (e.g., if channel 4 was expected to respond, !0400 will be the expected value).

4.4 TEST SECTION 4 — Register 1-15 Integrity

Test section 4 places the register decoding logic under test by verifying that registers 1, 3 through 13, and 15 return all zeros as the result of a register read, and then verifies that write to any register 1 thru 15 will not alter the contents of any register.

Step 13 - Registers 1, 3 thru 13, and 15 return zeros

Registers 1, 3 thru 13, and 15 are read, and the returned words verified to be zero. (Registers 2 and 14 were previously tested for correct value in Test Section 1)

"Register XX returned !YYYY expected !0000 Dev=D"

> XX is the value of the register number and YYYY is the
> returned non-zero value.

Step 14 - Registers affected by write register

> Write data commands on the IMB will have an effect on
> the ADCC only if the register number referenced is 0 or
> 12.   Any other register number referenced will cause a
> normal IMB handshake, but no data will actually be
> written.

> An all ones pattern is written into each register 1
> thru 15. After each write, all registers 1 thru 15 are
> read, and the returned values tested to verify that the
> contents are unaffected by the write.

> Note that the all-ones write into Register 12 will
> queue an interrupt, but the interrupt will not be
> processed because external interrupts are disabled at
> this time. The queued interrupt will be cleared at the
> end of this step by execution on an INIT instruction.

"Write to register X changed register Y Dev=D"

> X and Y are the values of the register numbers under
> test.

4.5 TEST SECTION 5 — Baud Rate Clock and UARTS

Step 15 - Baud Rate Clock and UARTs

> Step 15 verifies correct operation of the Baud Rate
> Clock logic, the device address latches logic, and a
> subset of the UART logic.

> Each device is individually initialized to transmit and
> receive a test byte at the various baud rates shown in
> Figure 4.4. The UART Control for this test is one start
> bit, two stop bits, even parity, and 8 data bits. A
> test byte (!55) is transmitted, and the program waits
> approximately 500 milliseconds to allow the looped-back
> test byte to be received by the UART.   The UART's
> receive buffer is read, and the contents are verified
> to be !55 (the value of the byte transmitted).   The
> test connector shown in part by Figure 4.3, on the J2
> edge connector, provides the data paths required to
> loop the UART's output to its input.

Once it has been established that the UART is capable
of transmitting and receiving data, test byte charac-
ters with various bit patterns are transmitted,
received, and verified allowing between transmit and
receive a delay time sufficient for one character. The
test bytes used are !0, !55, !AA, and !CC.

"!55 not received after 500 msec. Baud=XXXXX Dev = D"

"UART control=!ZZ. Sent !XX received !YY Dev = D"

XXXXX is the value of the baud rate being tested, ZZ is
the UART control byte (see Figure 4.7) XX is the value
of the test byte, YY is the value of the received byte
and D is the value of the device number under test when
the error was detected.

The step is complete when all devices on the ADCC have
been tested for all combinations of test bytes and all
baud rates.

Please note these limitations:

The ADCC requires a a set-up procedure in order to
transmit and receive data. The following describes the
minimum required ADCC initialization procedure:

a. WRITE MODIFIER 1 - UART Control A byte written
    with this modifier establishes UART control func-
    tions necessary for the UART to transmit and
    receive intelligible data.

b. WRITE MODIFIER 6 - Baud Rate Select Bytes written
    with this modifier are used for the selection of
    transmit and receive baud rates.

If, for any reason, the above writes cannot be com-
pleted or the control circuitry does not respond cor-
rectly to the data byte(s) written with one or both of
the above mentioned modifiers, the diagnostic program
will be unable to determine whether an unsuccessful
data transfer was due to a UART malfunction, a UART
control circuitry malfunction or a baud rate control
circuitry malfunction.

Figure 4.3 - J2 Loopback Connections (Data Lines)

The following values were calculated for the baud rate period.

Period = (number of bits in word* + 10%) / baud rate

* including 1 start bit, 1 parity bit, and 1 stop bit.

```
------------------------------------
| BAUD  |         |  PERIOD OF      |
| RATE  |  BAUD   |  BAUD RATE      |
| CODE  |  RATE   |  + %50          |
|-----------------------------------|
|       |    .    |                 |
|   2   |   50    |  0.30      sec  |
|   3   |   75    |  0.20      sec  |
|   4   |  134.5  |  0.111     sec  |
|   5   |  200    |  0.75      sec  |
|   6   |  600    |  0.25      sec  |
|   7   |  2400   |  0.00625   sec  |
|   8   |  9600   |  0.0015    sec  |
|   9   |  4800   |  0.0031    sec  |
|  10   |  1800   |  0.0083    sec  |
|  11   |  1200   |  0.0125    sec  |
|  12   |  2400   |  0.00625   sec  |
|  13   |  300    |  0.50      sec  |
|  14   |  150    |  0.10      sec  |
|  15   |  110    |  0.136     sec  |
------------------------------------
```

Figure 4.4 - Baud Rate Timing Table

4.6 TEST SECTION 6 — Special Character Detection

This Section verifies correct operation of the Special Character
Detection logic.   Note that although the function under test is
special character detection, the command decoding, channel ad-
dress recognition, device address latch, baud rate clock, and
UART logic functions must all work.

Note that optional Section 14 is the Extended Special Character
RAM Test.   The time required to fully test crosstalk, interfer-
ence, etc in the special character ram is several minutes.   If an
interference problem is suspected, the operator must specifically
select Section 14, otherwise, it will NOT be executed.

Step 16 - Clear Special Character RAM

Bytes written with a modifier 4 are removed from the
device's list of special characters. All 256 special
character RAM addresses (for each device) are cleared.
This is done with a modifier 4 write (to each device on
the ADCC) with all data bytes 0-255. Then a modifier 0
write is performed for all data bytes 0-255. Following
each modifier 0 write, the loopback path described in
Section 5 is used to transmit and receive the charac-
ter. Each test byte is then read (using an RIOC
instruction) and verified to have no special character
tag appended.

The step is complete when all addresses of the special
character RAM (all data bytes 0-255) for each device
have been tested.

There are two possible error conditions in this step.
The test byte received by the UART does not match the
test byte transmitted, or the test byte was incorrectly
tagged as a special character.

"Character transmitted is !XX, received !YY, Dev = D"

!XX is the test byte transmitted, !YY the byte received
and D is the value of the current device number under
test. If this error condition occurs, the problem is
probably in the UART or UART Control circuitry.

"!XX incorrectly tagged as special character Dev = D"

!XX is the test byte, and D is the value of the current
device number under test.

Step 17 - Set Special Character RAM

Bytes written with a modifier 5 are tagged by a device
on the ADCC as a special character. All 256 special
character RAM addresses (for each device) are set to
one. This is accomplished by a modifier 5 write for
all data bytes 0-255. The test method is the same as
described in Step 16, except that each received test
byte is verified to have the special character tag
appended.

Step 17 is complete when all addresses of the special
character RAM space for all devices present on the ADCC
have been tested. As in Step 16, there are two pos-
sible error conditions. The wrong character is
received or the character was not tagged as a special
character.

"Character transmitted is !XX received !YY. Dev = D"

> !XX,!YY and D are the values explained in the previous
> step.

"!XX not tagged as special character Dev = D"

> !XX is the character, and D is the value of the device
> number under test.

4.7 TEST SECTION 7 — Device Specified Jump (DSJ)

This Test Section places under test: the device specified jump
logic, a part of the UART's control and status logic, and the
break detection logic.

This section verifies the ability of the device specified jump
logic to return the correct value to the CPU according to the
applied stimulus. The test is run on all device numbers present
on the ADCC.

> A zero is returned if the ADCC is requesting service because
> the UART receive buffer of the device under test contains a
> character which may be read.

> A one is returned if the ADCC is requesting service because
> the UART transmit buffer of the device under test may be
> loaded with a character to be transmitted.

> A two is returned if the ADCC device status should be read.
> (In case of an overrun or parity error, or a modem status
> line has changed from reference.)

> A three is returned if no device on the ADCC is requesting
> service.

Please note that the device specified jump stimulus used in this
Test Section is a "software equivalent" DSJ, and not the channel
program instruction. This is done to minimize the amount of logic
under test at this time (see Paragraph 1.5 for a more detailed
explanation if necessary).

Step 18 - Board is not requesting service

> An INIT instruction is executed, a DSJ is issued for
> each device, and the resulting returned byte is tested
> to verify that DSJ returns a three for this condition
> (no request).

"Device Specified Jump returned X expected 3. Dev=D"

Step 19 - Board is requesting service because the receive buffer
          contains a character which may be read.

Each device on the ADCC is set up to request service when a character has been received. A character is then transmitted and received (using the loopback connector). The DSJ is issued, and the returned byte is verified to be a zero.

"Device Specified Jump returned X expected 0. Dev=D"

Step 20 - Board is requesting service because a character may be loaded into the UART transmit buffer.

Each device on the ADCC is set up to request service when the transmit buffer is empty. The DSJ is issued, and the returned byte is verified to be a one.

"Device Specified Jump returned X expected 1. Dev=D"

Step 21 - Board is requesting service - parity error

A parity error is induced by setting each device to transmit at 9600 baud and receive at 4800 baud. A character (chosen to generate a parity error with this setup) is transmitted and received. A UART status (modifier 1) read is done to verify the parity error is detected by the UART. Then a DSJ is issued, and the resulting returned byte is verified to be a 2.

The error conditions possible are:

a. The UART status (modifier 1) read shows that the induced error was not detected/reported by the UART and the DSJ did not respond with status return. This indicates that the fault probably lies in the UART and/or associated logic.

"Device D UART did not detect parity error"

b. The UART status read shows that the induced error was not detected/reported by the UART logic BUT the DSJ did respond with a status return. This indicates that the fault probably lies in the modifier 1 logic.

"Device D modifier 1 did not report parity error"

c. The UART status read shows that the induced error was reported by the UART logic, but the DSJ did not respond with the status return. This indicates that the fault probably lies in the DSJ logic.

"Device Specified Jump returned X expected Y. Dev=D"

Step 22 - Board is requesting service - overrun error

An overrun condition occurs when:

a. a byte has been received by the UART while 'character received' service condition is enabled (write modifier 3),

b. this character is not read, and

c. another character is received.

Note - the first character need not be 'valid' (it may be received with a parity, framing or overrun error) the second character received can still cause an overrun error.

Since the UART has a one-character receive buffer, two characters are transmitted and looped back to the UART's receiver (without reading the first.) The UART should provide 'overrun' status. A UART status (modifier 1) read is done to verify that the overrun was detected. Then a DSJ is issued and the resulting returned byte is verified to be a two.

The error conditions possible as a result of the stimulus described above are:

a. The UART status (modifier 1) read shows that the induced error was not detected/reported by the UART and the DSJ did not respond with status return. This indicates that the fault probably lies in the UART logic.

"Device D UART did not detect overrun error "

b. The UART status read shows that the induced error was not detected/reported by the UART logic but the DSJ did respond with a status return. This indicates that the fault probably lies in the modifier 1 logic.

"Device D modifier 1 did not report overrun error "

c. The UART status read shows that the induced error was reported by the UART logic, but the DSJ did not respond with the status return. This indicates that the fault probably lies in the DSJ logic.

"Device Specified Jump returned XX expected YY. Dev=D"

Step 23 - Board is requesting service - break detected

A break condition is emulated by setting the UART to receive at a much faster baud rate than is transmitted, then an all zeroes character is sent. A UART status (modifier 1) read is executed to verify that the break condition was detected, then a DSJ is issued, and the resulting returned byte is verified to be a two.

The error conditions possible are:

a. The UART status (modifier 1) read shows that the induced error was not detected/reported by the UART and the DSJ did not respond with status return. This indicates that the fault probably lies in the UART logic.

"Device D UART did not detect break"

b. The UART status read shows that the induced error was not detected/reported by the UART logic but the DSJ did respond with a status return. This indicates that the fault probably lies in the modifier 1 logic.

"Device D modifier 1 did not report break"

c. The UART status read shows that the induced error was reported by the UART logic, but the DSJ did not respond with the status return. This indicates that the fault probably lies in the DSJ logic.

"Device Specified Jump returned X expected Y. Dev=D"

Step 24 - Break detected - EOT appended.

A break condition is again emulated using the method described in the previous step. At this time, the END message (or EOT) is verified to be a one -- that the EOT is appended to the UART status read.

"EOT not appended at detected break Dev = D"

Step 25 - TBE status = 1

With service request condition 2 enabled, bit 15 of the word returned as the result of a modifier one read, when the UART transmit buffer is empty , should be set to a one. This step issues an initialize, sets up service request condition 2, executes a modifier one read and verifies the state of this bit. If an error is detected in this step, the message

"TBE not returned as 1 Dev = D"

is displayed.

Step 26 - TBE status = 0

With service request condition 2 enabled, bit 15 of the word returned as the result of a modifier one read, should be set to zero if the UART transmit buffer is not empty. This condition is set by setting the UART to transmit at 4800 baud, sending a character to the

buffer and then verifying that bit 15 of the resulting
returned word to be a zero. If an error is detected in
this step, the message

"TBE not returned as 0 Dev = D"

is displayed.

4.8 TEST SECTION 8 — Modem Status and Device Specified Jump

Test Section 8 places under test the modem controls and the modem
status monitor logic function blocks along with the poll and
device specified jump logic function blocks.

This is done by testing the RS-232-C output lines CA (request to
send), CD (data terminal ready), CH (speed select), and SCA
(secondary request to send) of each device on the ADCC.

These lines are looped-back via the J2 test connector (see Figure
4.5) to the modem status lines [CB(clear to send), CC(data set
ready), CE(ring detect), CF(signal detect), and SCF(secondary
signal detect)] and then verifying the ability of the device
specified jump logic to detect the change in the modem status
lines.

Test Condition 1 (reference = 1)

Each output/modem status line is set by sending a byte,
written with modifier 7, to establish the reference at one
for each line under test, then setting the mask bit for that
line, enabling the modem status monitor.

The stimulus is provided by writing a data byte, with
modifier 2, having the bit corresponding to the line under
test equal to zero, thus causing the desired deviation from
reference.

The result of the test is determined by a modifier 2 (modem
status) read, verifying that the returned byte has the
correct value, thereby verifying that the correct line
responded correctly to the applied stimulus. At this time,
the ability of the Device Specified Jump logic to recognize
a deviation of modem status lines from reference is tested.

Test Condition 2 (reference = 0)

Each output/modem status line is set using similiar test
sequences as described in Test Condition 1, but using the
reverse sense of the reference and stimulus (i.e. the
reference = 0, and the stimulus = 1) for each line, then
verifying that the Device Specified Jump logic also detects
this deviation.

Errors:

The error conditions possible as a result of the stimulus
applied by either test condition 1 or test condition 2
described above are:

    a. The modem status read (modifier 2) shows that the
        induced status change was not detected/ reported,
        and the DSJ did not respond with a status return.
        This would indicate that the fault probably lies in
        the modem status or modifier 2 logic.

"Modem status change not detected Dev = D"

    b. The modem status read shows that the wrong line(s)
        responded to the stimulus. This points at the
        modem status logic.

"Modifier 2 read is !XX expected !YY Dev = D"

    c. The modem status read shows that the induced status
        change was not deteced/reported, but the DSJ did
        respond with a status return. This indicates that
        the fault probably lies in the modifier 2 logic.

"Modifier 2 did not report status change Dev = D"

    d. The modem status read shows that the induced status
        change was detected, but the DSJ logic did not
        respond with a status return. This condition
        points at the Device Specified Jump logic.

"DSJ did not report modem status change Dev = D"

Step 27 - CA-CB,CF   (reference = 1)

    The procedure described above as Test Condition 1 is
    used to test the circuitry for each device associated
    with the lines CA and CB,CF.

Step 28 - CA-CB,CF   (reference = 0)

    The procedure described above as Test Condition 2 is
    used to test the circuitry for each device associated
    with the lines CA and CB,CF.

Step 29 - CD-CC   (reference = 1)

    The procedure described above as Test Condition 1 is
    used to test the circuitry for each device associated
    with the lines CD and CC.

Step 30 - CD-CC   (reference = 0)

>   The procedure described above as Test Condition 2 is
>   used to test the circuitry for each device associated
>   with the lines CD and CC.

Step 31 - SCA-SCF   (reference = 1)

>   The procedure described above as Test Condition 1 is
>   used to test the circuitry for each device associated
>   with the lines SCA and SCF.

Step 32 - SCA-SCF   (reference = 0)

>   The procedure described above as Test Condition 2 is
>   used to test the circuitry for each device associated
>   with the lines SCA and SCF.

Step 33 - CH-CE   (reference = 1)

>   Since the circuitry associated with these lines is
>   located on the Extended ADCC, this step will be
>   performed only if the Extended ADCC is present.

>   The procedure described above as Test Condition 1 is
>   used to test the circuitry for each device associated
>   with the lines CH and CE.

Step 34 - CH-CE   (reference = 0)

>   Since the circuitry associated with these lines is
>   located on the Extended ADCC, this step will be
>   performed only if the Extended ADCC is present.

>   The procedure described above as Test Condition 2 is
>   used to test the circuitry for each device associated
>   with the lines CH and CE.

Step 35 - Modem Status Line Crosstalk (reference = 0)

>   A test is made to check for changes in modem status
>   lines induced by noise or activity on surrounding
>   lines. This is done by setting the reference voltages
>   for all lines to 0, then transmitting all combinations
>   of characters through the loop back connector at 9600
>   baud, checking to be sure that the modem status lines
>   remain at zero reference.

"DSJ returned !XX expected !YY Dev = D"

"Modifier 2 read is !XX expected !00 Dev = D"

Step 36 - Modem Status Line Crosstalk (reference = 1)

> Another test is made to check for induced changes in modem status lines in this step. This time, the reference voltages are set to 1 for all lines, and the previous step's test sequence is repeated.

"DSJ returned !XX expected !YY Dev = D"

"Modifier 2 read is "XX expected !00 Dev = D"
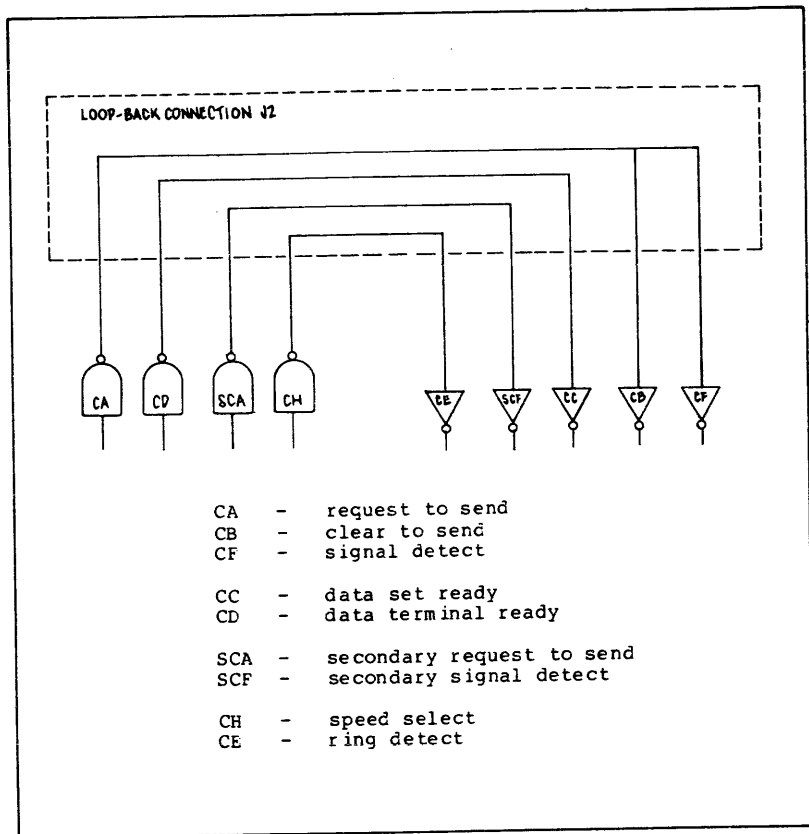
Figure 4.5 - J2 Loopback Connection [Modem Lines]

CA - request to send
CB - clear to send
CF - signal detect

CC - data set ready
CD - data terminal ready

SCA - secondary request to send
SCF - secondary signal detect

CH - speed select
CE - ring detect

4.9   TEST SECTION 9 — DSJ Priority Structure

This Test Section verifies the ADCC Device Specified Jump
priority structure. The ADCC is specified to return various
values as the result of a Device Specified Jump.   If more than
one condition exists at the time DSJ is issued, only the highest
priority condition is returned in the DSJ value. The priority for
DSJ returned values is:

Priority	Value Returned	Condition
highest	2	Status should be read
.	0	Read request pending
.	1	Write request pending
lowest	3	No request pending

Step 37 - DSJ Priority Structure

Each device is set up to request service when the
transmit buffer is empty, a character is present in the
UART receive buffer, or when a modem status line has
changed from reference. A character is transmitted and
received (using the loopback connection). This will
cause the board (when enabled) to request service
because the receive buffer has a character to be read.
Modem status line CC is made to deviate from reference
using the procedure described in Section 8 to cause the
board to request service because the modem status line
has changed from its established reference.

A DSJ is issued and the resulting returned byte is
verified to be a two (highest priority), which is
caused by the modem status line change. The condition
of the board requesting service for a modem status line
change is now masked off using modifier 7.

A DSJ is issued and the resulting returned byte is
verified to be a zero (next highest priority). This
condition should exist because of the character in the
UART receive buffer. Now the board requesting service
due to a character in the receive buffer is masked off
using Service Condition Modifier 3.

A DSJ is issued and the resulting returned byte is
verified to be a one, which is due to the empty UART
transmit buffer. The request due to this condition is
masked off using Service Condition Modifier 3.

A DSJ is issued and the resulting returned byte is
verified to be a three, which means there are no
requests and is the lowest priority.

"DSJ returned !XX expected !YY   Dev = D"

"Modem Status change not detected   Dev = D"

> Where D is the value of the device number under test
> when the error condition is detected, XX is the value
> of the returned byte and YY is the value of the ex-
> pected byte, the condition for which should have held
> off the condition causing the actual return.

4.10   TEST SECTION 10 — Register 2 READ Using Channel Program

The ability of the ADCC under test to return the correct response
to a register 2 read while the ADCC is asserting CSRQ on the IMB
is verified.   This step cannot be done with a software read,
hence a channel program instruction read is used to verify the
sense of bit 10.

Note that this is the first time that a channel program has been
attempted, which will be the first time that the assumed good
channel program processing micro-code will be used.

Step 38 – Register 2 Read using Channel Program

> The channel program instruction READ REGISTER is used
> to read the contents of Register 2. The channel pro-
> gram is then terminated, and the results of the read
> are verified to be !802A. Note the difference between
> the software register 2 read in Step 2 and this channel
> program instruction read. Bit 10 should be a one only
> while CSRQ is being asserted.

"Register 2 read is !XXXX expected !802A"

> XXXX is the value of the returned word.

4.11   TEST SECTION 11 — Identify

This Test Section verifies correct operation of the ADCC identify
logic to, upon request, provide two bytes (unique to ADCCs) used
to identify which devices are present.

Next, the ability of the ADCC to assert the Data Not Valid line
for the ADCC-Extended device numbers (4-7) when not present
during an Identify sequence is tested.

Step 39 – "Software Equivalent" Identify

> A direct I/O write with a special opcode is issued,
> followed by two direct I/O register zero reads which
> simulates the channel program instruction "IDENTIFY".
> This is done by issuing an address to talk to device
> 31, then telling each device to identify and reading
> the two returned bytes.

The two returned bytes are then verified to be !4080, which is the unique identify code of ADCCs.

"Identify device D returned !XXXX expected !4080"

D is the value of the device number under test, and XXXX is the value of the two returned two bytes. (first byte returned in most significant byte)

Step 40 - Identify each device

A channel program identify is executed for each device on the ADCC, and the returned bytes are verified to be !4080.

"Identify device D returned !XXXX expected !4080"

D is the value cf the device number under test, and XXXX is the value of the two returned two bytes. (first byte returned in most significant byte)

Step 41 - DNV set for ID of missing devices

A software equivalent identify is issued for each device (4-7) of the ADCC-Extended, and a test is made using the fact that Data Not Valid being asserted will set condition code CCL in the CPU status register.

"DNV not asserted Dev = D"

D is the value of the device number under test.

Note that this step will not be executed if the ADCC-Extended is present, unless the operator wishes to remove the inter- connecting cable between the ADCC Main and ADCC Extended prior to executing the test. See paragraph 2.3 (Limitations).

4.12 TEST SECTION 12 — Inhibit CSRQ Assertion

The ADCC logic is capable of suppressing the assertion of CSRQ on the IMB. This is done by sending an address to talk or listen for any of the following device numbers: 16, 18, 20, or 22, followed by any one of the following secondary addresses: 17, 19 21, 23, 25, 27, 29, or 31.

Any reset (PON,SRST,etc.) of the ADCC will re-enable CSRQ assertion. Also, CSRQ assertion can be re-enabled by sending an address to talk or listen for any of the following device numbers: 0, 1, 2, 3, 4, 5, 6, 7, or 31, followed by any of the following secondary addresses: 0, 1, 2, 3, 4, 5, 6, 7, or 16. This Test Section places this logic under test.

Step 42 - Inhibit CSRQ

All combinations of the above mentioned primary and secondary address commands are used to insure that all of these combinations will disable CSRQ assertion. The execution of a channel program using the instructions RREG 2 and INT,HLT is attempted. This attempt should not be successful. If, after one of the combinations of primary and secondary commands is used to disallow CSRQ assertion, the execution is successful, the combination will be reported as

"Primary XX, secondary YY does not disable CSRQ. Dev=D"

The Initialize Channel IMB command will be used to re-enable CSRQ after each combination is found to be successful in disabling CSRQ assertion. If the initialize does not re-enable CSRQ assertion:

"INIT does not re-enable CSRQ. Primary XX, secondary YY Dev=D"

Step 43 - Re-enable CSRQ

The primary/secondary combination, address to talk 16, secondary address 17, is used to disable CSRQ assertion, then all combinations of the above mentioned primary and secondary address commands are used to insure that all of these combinations will re-enable CSRQ assertion. The channel program used in the previous step is used in this step to insure channel program execution.

If one of these combinations proves unsuccessful in re-enabling CSRQ assertion:

"Primary XX, secondary YY do not re-enable CSRQ. Dev=D"

The next four steps place under test: the program status register logic and the logic which inhibits CSRQ assertion. This is done with a test for bit 6 of the word returned as a result of an OBSI.

This bit is defined to be a one when a service request is due to a parallel poll, or when there has been a change in channel program running status. The next four steps make a change to the channel program run status, and check this bit by inhibiting CSRQ assertion (using the method defined in the previous test section, then emulating SIOP or HIOP with the set/clear bit of the data lines set or clear as necessary to change the running status.

Step 44 - SIOP with Status Change

> SIOP is emulated with bit twelve of the IMB data lines set to zero. This condition will cause a change in the channel program running status, therefore, bit 6 of the OBSI return will be set to one indicating PPOLL or a change in running status.

Step 45 - SIOP without Status Change

> SIOP is again emulated, this time with bit twelve of the data lines set to one. This condition should not cause the POLL bit to be returned with the channel number as the result of an OBSI.

Step 46 - HIOP with Status Change

> HIOP is emulated this time, with bit twelve of the data lines set to zero. This condition will cause a change in the channel program running status, therefore, bit 6 of the OBSI result will be set to one.

Step 47 - HIOP without Status Change

> HIOP is again emulated. Bit twelve of the data lines is set to one, which will not cause a change in running status and bit 6 of the OBSI result will be set to zero.

Step 48 - Correct SPOLL1 response

> SIOP is again emulated and while the board is awaiting service from the CPU, an SPOLL1 is emulated using the WIOC instruction with a special opcode. The result of this SPOLL1 is tested to insure the correct response by the channel.

4.13 TEST SECTION 13 — ADCC Channel Program

This Section heavily employs the use of channel program processing micro-code, and places the entire ADCC under test. Each logic function block has been placed under test one or more times by the previous test sections.

Step 49 - ADCC Channel Program

> Using a channel program, this step initializes the ADCC, by setting up the various service conditions, baud rates, UART control, modem references and defining a special character. A 256 byte output transfer is started (for each device) with each byte being read (thru the loop back connector) and placed in memory. When the channel programs for all devices are complete, the data in the input buffers is verified.

The possible error conditions of this step and probable causes if definable, are as follows:

Symptom	Probable Cause
channel program did not start	- CSRQ not pulled (ADCC) - CSRQ not recognized (CPU)
channel program did not complete correctly	- overrun or parity error break detected (ADCC) - non-responding module (memory) - channel program abort (CPU) - special character was not detected (ADCC)
channel program timed out	- CSRQ logic
data error	- missing or extra bits or bytes

The test sequence is as follows:

      1. Execute channel programs for all devices present on ADCC.

      2. Wait loop to allow channel program execution.

      3. Test DRT word 3 to insure channel program started.

"Channel program did not start Dev = D"

      4. Test DRT word 3 to insure channel program not stuck in WAIT.

"Channel program stuck in wait Dev = D"

      5. Test DRT word 3 to insure no unexpected value.

"Unexpected value !XXXX found in DRT Dev = D"

      6. Test CPVA halt code for break detected.

"Break detected - Dev = D"

      7. Test CPVA halt code for overrun error.

"Overrun detected Dev = D"

      8. Test CPVA halt code for parity error.

"Parity error detected Dev = D"

9. Test CPVA halt code for unexpected value.

"Unexpected value !XXXX found in CPVA Dev = D"

10. Test CPVA halt code for unexpected DSJ value.

"Unexpected DSJ value returned Dev = D"

11. Test CPVA word zero to insure no CPU abort.

"Channel program aborted by CPU, CPVA=!XXXX Dev = D"

12. Test data output versus data input. (note: if any of the above error conditions exist, the data input/ data output check will not be done)

"Byte !AA is !XX, expected !ZZ Dev = D"

D is the value of the device number under test, XXXX is the value (placed by the micro-code) of the CPVA abort word, AA is the incorrect byte number (0-255), XX is the incorrect value and ZZ is the expected value.

4.14 TEST SECTION 14 — Extended Special Character RAM (optional)

This Test Section places the special character RAM under test using various patterns. Due to the lengthy execution time required for this test section, it will by executed only upon request by the operator.

Step 50 - Walking one

The entire special character RAM space is set to zero, and while a one is "walked" thru selected locations, all other locations are verified to remain at zero.

See next step for error

Step 51 - Walking zero

The entire special character RAM space is set to ones, and while a zero is "walked" thru selected locations, all other locations are verified to remain at one.

"Special Character RAM address XXXX affected by YYYY"

XXXX is the affected address, and YYYY is the address written.

(See Figure 4.6 for device-to-address translation)

```
--------------------------
| DEVICE |     RAM         |
| NUMBER |   ADDRESS       |
|--------------------------|
|        |                 |
|    0   |    0CC  -  255   |
|    1   |    256  -  511   |
|    2   |    512  -  767   |
|    3   |    768  -  1023  |
|    4   |   1024  -  1279  |
|    5   |   1280  -  1535  |
|    6   |   1536  -  1791  |
|    7   |   1792  -  2047  |
|        |                 |
--------------------------
```

Figure 4.6 – Special Character RAM Address Table

4.15   TEST SECTION 15 — Extended Baud Rate/UART Test (optional)

Step 52 - Extended Baud Rate/UART Test

This step gives a more stringent test of the UART logic than Step 15. In this step, various combinations of baud rates and UART control functions are used to provide stimulus to each device in the manner described below.

The test sequence is to set up the various UART control functions (see Figure 4.7) then transmitting, receiving and verifying the test bytes (or part of the byte in case 5, 6 and 7 data bit words) !0,!55,!AA and !CC.

"UART control=!ZZ.   Sent !XX received !YY Baud=XXXX Dev = D"

XX is the value of the test byte, XXXXX is the baud rate being tested, YY is the value of the byte received ZZ is the UART control byte (see Figure 4.7), and D is the value of the current device number under test.

The step is complete when all devices on the ADCC have been tested for all combinations of test bytes and all baud rates.

Note this limitation: The ADCC requires a set-up procedure in order to transmit and receive data. The following describes the minimum required ADCC initialization procedure:

a.   WRITE MODIFIER 1 – UART Control
     A byte written with this modifier establishes UART control
     functions necessary for the UART to transmit and receive
     intelligible data.

b. WRITE MODIFIER 6 - Baud Rate Select Bytes written with this modifier are used for the selection of transmit and receive baud rates.

If, for any reason, the above writes cannot be completed or the control circuitry does not respond correctly to the data byte(s) written with one or both of the above mentioned modifiers, the diagnostic program will be unable to determine whether an unsuccessful data transfer was due to a UART malfunction, a UART control circuitry malfunction or a baud rate control circuitry malfunction.

UART Control	Start Bits	Data Bits	Parity	Stop Bits
!80	1	5	odd	1
!81	1	5	none	1
!82	1	5	even	1
!84	1	5	odd	1.5
!85	1	5	none	1.5
!86	1	5	even	1.5
!88	1	6	odd	1
!89	1	6	none	1
!8A	1	6	even	1
!8C	1	6	odd	2
!8D	1	6	none	2
!8E	1	6	even	2
!90	1	7	odd	1
!91	1	7	none	1
!92	1	7	even	1
!94	1	7	odd	2
!95	1	7	none	2
!96	1	7	even	2
!98	1	8	odd	1
!99	1	8	none	1
!9A	1	8	even	1
!9C	1	8	odd	2
!9D	1	8	none	2
!9E	1	8	even	2

Figure 4.7 - UART Control Byte Table

4.16 TEST SECTION 16 — Channel Address Switch (optional)

This Test Section places the channel address recognition logic under test. This Test Section is executed only upon request by the operator. In order to execute this test section, all other channels present, except for the channel being used for AID communication, must be set to channel 0. The commands ROCL, OBII and OBSI are used to verify each setting of the channel address switch. The operator is asked to set the channel address of the ADCC under test to all values 1 thru 15 (except for the address of the channel being used by DUS) and the following tests are performed.

Since it would mean reloading DUS and this diagnostic program to test all 15 positions of the Channel Address Switch, only 14 positions are tested. The positions not tested are 0, because there is no channel 0, and the position which corresponds to the channel number being used for the DUS console.

"All other Channels except Console must be switched to 0"

Step 53 – ROCL

> The ability of the ADCC under test to respond to the global I/O instruction ROCL is tested. A ROCL is issued, and the resulting returned word is tested to verify that only the bits which correspond to the ADCC channel number under test and the channel number being used by AID are set to ones.

"Set Channel Address to XX"

"ADCC under test did not respond to ROCL"

"ROCL returned !XXXX expected !YYYY"

Step 54 – Obtain Interrupt Information (OBII)

> OBII is a command issued by the interrupt processing micro-code. This step issues a register 0 read with a special command code to emulate OBII. The resulting returned byte, bits 9 thru 12 are verified to contain the value of the channel address of the ADCC under test.

"OBII returned YY expected ZZ" (see Step 55, below)

Step 55 – Obtain Service Information (OBSI)

> OBSI is a command issued by the channel service process micro-code. This step issues a register 0 read with a special command code to emulate OBSI. The resulting returned byte, bits 9 thru 12 are verified to contain the value of the channel address of the ADCC under test.

"OBSI returned YY expected ZZ"

> YY is the value of bits 9 thru 12 in the returned byte
> and ZZ is the value of the channel address of the ADCC
> under test - the expected value.

4.17 TEST SECTION 17 — RS-232-C Cable Test (optional)

This test section places one RS-232-C cable under test using the
RS-232-C ´D´ loop connector (HP 0960-0475) Note that this test
section will be executed only upon request by the operator.

Step 56 - Cable Test - Data Lines

> The operator is requested to input the device number
> which is associated with the cable requiring test. The
> operator responds with the device number (0-7). The
> request is then made for the operator to attach the
> RS-232-C ´D´ loop connector to the cable to be tested,
> and the program pauses. When the operator responds
> ´GO´, a subset of the special character RAM test
> described in Test Section 6, chosen for maximum data
> pattern combinations, is executed for that device.

"Cable X test - transmitted !YY, received !ZZ"

> X is the value of the device number associated with the
> cable under test, YYY is the value of the byte
> transmitted and ZZZ is the value of the received byte.

Step 57 - Cable Test - Status Lines

> The stimulus described as Test Section 6 is used for
> the cable associated with the device number under test.
> The error messages used in Test Section 8 are also used
> for this step.

ADCC Diagnostic
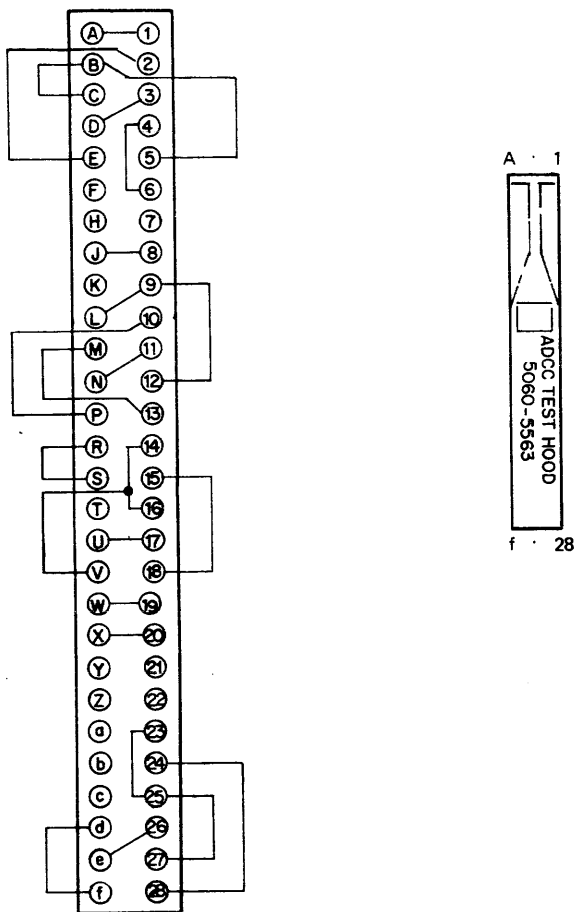
Figure 5.1 - ADCC Loopback Hood ("ADCC TEST HOOD", 5060-5563)
(See also Figures 4.3, 4.5, and 5.2)

Pin Number/Signal Name Cross Reference

Signal Name	Device 0/4 Pin Number	Device 1/5 Pin Number	Device 2/6 Pin Number	Device 3/7 Pin Number
BA	52	33	21	8
BB	56	34	24	5
CA	53	36	23	9
CB	45	27	17	4
CF	49	31	20	6
CC	47	29	19	3
CD	55	35	26	10
SCA	54	38	25	11
SCF	51	37	22	7
CH	39	30	15	2
CE	40	28	16	1

BA	-	transmit data
BB	-	receive data
CA	-	request to send
CB	-	clear to send
CF	-	signal detect
CC	-	data set ready
CD	-	data terminal ready
SCA	-	secondary request to send
SCF	-	secondary signal detect
CH	-	speed select
CE	-	ring detect

Figure 5.2 - J2 Pin Numbers
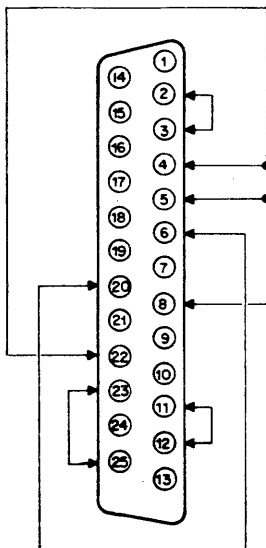
PATHS USED FOR
THIS DIAGNOSTIC

PIN	SIGNAL NAME
2	BB
3	BA
-4	CF
-5	CF
6	CD
-8	CA
11	SCF
12	SCA
20	CC
22	CB
23	CH
25	CE
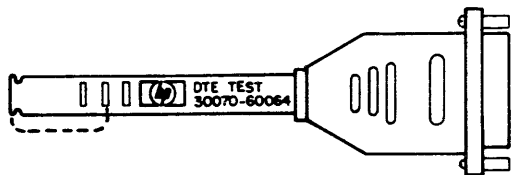
Figure 5.3 - RS-232 D-Type Loopback Connector
("DTE TEST," 30070-60064)

6.0 ERROR INTERPRETATION

The table on the following page relates functions and blocks tested (e.g. channel program execution) to specific tests (Sections in the Diagnostic) and subsystems (ADCC, IMB, CPU, Memory).

There are two ways to use the table:

(1) Enter the table at the failing section - this tells you:

    a. which function or block was under test

    b. the most likely failing subsystems in order of probability

(2) Enter the table at a block or function -- this tells you:

    a. which test section explicity tests that block or function

    b. which test sections depend on that block or function in order to pass

    c. which subsystems cause failure of that block or function

If every block or function were independent of the others and could be tested independently, then the table would have only a diagonal line of ´X´s. Since this is not the case, the table provides a compact guide to these interrelationships -- to deal with the cases in which the subsystem appears to fail due to the failure of other subsystems.

Test Sections

Block or Function Tested	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	Order of Failure Probability
Command Decoding	X	-	-	-	-	X	-	-	-	X	-	-	X	-	-	-	-	ADCC,IMB,BIC
Channel Addr. Recognition	X	-	X	-	-	X	-	-	-	X	-	-	X	-	-	X	-	ADCC,IMB,BIC
Register Decoding	X	-	-	X	-	-	-	-	-	X	-	-	X	-	-	-	-	ADCC,IMB,BIC
Global Resp.	X	X								-		-	X			X		ADCC,IMB,BIC
Device Address		X	-	X	X	-	-	-			-	-	X	-	-			ADCC
Control Logic	-	-	-	-	X	X	X	X	X	-	-	-	X	-	-	-	-	ADCC
Handshake Control	X	-	-	-	-	-	-	-	-	-	-	-	X	-	-	-	-	ADCC
Interrupt Queue			X	X						-	-	-	X					ADCC
Program Run Status Regs.							-			-	-	X	X					ADCC
DSJ Logic				-	-	X	X	X					X	-	-		-	ADCC, Hood
Special Char. Detect.					-	X	-	-	-				X	X	-		-	ADCC, Hood
Modem Status Monitor						X	X	-					X				-	ADCC, Hood
Baud Rate Clock						X	X	-	-	-			X	-	X		-	ADCC, Hood
UARTs						X	X	X	-	-			X	-	X		-	ADCC, Hood
Break Detection							X						X				-	ADCC, Hood
Identify												X	X					ADCC
Modem Controls						-	X						X				-	ADCC, Hood
CSRQ Inhibit										-	-	X	X					ADCC,IMB,BIC
Priority Encoding (Dev)	X	X						-					X				-	ADCC
Poll Logic		-						X			-	-	X					ADCC
Interrupt Mask	X	X								-	-	-	X					ADCC
Priority Encoding (Chan)	X	-								-	-	-	X					ADCC
RS-232-C Cable																	X	Cable

Key X = function or block under test
     - = function or block must work to pass test

Figure 6.1 - Function/Block Reference Table

GLOSSARY OF TERMS

ADCC   –   Asynchronous Data Communications Channel

BA   –   RS-232-C circuit designator for transmit data

BB   –   RS-232-C circuit designator for received data

Break   –   A continuation of spaces (logical zeros) on the
                  received data line for a duration exceeding one
                  character time

CA   –   RS-232-C circuit designator for request to send

CB   –   RS-232-C circuit designator for clear to send

CC   –   RS-232-C circuit designator for data set ready

CD   –   RS-232-C circuit designator for data terminal
                  ready

CE   –   RS-232-C circuit designator for signal detect

CF   –   RS-232-C circuit designator for signal detect

CH   –   RS-232-C circuit designator for speed select

CPP   –   Channel Program Processor

CPU   –   Central Processor Unit

CPVA   –   Channel Program Variable Area

CSRQ   –   Channel Service Request

'D' loop   –   A connector used to test RS-232-C cables

DNV   –   Data Not Valid

DRT   –   Data Reference Table

DSJ   –   Device Specifed Jump

Extended ADCC –   A PCA which, when added to the ADCC, provides
                  circuitry for device numbers 4-7.

ADCC Diagnostic

IMB	–	Inter-Module Bus
INIT	–	Initialize channel instruction
IPOLL	–	Interrupt Poll
J2	–	a frontplane edge connector
Main ADCC	–	A PCA which contains circuitry for control and supports device numbers 0-3.
Modem	–	An instrument which prvides for the transmission of binary serial data on telephone lines
Modifier	–	A secondary command, corresponding to an HP-IB secondary address, which tells the device how to interpret forthcoming data.
Modifier 0	–	Read – Read Data Write – Write Data
Modifier 1	–	Read – UART Status – Write – UART Control
Modifier 2	–	Read – Interface Control
Modifier 3	–	Service Condition Masks
Modifier 4	–	Clear Special Character
Modifier 5	–	Set Special Character
Modifier 6	–	Baud Rate Select
Modifier 7	–	Status/Reference Mask
OBII	–	Obtain Interrupt Information
OBSI	–	Obtain Service Information
Overrun	–	A character was received by the UART with a valid character present in the receive buffer
RAM	–	Random Access Memory
RIOC	–	Read I/O Channel Instruction
RREG	–	Read Register Instruction
UART	–	Universal Asynchronous Receiver/Transmitter chip

HP 3000 Computer System

GENERAL I/O CHANNEL (GIC)
DIAGNOSTIC MANUAL

LIST OF EFFECTIVE PAGES

The List of Effective Pages gives the date of the current edition and of any pages changed in updates to that edition. Within the manual, any page changed since the last edition is indicated by printing the date the changes were made on the bottom of the page. Changes are marked with a vertical bar in the margin. If an update is incorporated when an edition is reprinted, these bars are removed but the dates remain.

All pages original issue . . . . Mar 1982

PRINTING HISTORY

New editions are complete revisions of the manual. Update packages, which are issued between editions, contain additional and replacement pages to be merged into the manual by the customer. The date of the title page of the manual changes only when a new edition is published. When an edition is reprinted, all the prior updates to the edition are incorporated.

First Edition . . . . Mar 1982

| **CONTENTS** |

```
-------------------------------------------------------------------
| CONTENTS (Cont)                                                 |
-------------------------------------------------------------------
```

Paragraph   Page

```
-------------------------------------------------------------------
| ILLUSTRATIONS                                                   |
-------------------------------------------------------------------
```


Title   Page

```
+------------------------------------------------------+ +---------+
|                                                      | | SECTION |
|                           GENERAL INFORMATION  |    | |---------|
|                                                      | |    I    |
+---------------- -------------------------------------+ +---------+
```

1.0 INTRODUCTION

This document describes the diagnostic test program for the
General I/O Channel (hereinafter referred to as the GIC). This
program is designed to verify correct operation of all func-
tions of the GIC provided that all sections, both standard and
extended, are executed. This is accomplished by exhaustively act-
ivating and checking each logical function of the unit under
test. This program is written in the SPL-II language.

1.1 REQUIRED HARDWARE

Special logic functions have been included on the GIC to permit
the state machines to be exercised and meaningful failure data to
be obtained. Activation of these functions via Mode Switch S5,
however, makes the unit under test unsuitable as a device to
communicate with other I/O devices, such as the flexible disc or
magnetic tape unit. Several methods of operation, each requiring
different equipment, are possible.

In configuring the hardware for this diagnostic, there are four
considerations:

(1) Hardware. An HP3000 HP-IB version system with minimum sys-
    tem memory configuration.

(2) Loading. Either the GIC under test must work well enough to
    load DUS or an additional GIC is required.

(3) Isolation from HP-IB. The GIC under test must be discon-
    nected from the HP-IB except in Paragraph 4.25.

(4) Paragraph 4.25. If the GIC-to-GIC test is to be run, a second
    GIC is required to communicate with the GIC under test via
    the HP-IB.

This is summarized in the table and block diagrams which follow:

GIC Diagnostic

```
+-----------------------------------------------+
|    |  GIC |  Add'l|                       |Test|
|Mode|Under |  Good |                       |Sect|
|    |  Test|   GIC |        Comment        | 25*|
+----+------+-------+-----------------------+----+
|    |      |       |                       |    |
| 2a |   X  |       | Only if GIC under     |    |
|    |      |       | test can load         |    |
|    |      |       |                       |    |
| 2b |   X  |   X   | Load through good     | X  |
|    |      |       | GIC                   |    |
|    |      |       |                       |    |
+-----------------------------------------------+
```
*There is enough hardware to run optional Test Section 25.

Figure 1.1   Required Hardware Summary

Mode 2a

```
IMB  _____
              |                                           |
        +-------+                             +-------+
        |  ATP  |                             |  GIC  |
        |  or   |                             | Under |
        | ADCC  |                             | Test  |
        +-------+                             +-------+
            |                                     |
        Terminal                             |---Disc or Tape
```
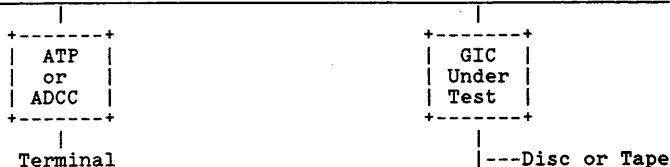
Note: Disconnect Disc or Tape Controller from GIC after loading.
      Use LOOP command to ensure that Diagnostic does not have to
      be continually reloaded.

Mode 2b - Loading and Running Test Sections 1 through 24
```
IMB  _____
          |                    |                       |
          |                    |                       |
      +-------+            +-------+               +-------+
      |  ATP  |            |  GIC  |               | Good  |
      |  or   |            | Under |               |  GIC  |
      | ADCC  |            | Test  |               |       |
      +-------+            +-------+               +-------+
          |                    |                       |
      Terminal           (no connection)          |---Disc
                                                       or
                                                     Tape
```
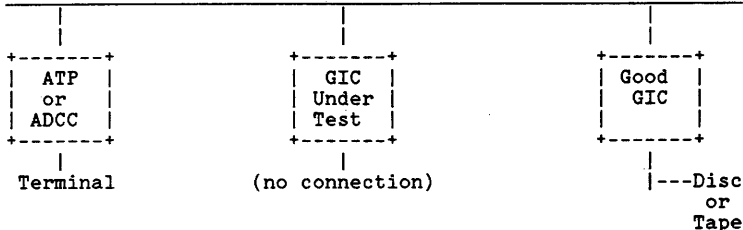
Mode 2b - Running Test Section 25

```
IMB  _____
          |                    |                       |
      +-------+            +-------+               +-------+
      |  ATP  |            |  GIC  |               |Second |
      |  or   |            | Under |               |  GIC  |
      | ADCC  |            | Test  |               |       |
      +-------+            +-------+               +-------+
          |                    |_____|
      Terminal              (no other connection)
```
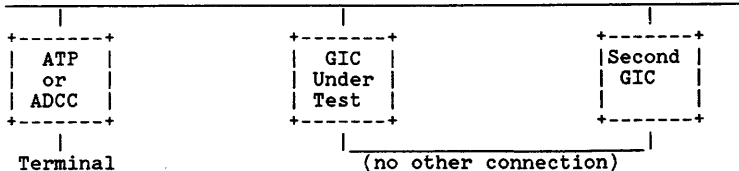
Figure 1.2   Hardware Configuration for Modes 2a and 2b

1-3

1.2 REQUIRED SOFTWARE

Diagnostic/Utility System file system plus the GIC diagnostic program.

1.3 DIAGNOSTIC PROGRAM STRUCTURE

The heirarchy for this diagnostic program is:

Program
  Section
    Step
      Case

where a step is the smallest entity (not necessarily loopable) which contains at least one case and verifies operation of one conceptually separate function. Steps include one or more cases when a single stimulus sets up multiple conditions to be verified.

A case is defined as the entity which verifies proper circuit conditions along a distinct path in the logic of the unit under test. Associated with each case is a subsection of the hardware which is required to be functional for that case to succeed. Cases occur in the program flow wherever program statements have established proper conditions in the unit under test to verify that a function has or has not operated properly. As some cases are used in several steps, case numbers do not appear in a strictly ascending order from the beginning to the end of the diagnostic program. Step numbers, however, are in strictly ascending order, with each step assigned a unique number.

1.4 TESTING PHILOSOPHY

The GIC circuitry is not divisible into simple blocks for diagnosis; many logical paths contribute to proper operation of several board functions.

This diagnostic program is substantially different from other programs of this nature previously produced for Hewlett-Packard data products. This unique design is an attempt to achieve the goal of providing the diagnostic program user with the intimate circuit knowledge of the design engineer. The program and documentation is optimized to provide the user with a maximum amount of useable data about the tests being performed, the circuitry involved in these tests, the circuit elements which are suspected of failure, as well as symptom data about each failure.

This program is used with the "board-swap" method of trouble-shooting. By running the diagnostic, a GIC board is tested to determine if any error occurs. If one does, the board is replaced.

The goals of this diagnostic are achieved by first testing a minimum subset of the logic which must be functional before any data can be obtained from the unit under test. When this section of logic has been verified, another small section is added to this original kernel and tested with it, assuming that the first section tested operates properly. Additional logic functions are included in the cases, each piece being verified before it is used to test adjacent functions. As errors are detected, they are reported in two ways:

(1) Error messages are generated at the time the errors are detected. These may contain expected/obtained values, as well as reporting other symptoms of the malfunction.

(2) Each case which fails or passes is recorded. After the diagnostic has run to completion, the numbers of the cases which have failed are supplied by the program to the user in the Fault Data Summary. One may then correlate this data through the use of the GIC Diagnostic IMS to determine which hardware module is most likely to have failed, which modules are suspected of failure, and which modules have not been involved in cases which failed. One may also use this information with the IMS to trace on a schematic, in a follow-the-dots fashion, the exact path traversed through the logic in the cases that failed.

1.5 TEST LIMITATIONS

Running the standard set of Sections rather than the complete set results in some limitations.

a. Section 25 tests the GIC by communicating on the HP-IB back and forth to a known-good GIC, thus adding tests for:

   (1) The pin-drivers on the PHI chip

   (2) The HP-IB Transceiver chips

   (3) Master-slave handshake contention on the IMB

   (4) Running the DMA machine at higher speed than the remainder of the diagnostic.

   Conditions 3 and 4 cannot be checked by the 9570 test for the GIC.

b. The SWITCHES command adds steps to Section 1 which require the operator to set all switches to all positions, thus adding tests for:

   (1) channel response at CHAN ADDRs not otherwise used (console etc.)

   (2) DEVICE TYPE switch in both positions

   (3) PROCESSOR switch in both positions

The limitations of the complete program come under these general headings: limitations due to the test cases used and limitations resulting from the inability to test certain logic.

a. Limitations due to inability to test certain logic:

   (1) Bus Request Logic is not checked.

   (2) Race conditions involving asynchronous circuitry cannot be checked.

   (3) Output signals from certain DMA states cannot be verified.

   (4) Special PHI handshakes included to provide for early asynchronous PHI handshake assertion and completion cannot be checked.

   (5) Logic to prevent unassertion of memory read requests before completion of the master handshake cannot be verified.

   (6) HSEN on the HP-IB transceivers is not checked.

   (7) Data rate on the HP-IB is never checked.

   Conditions 2, 3, 4, 5, 6, and 7 cannot be checked by the 9570 test for the GIC.

b. Limitations due to test cases

Because the test cases were generated using the idea that each IC output should be "active" at least once (see the GIC Diagnostic IMS for definition of active), no rigorous verification was ever made that all outputs had been verified in each state which they can assume. No rigorous verification was made to determine that adjacent IC pins or bus lines on some internal data paths were not shorted together. The test cases do, however, provide a very thorough and strenuous exercise of the PCA through as many of its possible states as is reasonably practical.

```
+------------------------------------------------------+ +---------+
|                                                      | | SECTION |
|                             OPERATING INSTRUCTIONS   | |---------|
|                                                      | |   II    |
+------------------------------------------------------+ +---------+
```

2.0 INTRODUCTION

Before running this diagnostic be sure that the physical config-
uration (switches on the frontplane) match the logical config-
uration required by the operating system.

There are two modes of operation possible for this diagnostic
program, the standard (default) mode and the optional mode.
To operate in either mode, the following steps must be executed:

Note: Mode switch S5 must be set to TEST position to run diag-
        nostics and then reset to OPER (normal) position when com-
        pleted.

(1) Bring up the Diagnostic/Utility System (DUS).

(2) The DUS prompt character (:) is displayed.

(3) Respond 'GICDIAG', to load the Diagnostic.

(4) The GIC Diagnostic program displays its title message and
    prompt character '>'.

The next step (step 5) depends on whether you want the Standard
Mode or not.

2.1 STANDARD MODE

(5)   If the standard mode is to be executed, the operator res-
      ponds 'GO' and diagnostic execution begins.

      The standard mode is defined as follows:

      (a) Execute all Diagnostic Sections except Sections 24 and
          25.

      (b) Display error, information, and prompt messages.

      (c) Pause on errors and prompts.

2.2 EXTENDED MODE

(5)   If the optional mode is to be executed, the operator may
      input one or more of the commands discussed in paragraph
      2.3.

      After all desired options have been entered, the operator
      enters 'GO', and the diagnostic will begin execution.

The diagnostic will run until an error condition is detected
or all selected sections have been executed.

2.3 GIC DIAGNOSTIC COMMANDS

These commands are used when the GIC diagnostic is not in the run
mode (i.e., it is in a pause or wait state) and they are usually
executed before the GO command.

OUTPUT AND PAUSES

PRINTER- print error messages and the Fault Data Summary.

*EEPR   - enable error messages.
 SEPR   - suppress error messages.

*ENPR   - enable non-error messages.
 SNPR   - suppress non-error messages.

*EEPS   - enable pauses after error messages.
 SEPS   - suppress pauses after error messages.

 RST   - reset message and pause commands to default state;
          supercede PRINTER command.

*Default Value

TEST SELECTION

SWITCHES- execute switch test portions of Section 1

NOSWITCHES- supercedes SWITCHES command

TEST   - change from the default set of section execution:
          'TEST 1,5,8' -- execute sections 1,5 and 8.
          'TEST 1/3,8' -- execute sections 1,2,3, and 8.
          'TEST +3,6' -- add sections 3 and 6.
          'TEST -3,6' -- remove sections 3 and 6.

PROGRAM CONTROL

GO   - continue diagnostic execution from a pause.

EXIT   - stop diagnostic execution and return to DUS.

RUN   - restart execution of diagnostic at the beginning.
          (supercedes SWITCHES and PRINTER commands;
          does not affect LOOP and TEST commands)

LOOP   - loop on the selected sections.

LOOPOFF- supercedes the LOOP command.

```
+---------------------------------------------------+  +---------+
|                                                   |  | SECTION |
|                                 EXECUTION TIMES   |  |---------|
|                                                   |  |   III   |
+---------------------------------------------------+  +---------+
```

3.0 INTRODUCTION

The following information provides timing information for all
test sections and information regarding the execution status of
each test section.

```
+------+--------------------+----------------------------------+--------+-------+
|Std. |Sect   Steps          | Description                      |  Time  |Notes  |
|------+--------------------+----------------------------------+--------+-------|
|  *   |  1       1-19        | Switches and Interrupts          |        |   4   |
|  *   |  2      20-21        | PHI Verification                 | 3 sec  |       |
|  *   |  3      22-24        | Partial CSRQ Test                |  ms    |       |
|  *   |  4      25-27        | Partial CSRQ Test                |  ms    |       |
|  *   |  5      28-29        | Parallel Poll Priority Encoder   |  ms    |       |
|  *   |  6      30-34        | Partial CSRQ Test                |  ms    |       |
|  *   |  7      35-37        | Registers 8, 9, and 10           |  ms    |       |
|  *   |  8      38-57        | DMA State Machine                |  ms    |   1   |
|  *   |  9      58-62        | Right Output DMA Transfer        |  ms    |   1   |
|  *   | 10      63-65        | Right Output DMA Transfer        |  ms    |   1   |
|  *   | 11      66-77        | Left Input DMA Transfer          |  ms    |   1   |
|  *   | 12      78-82        | DMA Input Transfers              |  ms    |   1   |
|  *   | 13      83-85        | Right to Left Byte DMA Path      |  ms    |   1   |
|  *   | 14      86-87        | DMA Wait States                  |  ms    |   1   |
|  *   | 15      88-89        | DMA Wait States                  |  ms    |   1   |
|  *   | 16      90-91        | Address Rollover                 |  ms    |   1   |
|  *   | 17         92        | IMB Memory Timeout               |  ms    |   1   |
|  *   | 18      93-94        | Parity Error Abort               |  ms    |   1   |
|  *   | 19         95        | DNV Assertion                    |  ms    |   1   |
|  *   | 20         96        | DNV Assertion                    |  ms    |   1   |
|  *   | 21         97        | New Status when not CACS         |  ms    |   1   |
|  *   | 22         98        | DNV Assertion                    |  ms    |   1   |
|  *   | 23     99-100        | DMA Loopback Tests               |  ms    |   1   |
|      | 24    101-104        | Timeout Logic                    | 4 sec  |   2   |
|      | 25        105        | GIC-to-GIC Transfers             |40 sec  |  2,3  |
+------+--------------------+----------------------------------+--------+-------+
```

Figure 3.1 Table of Sections

3-1

3.1 NOTES

total running time per pass:
    standard set = 3 seconds
    complete set = 5 minutes including operator cabling

* Part of standard set of Sections

(1) DMA machine is single-stepped by the slave
    handshake. Refer to Section VI for details.

(2) Channel program microcode utilized.

(3) Requires both the GIC board under test and a known
    good GIC board dedicated to use as a communication
    target for the GIC-to-GIC transfers. Refer to
    paragraph 1.1 for details.

(4) If "SWITCHES" option is selected, operator
    intervention is required to test switch positions.

```
+-----------------------------------------------------+ +---------+
|                                                   | | SECTION |
|                         TEST DESCRIPTIONS | |---------|
|                                                   | |   IV    |
+-----------------------------------------------------+ +---------+
```

4.0 INTRODUCTION

This section of the manual will provide you with a fair description
tion of the test sections and the steps within each step along
with possible error conditions, error messages, and prompt mes-
sages.

4.1 SECTION 1 SWITCHES AND INTERRUPTS

The portions of this section which require all switches to be set
to all positions are optional. They are run only when the oper-
ator desires to verify the ability of all the switches mounted on
the GIC to correctly respond in all positions.   These portions
also test the circuitry associated with these switches. This
option is selected by the "SWITCHES" option at runtime. Data for
the Fault Data Summary is collected during this section only when
the "SWITCHES" option has been selected.

Step 1 - Verifies that the channel under test responds to ROCL
           at the address to which its CHAN ADDR switch is set. It
           also verifies that the channel responds at only one
           address.

           Case 1 Verifies channel addresses 1 through 7.

           Case 2 Verifies channel addresses 8 through 15.

Step 2 - Case 3 Verifies that the channel number set on the
           CHAN ADDR switch is the same as that read from register
           D by OBII.

Step 3 - Case 4 Verifies that the channel number received by
           OBSI is the same as that set on the CHAN ADDR switch.

Step 4 - Verifies that bits 0 and 3-15 of register E (CHANNEL
           CONFIGURATION REGISTER) are read as zeroes and that the
           DEVICE TYPE and PROCESSOR switches operate properly.
           Should this step indicate that both switches are
           inoperable   (a   highly   unlikely   situation),   the
           diagnostic is exited. Should manual checking indicate
           that the switches are operational, it is likely that
           register E cannot be read.

           Case 5 Verifies bits 0 and 3-15 of register E.

           Case 6 Verifies both switches.

           Case 7 Verifies the DEVICE TYPE switch.

Case 8 Verifies the PROCESSOR switch.

Step 5 - Verifies SYS CTRL switch.

Step 6 - Case 9 Executed only if interrupts are found to be set in register C at this point by reading this register. Verifies that such interrupts are cleared by the IMB INIT command.

Step 7 - Verifies that interrupts may be set for all device numbers and that each device greater in priority than device 7 has priority over device 7. Device numbers not responding properly are reported through error messages.

Case 10 Verifies devices 0 through 3

Case 11 Verifies devices 4 through 7

Case 14 Verifies that the proper device number is received from OBII and that the NOT VAL bit is clear, indicating valid interrupt data.

Step 8 - Case 12 Verifies that the GIC does not assert IRQ when interrupts have been set but SMSK has disabled IRQ.

Step 9 - Case 13 Verifies that the GIC under test responds properly to IPOLL when interrupts have been set and SMSK enables IRQ. Interrupts are turned off at the CPU for this step.

Step 10 - Case 12 Verifies that the GIC does assert IRQ when interrupts have been set and SMSK has enabled IRQ. Case 14 Verifies that the proper device number is received from OBII.

Step 11 - Checks that IRQ was cleared by interrupt service.

Step 12 - Case 9 Verifies that IMB INIT clears pending interrupts.

Case 15 Verifies that writing to register C does not assert DNV on the IMB.

Step 13 - Case 16 Verifies that INIT clears the interrupt mask for the channel under test by setting an interrupt for device 7 and verifying that IRQ is not asserted on the IMB.

Step 14 - Verifies that setting NEW STATUS using SIOP clears the NOT VAL bit in register F. Were the MODE switch in the OPERATE (in) position at this time, clearing bit 12 of register F would not inhibit setting CSRQ when NOT VAL

is cleared by this step, causing a system crash. Therefore, before the SIOP is executed to set the NEW STATUS, the value of this switch is read, and if incorrectly set, the operator is prompted until he corrects the setting (to the out position) before the diagnostic will proceed. The NEW STATUS is not set if NOT VAL, register F, is already cleared when tested before SIOP, but the remainder of this step is executed.

The remainder of this step verifies that NOT VAL is cleared by SIOP setting NEW STATUS, that the proper device number appears in register F, that only a DEV REQ is indicated, and that bits 0-4, register F, are zeroes.

This step is executed eight times, starting with device number 7 and counting down, and therefore verifies that the priority encoder functions properly for this particular sequence of NEW STATUS bits set.
Case 17 Verifies the NOT VAL bit is cleared.

Case 20 Verifies proper device number (checks the priority encoder).

Case 18 Verifies SRQ, CHAN REQ cleared, DEV REQ set.

Case 19 Verifies that bits 0-4 are zero.

Step 15 - Case 20 Verifies that SIOP can clear the NEW STATUS previously set for device 0.

Step 16 - Case 20 Verifies that HIOP can clear the NEW STATUS for device 1.

Step 17 - Case 21 Verifies that the GIC under test responds to SPOLL2 at its proper channel address when MYCSRQ is asserted.

Step 18 - Case 22 Verifies that the GIC under test responds to SPOLL1 at its proper channel address when MYCSRQ is asserted.

Step 19 - Case 23 Verifies that OBSI returns the proper information previously set up by the series of steps preceeding this one. The data returned by OBSI should be !027X where X may be !8, !9 or !A. If all tests involving SIOP and HIOP have passed, OBSI should return !027A.

4.2 SECTION 2 – PHI VERIFICATION

Section 2 contains the PHI diagnostic.

Step 20 – Case 24   Verifies that INIT   clears   the   NEW   STATUS
register.   Note, however, that the conditions that are
being cleared are the conditions set by section 1.
Although   section   2 may be looped and no error condi-
tion created by not   preceeding it with section 1,
neither   will   this   step provide any meaningful data
about the ability of INIT to clear NEW STATUS.

Step 21 – Case 25   Checks the PHI in off-line mode by writing
all   bit combinations to the registers and reading the
results.   None of the non-data lines whose outputs
drive GIC circuitry (DMARQ, etc.) are checked.   This
test runs approximately one minute. Failure of this
test   does not necessarily indicate a bad PHI, as much
of the GIC logic is involved in the reads and writes
to   the   PHI.   If a replacement PHI yields the same
symptoms, the problem is probably elsewhere on the
PCA.   If this step fails, the diagnostic is automati-
cally exited, as many of the tests performed after
this point would be meaningless.

4.3 SECTION 3 – PARTIAL CSRQ TEST

This section tests part of the CSRQ logic.

Step 22 – Case 26   Verifies that the DMA BUSY and PHI INT bits
in register B are both cleared. If this step fails,
further testing is probably meaningless, so the diag-
nostic is exited automatically.

Step 23 – Case 27   Verifies that no service requests are pending
on the GIC under test. Should this test fail, the
diagnostic   is exited, as further testing is probably
meaningless.

Step 24 – Verifies that setting the DO SRQ bit in register F
causes   only   the SRQ bit in register F to be set, the
NOT VAL bit cleared, and device 7 to be indicated.
This tests a portion of the CSRQ logic.

Case 28   Verifies SRQ set, CHAN REQ, DEV REQ cleared.

Case 29   Verifies NOT VAL cleared.

Case 30   Verifies device number equals 7.

4.4 SECTION 4 - PARTIAL CSRQ TEST

This section tests more of the CSRQ logic.

Step 25 - Case 31 Verifies that writing device number 3 to register F causes device number 3 to be read from register B.

Step 26 - Causes a simulated parallel poll response for device 7 on the HP-IB and then verifies that this response causes NOT VAL to be cleared, device 3 to be indicated, and only CHAN REQ to be asserted in register F.
Case 32 Verifies NOT VAL is cleared.
Case 33 Verifies device number equals 3.
Case 34 Verifies only CHAN REQ asserted.

Step 27 - Performs an OBSI, which should receive the parallel poll response device number 7 instead of the DMA device 3. The OBSI should show the NOT VAL bit cleared and a DEV REQ only.

Case 38 Verifies device 7.

Case 35 Verifies NOT VAL cleared.

Case 36 Verifies DEV REQ only.

4.5 SECTION 5 - PARALLEL POLL PRIORITY

The parallel poll priority encoder is checked by this section.

Step 28 - Case 37 Verifies that the INIT command clears the DMA device number read from register B.

Step 29 - Case 38 Partially tests the parallel poll priority encoder. To do this, a parallel poll response is generated for device 7 with each of the other device numbers, one at a time. Since device 7 has lowest priority, the other device numbers will be read by OBSI. This is a partial test only, since there are many conditions in the priority encoder not checked by this test.

4.6 SECTION 6 - PARTIAL CSRQ TEST

This section tests certain paths in the CSRQ logic. The condi-
tion of the channel is altered and the effect observed by noting
the state of the NOT VAL bit in register F.

Step 31 – Case 40 The NO POL bit, register F, is set and
verified by reading register B. If this bit cannot be
set, step 32 is bypassed as the CSRQ function it tests
cannot be activated.

Step 32 – Case 41 Reads register F to verify that NO POL, set by
the previous step, set NOT VAL. Since the ability to
clear the NOT VAL bit by asserting SRQ has already been
tested by Step 30, the only logic included in this test
is the data path from the NO POL bit to the IMB through
the read register F logic.

Step 33 – Case 42 Verifies that asserting DMARQ causes the CSRQ
logic to assert NOT VAL in register F. First the NO POL
bit, register B, is cleared, then register 6 is written
with !0002 to generate DMARQ from the PHI. Register F
is read to verify NOT VAL set.

Step 34 – Case 43 Clears DMARQ by writing !0 to register 6 to
clear NOT VAL, then sets NOT VAL by clearing EOI (by
writing !405C to register 0). Reading register F
verifies that NOT VAL is set.

4.7 SECTION 7 - REGISTERS 8,9, AND A

This section verifies that each bit in registers 8, 9 and A can
be written and read independently of all other bits in those
registers. Testing these registers also checks the IMB and GIC
data buffers for adjacent data line shorts.

Step 35 – Verifies that bits in register 8 are independent and
operational.

Case 44 Verifies bits 12-15, register 8.

Case 45 Verifies bits 8-11, register 8.

Step 36 – Verifies that bits in register 9 are independent and
operational.

Case 46 Verifies bits 12-15, register 9.

Case 47 Verifies bits 8-11, register 9.

Case 48 Verifies bits 4-7, register 9.

Case 49   Verifies bits 0-3, register 9.

Step 37 - Verifies that bits in register A are independent and
operational.

Case 50   Verifies bits 12-15, register A.
Case 51   Verifies bits 8-11, register A.

Case 52   Verifies bits 4-7, register A.
Case 53   Verifies bits 0-3, register A.

4.8 SECTION 8 - DMA STATE MACHINE

This section begins the test of the DMA state machine.   To test
this sequential state machine, which is normally driven by an
independent oscillator, the DIAGNOS bit in register F is used to
disable the state machine oscillator, and allow the timing of the
DMA machine to be synchronized with the slave handshake.   The
state machine is clocked by the slave handshake, one clock to the
DMA machine for every slave handshake performed.   Three slave
handshakes are required for one state time of the DMA machine.
Before starting DMA, the DMA control register is checked to see
that all bits operate properly. Short DMA transfers are relied
upon for all synchronized DMA operations in this and later
sections.

Step 38 - Case 54 Verifies that the DMA EN bit, register 8, is
not set when this section of the diagnostic is begun.

Step 39 - Verifies that bits 9-11 of register B can be set by
writing to register F, and that writing to register F
does not start DMA (indicated by DMA BUSY, register B,
not being set).

Case 55   Verifies DMA BUSY is not set.

Case 58   Verifies RT BYT bit.

Case 56   Verifies NO END bit.

Case 57   Verifies DMA OUT bit.

Step 40 - Verifies that bits 9-11, register B, can be cleared by
writing to register F.

Case 58   Verifies RT BYT bit.

Case 56   Verifies NO END bit.

Case 57   Verifies DMA OUT bit.

Step 41 - Sets up a synchronized DMA transfer, an output transfer
starting on the left byte, with initial byte count of

zero, with NO END set, NO POL cleared, for device 3.
The starting memory address is !FFFE. DMARQ assert-
ed throughout the transfer by loading register 6 with
!0002. The DMA EN bit, register 8, is checked to veri-
fy that starting DMA sets it. The DMA BSY bit, regis-
ter B, is checked to verify that it is set. PAR ERR,
ADR OVF, MEM TIM, and DMA STATUS are verified to be
cleared. Register E is written to the DMA transfer to
verify the abort path. Register 8 is read to assure
that state 26 is reached within 10 slave hand-shake
operations of the write to register B.
Case 59 Verifies that write register B set DMA EN,
register 8.

Case 60 Verifies that write register B set DMA BSY.

Case 61 Verifies that the PAR ERR bit can be cleared.

Case 62 Verifies that the ADR OVF bit is cleared.

Case 63 Verifies that the MEM TIM bit is cleared.

Case 64 Verifies DMA STATUS can be cleared.

Case 65 Verifies DMA state progression from state 0 to
26.   Failure of this test terminates diagnostic
execution.

Step 42 - Case 66 Verifies the CSRQ information from register F
at this point in the DMA transfer. Although DMA was
aborted between Case 64 and Case 65, DMA ABT is not yet
set, as the DMA machine has not yet entered state 5.
Contents of register F should be !00FF.

Step 43 - Verifies that DMA BSY is still set, but that DMA STATUS
indicated 11 (abort). The abort occurred between Case
64 and Case 65.

Case 67 Verifies DMA BSY still set.

Case 68 Verifies DMA STATUS is 11.

Step 44 - Verifies state progression from state 26 to state 5. It
also verifies that writing to register E cleared the
DMA EN bit in register 8.

Case 69 Verifies DMA EN cleared.

Case 70 Verifies DMA state progression from state 26
to state 5. Failure of this test causes the diagnostic
to be exited.
Step 45 - Case 71 Verifies the CSRQ information in register F at
this point in the DMA transfer. Register F should
contain !09FB. For this test to pass, the DMA machine

must have reached state 5, and the PHI have asserted its INTERRUPT line. To insure that PHI INTERRUPT will be asserted, register 3 was loaded with !FFFF before this transfer was initiated. If this test fails, the diagnostic is exited, since the CSRQ logic has now been thoroughly tested and should be diagnosable. This allows for simplification of the following test data.

Step 46 - Case 72 Verifies that the RT BYT bit was toggled during state 26.

Step 47 - Verifies that the DMA BYTE COUNT register, register A, previously loaded with !0, was decremented to !FFFF following state 26. This value was chosen to verify that the ripple clock outputs properly decremented the succeeding clock stages. This also verifies that the counter bits may be set by the clock inputs. This does not verify that the bits in the counter are completely functional, as some bits may conceivably set but not clear when the counter is clocked.

Case 73 Verifies bits 12-15, register A, for !F.

Case 74 Verifies bits 8-11, register A, for !F.

Case 75 Verifies bits 4-7, register A, for !F.

Case 76 Verifies bits 0-3, register A, for !F.

Step 48 - Case 77 Verifies that the state machine remains in state 5 for multiple state times until OBSI is executed to the channel. If register 8 indicates state 5, this is verified, since it has been more than one state time since state 5 was executed. (State 5 was entered before case 71). If this step fails because the wait for OBSI fails, the state read will be either 0 or 4. If this test fails, diagnostic execution is terminated.

Step 49 - Case 79 Performs a read of register 2 while DMA is active, and verifies that this causes DNV to be asserted in the IMB.

Step 50 - Case 78 Issues OBSI to the channel under test, then clocks the state machine one state time, and verifies that state 4 is then entered. This checks that OBSI allows progression from state 5 to state 4. If this test fails, diagnostic execution is terminated.

Step 51 - Case 80 Performs a read of register F while DMA is active and verifies that DNV is not set on the IMB.

Step 52 - Case 81 Verifies that three DMA clock cycles after state 4 was entered, state 0 has been reached. If this test fails, diagnostic execution is terminated.

Step 53 - Case 82 Verifies that starting a left byte output DMA
         transfer drives the state machine through state 26 to
         state 27 from state 0. If either state 26 or 27 is not
         reached, diagnostic execution is terminated.

Step 54 - Verifies the state machine progression from state 27
         through state 17 to state 16. The transition from
         state 17 to 16 depends on the assertion of MEMDN and
         the unassertion of CNT0.   During this step, DMA is
         aborted by a write to register E. If improper states
         are reached diagnostic execution terminates.

         Case 81 Verifies transition from state 27 to 17.

         Case 83 Verifies state 17 to 16 transition.

Step 55 - Case 84 Verifies DMA state machine progression from
         state 16 to state 24.   This requires assertion of
         IOEND, indicating a PHI handshake completion. However,
         no check of the data is made so this step could succeed
         and the PHI data handshake still be defective. If an
         incorrect state is reached, execution is terminated.

Step 56 - Case 85 Verifies that register A (DMA BYTE COUNT) has
         been decremented twice since this DMA started, from
         !FFFF to !FFFD. The starting value was the result of
         the preceeding transfer in this section. If !FFFD is
         not read, execution terminates.

Step 57 - Case 86 Verifies that the write to register 14 execut-
         ed during step 54 causes the state machine to progress
         from state 24 to state 5. If state 5 is not reached,
         the diagnostic is terminated. OBSI is also issued and
         the progression from state 5 to state 4 is verified.
         Failure of the state machine to reach state 4 causes
         diagnostic execution to terminate, but test 86 is not
         marked as failing in that case.

4.9 SECTION 9 - RIGHT OUTPUT DMA TRANSFER

This section sets up and executes a synchronized output DMA
transfer of two bytes, starting with the right byte, which ter-
minates normally on a count-equal-zero condition.   Refer to Sec-
tion VI for an explanation of synchronized DMA transfers.

Step 58 - Case 87 Asserts DMARQ from the PHI by writing !2 to
         register 6 to cause DMARQ to look for outbound FIFO not
         full (always true for a two-byte transfer), sets up
         registers 8,9, and A to !0, !CFFF, !FFFD respectively,
         starts the DMA transfer, and synchronizes the program
         with the DMA clock.   It verifies that DMA goes from
         state 0 to state 28. Reaching an incorrect state ter-
         minates the diagnostic.

Step 59 - Case 88 Verifies state progression from state 28 to 24. This progression requires the assertion of MEMDN by the master handshake. This test does not verify that any data has been read as a result of this handshake. Failure to reach state 24 terminates diagnostic execution.

Step 60 - Case 89 Verifies that DMA will go from state 24 to 25. This requires assertion of both DMARQ and DMAENF. If this test fails, execution terminates.

Step 61 - Verifies DMA state machine progression from state 25 through state 29 to state 30, and verifies that register 9 (DMA ADDRESS REGISTER) was incremented to !D000. Arrival in incorrect states will terminate diagnostic execution.

Case 81 Verifies progression from state 25 to 29.

Case 90 Verifies bits 12-15, register 9, are !0.

Case 91 Verifies bits 8-11, register 9, are !0.

Case 92 Verifies bits 4-7, register 9, are !0.

Case 93 Verifies bits 0-3, register 9, are !D.

Case 94 Verifies progression from state 29 to 30.

Step 62 - Transfers the second byte of this transfer to the PHI. It verifies state changes from 30 through 26, 27, 17, 1 and 5 to 4. State changes occuring incorrectly terminate the diagnostic. Fault data is not entered for state changes that have already been verified, but the failures are reported through error messages. The byte count equal zero condition to terminate the transfer is created artificially by writing !0 to register A during this step.

Case 95 Verifies change from state 30 to 26.
Case 83 Verifies change from state 17 to 1.
Case 95 Verifies change from state 1 to 5.
Case 96 Verifies DMA STATUS is 0, indicating normal termination.

4.10 SECTION 10 – RIGHT OUTPUT DMA TRANSFER

This section performs a test of the DMA state machine to verify
state transitions in the output, right byte branch of the DMA
state diagram. State transitions, once successfully completed,
but which now fail are reported by error messages, but no data is
entered into the Fault Data Summary. All incorrect state changes
terminate execution of the diagnostic. Refer to Section VI for
an explanation of synchronized DMA transfers.

Step 63 – Case 94 An output transfer of 1 byte, to start on the
right byte, is initiated by a write to register B.
DMARQ is held asserted by writing !2 to register 6.
States 28, 24, 25 and 29 are traversed without FAULT
DATA reporting, but errors are reported through error
messages. These transitions depend on assertion of
MEMDN, DMA EN, and DMARQ. The change from state 29 to 7
requires assertion of CNT0. Failure of this transition
is reported in the FAULT DATA SUMMARY if this failure
occurs. If any of these state transitions fail,
execution is terminated.

Step 64 – Case 95 Verifies the state transition from state 7 to
state 5. This transition depends on the assertion of
IOEND. However, the data output is not verified, and
this test succeeding does not indicate that the PHI
will properly receive data. Failure of this step
terminates execution.

Step 65 – Case 97 Executes OBSI to the channel under test and
clocks the state machine to state 4. It verifies that
DMA EN is cleared when state 4 is entered. Failure to
enter state 4 terminates diagnostic execution.

4.11 SECTION 11 – LEFT INPUT DMA TRANSFER

This section performs a test of a DMA input transfer, receiving
and writing only 1 byte. This requires that a whole word be read
from memory, the proper byte be replaced by the byte input from
the PHI, and the word be written back into memory. This allows
testing of some of the byte packing and unpacking logic,
decrementing of the byte count, and setting of proper status.

Step 66 – Case 87 Sets up the DMA transfer by programming the
PHI for loop-back (!60 to register 7), and loading !CC
into the inbound FIFO, by loading memory address !CFFF
with !3333 and by writing the proper data to register B
to start the transfer. The DMA clock is synchronized.
Arrival in state 8 from state 0 is verified. If this
state is not entered, the diagnostic is exited.

Step 67 – Case 98 Verifies that the state machine can go from
state 8 to state 9. This depends on assertion of DMA

EN and DMARQ. Special circuitry is required to keep
DMARQ asserted after DMIOGO1 is asserted when operating
a synchronized transfer. This logic may cause failure
in the synchronized DIAGNOS mode but not in normal
operation. Failure of this state transition   will
terminate program execution.

Step 68 - Case 73 Verifies that state 8 decremented the byte
          count to !0.

Step 69 - Case 81 Verifies the state change from state 9 to 20.
          Failure of this test will terminate program execution.

Step 70 - Case 99 Verifies assertion of IOEND as a result of
          DMIOGO1 in state 20. State 22 cannot be reached until
          IOEND is asserted. Data from this PHI handshake is
          checked later in the diagnostic, but failure of the
          master handshake or byte packing circuitry could mask a
          correct data transfer.   Failure   will   terminate
          execution.

Step 71 - Case 100 Verifies the state transition from state 22 to
          18.   This is dependent on assertion of CNTO, indicating
          the detection of the byte count being decremented to 0.
          Failure of this test terminates diagnostic execution.

Step 72 - Case 101 Verifies that DMA STATUS, set during step 71,
          is 10. If this test fails, it might indicate PHI mal-
          function.

Step 73 - Case 81 Verifies state transition from state 18 to 19.
          If this test fails, the diagnostic terminates.

Step 74 - Case 88 Verifies the state transition from state 19 to
          23.   This requires the assertion of MEMDN in response
          to DMRDRQ. Failure of this test terminates diagnostic
          execution.

Step 75 - Verifies state machine progression from state   23
          through 21 to 5.   Arrival in state 5 indicates that
          DMWRRQ generated MEMDN. This step does not check the
          validity of the data sent. Failure to arrive in the
          correct states   will   cause   termination   of   the
          diagnostic.

          Case 81 Verifies transition from 23 to 21.

          Case 102 Verifies transition from 21 to 5.

Step 76 - Verifies that the word in memory accessed by this DMA
          (location !CFFF) has had the left byte written as !CC
          and the right byte left untouched as !33. This is the
          first time that the data from DMA reads and writes has
          been checked so extensive blocks of circuitry are

implicated should this step fail. If it does fail, diagnostic execution is terminated.

Case 103 Verifies left byte is !CC

Case 104 Verifies right byte is !33

Step 77 - Case 105 Verifies that the IMB command INIT will clear the state to zero when executed.

4.12 SECTION 12 - DMA INPUT TRANSFERS

This section performs several DMA transfers to test various paths through the DMA state diagram.

Step 78 - Verifies that register 9 is incremented, register A is decremented and DMA STATUS set to 10 when this right-byte input transfer is performed. States traversed are 6, 2, 10, 11, 15, 13, 5, 4 and 0, in that order. State transition failures are reported through error messages, but data on these failures is not reported in the FAULT DATA SUMMARY. However, when an incorrect state is entered, the diagnostic execution is terminated. Correct decrementing of register A is indicated by a correct transition from state 15 to 13.

Case 101 Verifies DMA STATUS is 10

Case 106 Verifies DMA ADDRESS is !D000

Step 79 - Case 107 Verifies that the right data byte is properly read from the PHI and written into the proper word in memory (location !CFFF). The right byte should be !44. Failure terminates execution.

Step 80 - Case 108 Verifies that the left data byte of the memory word (location !CFFF) was restored by the DMA machine as it was before the transfer. The left data byte should be !33.

Step 81 - Case 70 Initiates a DMA input transfer, starting on the left byte, and then aborts it by writing to register E to verify that the state machine can go from state 8 to 5. OBSI is then issued, and DMA driven to state 4. Incorrect state transitions cause execution to terminate.

Step 82 - Verifies that the DMA state machine detects the condition where CNT0 is not asserted by starting an input transfer of more than 1 byte on the left byte, allowing it to progress from state 0 through states 8, 9, 20, and 22 to state 10. Arrival in state 10 indicates CNT0 not asserted was detected. A write register

E in state 20 causes the next state after 10 to be state 18, indicating the abort succeeded.

Case 100 Verifies CNT0 unasserted

Case 70 Verifies state 10 to 18 transition due to WREG E.

4.13 SECTION 13 - RIGHT TO LEFT BYTE DMA PATH

This section checks the path from state 15 to state 14. (Refer to Section VI for notes on synchronized DMA.)

Step 83 - Case 109 Verifies DMA state transitions from state 0 through states 6, 2, 10, 11, 15, and 14. Arrival in state 14 requires that CNT0 be unasserted and MEMDN be asserted. Only the arrival in state 14 is checked, and failure to arrive in state 14 will cause termination of the diagnostic. This path is checked by an input transfer starting on the right byte.

Step 84 - Case 81 Verifies the state transition from state 14 to state 8. Failure to reach state 8 terminates execution of the diagnostic.

Step 85 - Case 106 Verifies that state 14 incremented register 9 (DMA address register) to !D000. This checks the ripple clock propagation to all states.

4.14 SECTION 14 - DMA WAIT STATES

This section checks various wait states where the DMA state machine is to wait for the assertion of DMARQ. These are checked by removing DMARQ and then entering those states and verifying that those states are not exited. (Refer to Section VI for notes on synchronized DMA.)

Step 86 - Case 110 Starts a right byte output transfer and verifies that the state machine hangs in state 24. Failure to remain in state 24 terminates program execution.

Step 87 - Case 110 Starts a left byte output transfer and verifies that the state machine hangs in state 26. Failure to remain in state 26 causes diagnostic execution to be terminated.

4.15 SECTION 15 - DMA WAIT STATES

This section checks various wait states where the DMA state machine is to wait for the assertion of DMARQ. These are checked by removing DMARQ and then entering those states and verifying that the state machine is still in the same state one state time later. (Refer to Section VI for notes on synchronized DMA.)

Step 88 – Case 110 Starts a left byte input transfer and verifies that the state machine hangs in state 8. Failure to do so results in termination of the diagnostic.

Step 89 – Case 110 Starts a right byte input transfer and verifies that the state machine hangs in state 10 when DMARQ is not asserted. Failure to remain in state 10 causes the diagnostic to terminate.

4.16 SECTION 16 - ADDRESS ROLLOVER

This section generates address rollover, which should abort the DMA transfer and set DMA states to 11. (Refer to Section VI for notes on synchronized DMA).

Step 90 – Case 111 Starts a right byte output transfer with address equal to !FFFF. States 0, 28, 24, and 25 are traversed, and an address roll-over is generated. DMA EN is checked to verify that it has been cleared.

Step 91 – Case 112 Verifies that the address rollover generated in step 90 set DMA STATUS to 11 and set the ADR OVF bit in register B.

4.17 SECTION 17 - IMB MEMORY TIMEOUT

This section tests the memory timeout logic by accessing non-existent memory.(Refer to Section VI for notes on synchroniz-ed DMA.) This section is skipped unless memory for the 3000/30 and 33 is present.

Step 92 – Case 113 Starts a DMA transfer that accesses memory location !0FF 0003; an address that is hopefully non-existent. This causes a memory timeout to be generated. This occurrence is verified by reading register B and seeing that DMA STATUS is 11 and MEM TIM is set.

4.18 SECTION 18 - PARITY ERROR ABORT

This section checks the parity error circuitry in the master handshake and clearing of abort DMA STATUS. (Refer to Section VI notes on synchronized DMA.) This section is skipped if CPU is 3000/64.

Step 93 - Case 114 Checks the parity error logic in the master
          handshake by creating a word in memory with incorrect
          parity and reading it with a DMA transfer.   The PAR ERR
          bit, register B, is read to be sure it is set.

Step 94 - Case 115 Verifies that a write to register B clears
          DMA STATUS from 11 to 00.

4.19 SECTION 19 - DNV ASSERTION

          (Refer to Section VI for notes on synchronized DMA.)

Step 95 - Case 116 Verifies that DNV is asserted on the IMB when
          a read of register 0 is attempted with no parallel poll
          in progress and no byte in the inbound FIFO.

4.20 SECTION 20 - DNV ASSERTION

          (Refer to Section VI for notes on synchronized DMA.)

Step 96 - Case 117 Verifies that DNV is not asserted on the IMB
          when register 0 is read and a parallel poll is in pro-
          gress on the HP-IB.

4.21 SECTION 21 - NEW STATUS WHEN NOT CACS

          (Refer to Section VI for notes on synchronized DMA.)

Step 97 - Case 118 Determines that NEW STATUS may be set by SIOP
          when the channel is not CACS by verifying that NOT VAL
          is cleared following the operation.

4.22 SECTION 22 - DNV ASSERTION

          (Refer to Section VI for notes on synchronized DMA.)

Step 98 - Case 119 Verifies that DNV is not asserted when
          register 0 is read, no parallel poll is in progress but
          DMARQ is asserted (signifying a byte in the inbound
          FIFO).

4.23 SECTION 23 – DMA LOOPBACK TESTS

Loopback tests of DMA. (Refer to Section VI for notes on synch-ronized DMA).

Step 99 - Case 121 Verifies that the last byte of an output transfer has EOI appended to it when this function has been enabled. This is checked by doing an output transfer with the PHI in loop-back mode and then reading the byte transferred from the inbound FIFO. Bits 0 and 1 should be 11.

Step 100 - Case 122 Verifies that the right byte of the last word read before an address rollover is detected by the state machine is not cleared before it is transferred to the HP-IB. If both right and left bytes are incorrect (reported by error messages), a more serious problem is indicated than only the alteration of the right byte. Perhaps memory reads or PHI writes are not operating properly.

4.24 SECTION 24 – GIC TIMEOUT LOGIC CHECK

This section verifies correct operation of the GIC timeout logic. The section is done in four steps.

Step 101 - Normal interrupt, halt

The channel program:
    WREG 0,!405E   <<address chan to talk>>
    INT H, 1, F   << end >>
is executed. Normal channel program processing is expected. After allowing sufficient time (milliseconds) for the channel program to complete, the second CPVA word is checked to insure a value of !800F, the expected halt code.

Step 102 - Receive data request sent and no data

The channel program:

    WREG 0, !C000   <<receive data request>>
    Int H, 1, 1   <<end>>

is executed. This program should not complete normally, but should be aborted by the CPU because the inbound FIFO of PHI has been told to receive data then no data is transfered. The 4th DRT word is checked during the timeout to insure it contains the value !8004 (the program is suspended waiting for another service request), a sufficient time (1 second) is

allowed for the timeout abort, then CPVA word 0 is checked to insure a value of !E004 (timeout).

Step 103 - Receive data request - DMA INPUT

The channel program:

```
WREG 0,!C000          <<receive data request>>
EXECUTE DMA           <<byte count=9, input>>
END
```

is executed. This program should time out because more bytes are expected by the channel program (9 bytes) then can be provided by the inbound FIFO. The 4th DRT word is verified to be !8002 (channel program and DMA active) a sufficient time (1 second) is allowed for the timeout to occur, then CPVA word 0 is verified to be !E004 (timeout abort).

Step 104 - Receive data request - DMA output
The channel program:

```
WREG, 0, !C000        <<receive data request>>
EXECUTE DMA           <<output, byte count=2>>
END
```

is executed. This program should time out because a read data request is sent, then an OUTPUT transfer is executed. The same tests of the DRT and CPVA words described in the previous step are executed.

4.25 SECTION 25 - GIC-TO-GIC TRANSFERS

Step 105 - Case 120

This section uses a known good second GIC to provide stimulus to, and to respond to stimulus provided by the GIC under test. (Refer to paragraph 1.1 for cabling instructions.)

The GIC under test, using HP-IB device address 5, transfers a 256 byte buffer on the HP-IB to device address 3 on the known good GIC. Then, the direction of transfer is reversed and 256 bytes are transferred on the HP-IB from device address 3 of the known good GIC to device address 5 of the GIC under test. This sequence is repeated 100 times. A dot appears on the display every other time. The DRT and CPVA areas of both channel/device number are tested to insure completion of these channel programs. Then, the transfer buffers are checked to verify the ability of the GIC under test to transfer data in and out of memory. Interface Clear signal is sent via the second GIC to the one under test to check its response.

```
+-----------------------------------------------------+ +---------+
|                                                     | | SECTION |
|                               ERROR INTERPRETATION  | |---------|
|                                                     | |    V    |
+-----------------------------------------------------+ +---------+
```

5.0 INTRODUCTION

The table on the following page relates functions and blocks
tested (e.g., channel program execution; PHI) to specific tests
(Sections in the Diagnostic) and subsystems (GIC, IMB, CPU,
and Memory).

There are two ways to use the table:

(1) Enter the table at the failing section. This tells you:
    a. which function or block was under test
    b. the most likely failing subsystems in order of probability

(2) Enter the table at a block or function. This tells you:

    a. which test section explicity tests that block or function

    b. which test sections depend on that block or function in
       order to pass

    c. which subsystems cause failure of that block or function

If every block or function were independent of the others and
could be tested independently, then the table would have only a
diagonal line of 'X's. Since this is not the case, the table
provides a compact guide to these interrelationships; to deal
with the cases in which the subsystem appears to fail due to the
failure of other subsystems.

| Block or Function Tested | Section Number | | | | | | | | | | Subsystems In Order Of Failure Probability |
	1	2 v 7	3	8 v 16	17	18	19 v 22	23	24	25	
Switches & Interrupts	X	-	-	-	-	-	-	-	-	-	GIC, IMB, CPU
PHI	-	X	-	-			-	X	-	-	PHI, GIC
CSRQ and Registers		-	X	-	-	-	-	-	-	-	GIC, PHI, IMB
DMA State Machine				X	-	-	-	-	-	-	GIC, PHI, IMB, Memory
Memory Timeout					X						GIC, IMB, Memory, CPU
Mem Parity Error						X					GIC, IMB, Memory
DNV on IMB							X				GIC, IMB, CPU
Channel Programs									X	X	GIC, CPU, Memory
GIC-GIC Transfer										X	PHI, GIC, HP-IB

Key: X = function or block under test
      - = function or block must work to pass test

Figure 5.1   Function/Block Reference Table

```
+------------------------------------------------+ +---------+
|                                                | | SECTION |
|                             GLOSSARY OF TERMS  | |---------|
|                                                | |   VI    |
+------------------------------------------------+ +---------+
```

Case - The entity which verifies proper circuit conditions along a distinct path in the logic, which has a unique case number and has associated with it one or more location designators. Cases verify portions of the board which correspond to the location designators associated with all signals which are active during the test.

Channel - In this diagnostic, this refers to the 31262A General I/O Channel.

Error Message - A message output when an error condition is detected in response to a diagnostic stimulation. Error Messages generally contain symptom data specific to the error encountered.

PHI - Processor to HP-IB Interface chip.

Synchronized DMA Transfer - A DMA transfer which is run in synchronization with the slave handshake, where the DMA state variables are updated once every three slave handshake operations of the channel being diagnosed. To enable the DMA state clock to become properly synchronized, so that the state variable change occurs after the third slave handshake, the state variables are sampled until the change is detected by a diagnostic program sub-routine before any section requiring synchronized transfers is executed.

HP 3000 Computer System

Error-Correcting Memory
Diagnostic Manual

HEWLETT PACKARD

LIST OF EFFECTIVE PAGES

The List of Effective Pages gives the date of the current edition and of any pages changed in updates to that edition. Within the manual, any page changed since the last edition is indicated by printing the date the changes were made on the bottom of the page. Changes are marked with a vertical bar in the margin. If an update is incorporated when an edition is reprinted, these bars are removed but the dates remain.

PRINTING HISTORY

New editions are complete revisions of the manual. Update packages, which are issued between editions, contain additional and replacement pages to be merged into the manual by the customer. The date of the title page of the manual changes only when a new edition is published. When an edition is reprinted, all the prior updates to the edition are incorporated.

Print Date . . . . Nov 1978
Update No. 1 . . . . Jun 1980
Update No. 2 . . . . Jan 1981

CONTENTS

TABLE OF CONTENTS

CONTENTS (continued)

1.0 INTRODUCTION

The HP Error Correcting Memory Diagnostic, hereafter referred to
as "the diagnostic", is designed to test memory sub-systems based
on the HP Memory Controller, for malfunctions associated with the
semiconductor memories.

The HP memory diagnostic is a departure from traditional semicon-
ductor memory diagnostics. The departure evolved as a direct re-
sult of the technological breakthrough in RAM size and a hardware
error logging mechanism. The traditional approach involved the
generation of so-called "worst case" patterns to exercise memory
which usually always included "N-squared type" tests. Typically
these attempts fell short when compared with the exercising of
memory with an operating system that is implementing "real world
tasks". Relative execution times for "N-squared testing" of 16K
of memory with 4K RAMs is two hours and eight hours for 16K RAMS.
One can readily see that testing times become unreasonable even
with small memories.

With this in mind, the HP memory diagnostic's primary function is
to identify all "hard" memory subsystem failures using "N-type"
tests and if a RAM is involved, pinpoint its physical location on
an array board. "N-type" test execution time is in the order of
minutes as opposed to "N-squared" test execution time, which is
in the order of hours. The task of exercising the memory subsys-
tem has been left to the operating system. "Soft" and "hard"
single bit errors detected during system operation will be auto-
matically logged in the error log. The contents of the error log
may be displayed and the RAMs that have failed located and
replaced.

1.1 REQUIRED HARDWARE

a. An HP 3000 HP-IB version CPU with a minimum of 128K bytes of
   memory. A memory with no errors is desirable, but correctable
   (single bit) errors are acceptable (see limitations c,i and j)

b. Channel and loading device for retrieving the diagnostic from
   a diagnostic storage medium (flexible disc or tape).

c. Communication link between the user and the diagnostic. (An
   IDS or other terminal supported by the Diagnostic/Utility
   System.)

1.2 REQUIRED SOFTWARE

a. Diagnostic/Utility Disc or Tape containing the HP Stand-Alone
   Error Correcting Semiconductor Memory Diagnostic.

1.3 MESSAGES

Two types of messages are output by the diagnostic: error and
information. Error messages are used to inform the operator of a
failure of the memory subsystem to respond properly to a test.
IN the case of Controller tests (Test Sections 2,3,4, and 5),
Surround Disturb (Test Sections 11 & 16), and Address Paths (Test
Section 19) the diagnostic will output an error message and
attempt to continue executing if error pause is suppressed.
Information messages are used to inform the operator of the
progress of the diagnostic or to instruct the operator to perform
an operation. Information messages, with the exception of the
diagnostic heading, interrupt or pause prompt post diagnostic
dialogue and the diagnostic "heartbeat" are suppressable at
diagnostic configuration time. Error messages are not
suppressable.

Error messages format:

error in step nn:
xxxxxxxxxxxxxx error message text xxxxxxxxxxxxxx

(nn => step number in which error was detected)

1.4 LIMITATIONS

The limitations listed here apply to the whole diagnostic in
general. In the case of Relocation, tests (Test Sections 15, 16,
17, and 18) require relocation of the diagnostic. NO messages
will be displayed during these test sections. A message,
'Diagnostic is relocating to execute selected relocation tests',
will be displayed before execution of Relocation Tests. The
diagnostic will execute the selected Relocation tests. Upon
completion of the selected test sections and re-relocation of the
diagnostic, another message will be displayed:

    'Selected relocation tests completed - Diagnostic has
    relocated back.'

Any errors detected during relocation will cause the diagnostic
to halt.

a. If the lowest 64K bytes of memory is included in the memory
   test limits, it will be tested separately from the rest of
   memory by relocating the diagnostic into upper memory and then
   testing the lower 64K bytes.

b. Memory configuration errors cannot be directly detected. (e.g. two array boards or two controllers with the same switch settings) These types of problems may be discerned by the operator when the Memory Configuration Map is displayed.

c. Multi-bit errors in the lowest 128K bytes of memory may cause errors in diagnostic execution if relocation test(s) are requested.

d. Program restart after diagnostic failure may require reloading of the diagnostic.

e. Will not detect user configuration errors. (e.g. user tells diagnostic to run over non-existent memory that is less than the maximum memory found. The diagnostic will test to the specified limits)

f. No attempt is made to test any switches.

g. There is no facility to differentiate between hard and soft errors.

h. Tests are done on a logical level and do not attempt to test RAMs with actual physical patterns. Thus a checkerboard pattern written logically to a RAM (with respect to RAM addresses) may not actually be a checkerboard pattern in the RAM due to address scrambling and data inversion within the RAM.

i. Address(es) !3FFF (and !23FFF) must be 22-bit error free to perform controller tests for controller(s) 0 and 1 respectively.

j. If the lowest 64K bytes of memory has correctable errors, the error log tests (Test Sections 2 & 3) most likely will not pass. This limitation arises as a result of an error being logged in the error log not under diagnostic control.

k. Improper logging of errors in the error log RAM will occur if multi-bit hardware-undetectable errors are encountered.

MEMORY DIAGNOSTIC

OPERATING INSTRUCTIONS

2.0 INTRODUCTION

All responses made to the diagnostic or the Diagnostic/Utility System prompts should be followed by pressing the ENTER or RETURN key depending on the type of communication link being used.

Basic diagnostic procedure overview:

```
begin
  |
  |
  v
3.1 Coldload Diagnostic/Utility System
  |
  |
  v
3.2 Invoke diagnostic
  |
  |
  v
3.3 Configure Diagnostic
  |
  |
  v
3.4 Diagnostic executes per selected and/or default options
  |
  |      Diagnostic may be reconfigured during program pauses.
  |      Reconfiguration will not affect diagnostic execution
  |      until the diagnostic makes its normal check for con-
  |      figuration options. (e.g. deletion of a test
  |      section that is currently being executed will not
  |      cause that test section to be exited, but rather,
  |      the test section will complete and not be executed
  |      then next time).
  v
end of diagnostic and return to Diagnostic/Utility System
```

2.1 COLDLOAD DIAGNOSTIC/UTILITY SYSTEM

a. Install a Diagnostic/Utility Disc or Tape.

b. Set Control Panel COLD LOAD CHANNEL and DEVICE switches to the channel and device number of the cold load device.

Memory Diagnostic

c. Press Control Panel HALT.

d. Press Control Panel SYSTEM RESET.

e. Press Control Panel LOAD. (Control Panel RUN light will light for a while and extinguish. The Diagnostic/Utility System will respond with the following message within 50 seconds.

```
Diagnostic/Utility System  Revision nn.nn
Enter Your Program Name (type HELP for program information)
:
```

2.2 INVOKE DIAGNOSTIC

a. Respond to the Diagnostic/Utility System prompt with:

MEMDIAG

2.3 CONFIGURE DIAGNOSTIC

Commands available allow the user to configure or reconfigure the diagnostic according to his needs if they differ from the existing or default mode. These commands may be entered after diagnostic interruption has been achieved.

Command	Parameter(s)	Description
EEPS		Enable Error PauSe
ENPR		Enable Non-error PRint
EXIT		EXIT and return to DUS
GO		GO, resume diagnostic execution
HITM	32K byte mod	Diagnostic high test limit
IEL		Initialize Error Log(s)
LC		List Commands
LEL	controller #	List Error Log
LOOP		Loop on current test list
LOOPOFF		Do not loop on current test list
LOTM	32K byte mod	Diagnostic low test limit
RUN		Restart diagnostic with current configuration
SEPS		Suppress Error PauSe
SNPR		Suppress Non-error PRint
TEST	test list	Specify TEST(s) to be executed

LISTING COMMANDS (LC)

lists commands avaliable and gives abbreviated description and parameters where applicable

CONTINUING DIAGNOSTIC EXECUTION (GO)

allows the user to continue/resume diagnostic execution

EXITING DIAGNOSTIC (EXIT)

allows the user to exit diagnostic and return to the Diagnostic/Utility System

RESTARTING THE DIAGNOSTIC (RUN)

allows the user to, at the point of diagnostic interruption, re-start the diagnostic without having to re-invoke the diagnostic. When the diagnostic is restarted in this manner a message (re-started by RUN) is output instead of the System Memory Map. ALL flags and limits are left not modified.

LISTING CONTENTS OF AN ERROR LOG (LEL n)

(where n is the controller number of interest) allows the user to list the contents of a specific controller error log. If the second controller (controller number 1) is not present, a message to that effect is output.

INITIALIZING THE ERROR LOG(S) ON A SYSTEM (IEL)

Allows the user to initialize all error logs known to the diag-nostic.

SPECIFYING OR CHANGING MEMORY TEST LIMITS (LOTM n / HITM n)

(where n is the 32K byte module to be used as the lower or upper limit) for either limit n must not be less than 0 nor greater than the maximum memory found by the diagnostic in Memory Confi-guration Determination. A check is made at execution time to determine that the hi test limit is greater than or equal to the lo test limit.

HALTING OR CONTINUING AFTER ERROR OCCURS (EEPS and SEPS)

Allows the user to specify action to be taken by the diagnostic after it has detected an error (halt (EEPS) or continue (SEPS)).

LOOPING OR NOT LOOPING ON CURRENT LIST OF TESTS TO BE EXECUTED (LOOP and LOOPOFF)

Allows the user to loop (LOOP) or not loop (LOOPOFF) on current list of tests to be executed.

MEMORY DIAGNOSTIC

SUPPRESSING OR ALLOWING NON-ERROR MESSAGES TO BE OUTPUT (SNPR and ENPR)

Allows the user to specify the output (ENPR) or suppression (SNPR) of informational messages.

SPECIFYING OR CHANGING SECTIONS TO BE EXECUTED (TEST test list)

(a test list is a list of sections) allows the user to replace or modify the list of sections to be executed according to directions specified in the test list

a plus (+) in front of the test list will cause the sections specified in the test list to be added to the current list of sections to be executed

a minus (-) in front of the test list will cause the sections specified in the test list to be deleted from the current list of sections to be executed. If the resulting list of sections to be executed is null the list of sections to be executed will revert to the default set.

a test list without a plus or minus will cause the current list of sections to be executed to be replaced by the test list.

EXAMPLES

for each of the following examples assume current list of sections to be executed is: 1,2,3,4,5,10,15,16

(1) TEST +11,12,13 or TEST +11/13 will change the list of sections to be executed to:

     1,2,3,4,5,10,11,12,13,15,16

(2) TEST -1,2,3,4 or TEST -1/4 will change the list of sections to be executed to: 5,10,15,16

(3) TEST +5 or TEST -2 will not alter the list of sections to be executed

(4) TEST -2,1,3/5,10,16 will change the list of sections to be executed to: 15

(5) TEST -1/5,10,15/16 will cause the list of sections to be replaced by the default set

(6) TEST 1/4,9/12,15,18/20,23 will change the list of sections to be executed to: 1,2,3,4,9,10,11,12,15,18, 19,20,23

(7) TEST 19 will change the list of sections to be executed to:
     19

2.4 OUTPUT DIAGNOSTIC HEADING AND SYSTEM MEMORY CONFIGURATION MAP

The diagnostic will output a heading, perform Self-read (step 1),
Diagnostic Compatibility (step 3), output the Memory Configura-
tion Map. Before the prompt, the diagnostic may output a message
for each controller on the system that has been found to have
logged error(s).   If the diagnostic fails to reach the point of
outputting the prompt, most likely hardware requirements have not
been satisfied or a procedural error has occurred.

The Memory Configuration Determination step determines what 32K
byte blocks of memory are present and displays them in a config-
uration map.

Example of memory configuration map:

```
----------------------------------------------------------------
 HP 31202A Error Correcting Memory Diagnostic  00.00

 System memory configuration

   controller 0

     0    1    2    3    4    5    6    7     128K byte block
     1111 1111 1111 ?111 11.1 1111 .... ....

   controller 1

     0    1    2    3    4    5    6    7     128K byte block
     1111 11?. .... .... .... .... .... ....

                   . => absent
                   1 => present
                   ? => present with multi-bit error(s)

 type GO to continue (LC to list commands)
 >

----------------------------------------------------------------
```

Figure 3.1 - Example of Memory Configuration Map

This example shows a system containing 2 controllers (0 & 1) with
768K bytes of memory on controller 0 of which there is a problem
with the 4th and 5th array boards (arrays 3 and 4) and 224K bytes
of memory on controller 1 of which there is a problem with the
2nd array board.

NOTE: Systems with one controller will only output memory found
for controller 0.

2.5 DIAGNOSTIC EXECUTES PER SELECTED AND/OR DEFAULT OPTIONS

```
begin
 |
memory configuration map; then diagnostic pause
 |
 |<---------------------------------------------------------
 v                                                         ^
execute selected controller test sections                 |
 |                                                         |
 v                                                         |
execute selected >=64K byte RAM test sections [*]          |
 |                                                         |
 v                                                         |
any relocation test sections selected?---------->          |
 |                                               |         |
 | yes                                           | no      |
 v                                               |         |
relocate diagnostic out of lowest 64K bytes      |         |
 |                                               |         |
 v                                               |         |
execute selected <64K byte RAM test sections     |         |
 |                                               |         |
 v                                               |         |
re-relocate diagnostic back to lowest 64K bytes  |         |
 |                                               v         |
 |<--------------------------------------------------      |
 v                                                         |
execute selected special test sections                    |
 |                                                         |
 v                                                         |
looping?--------------------------------------------------->
 |            yes
 | no
 |
 |            ------------------------------------------
 |           |                                          |
 |           |  [*] => operator intervention requested  |
 |           |         (ATTENTION or control-Y) returns |
 |           |         to point of interruption if exit |
 |           |         not requested.                   |
 |           ------------------------------------------
 |                              |
 |                              | exit request made
 |                              v
 |<---------------------------
 |
 v
end of diagnostic execution per selected
and/or default options
```

        Figure 3.2 - Selected and/or Default Options

2.6 POST-DIAGNOSTIC EXECUTION

```
begin
  |
  v
post-diagnostic information
  |
  v
return to DUS
```

To allow saving of information that will be displaced by the error
log map, the diagnostic will pause before outputting the er-
ror log, waiting for an operator response.

```
---------------------------------------------------------------
     0    1    2    3    4    5    6    7    128K byte block
0   ....  ....  ....  ....  .X.X XXXX ....  XXXX
1   ....  ....  ....  ....  .X.X XXXX ....  XXXX
2   ....  XX..  ....  ....  .X.X XXXX ....  ....
3   ....  ....  ....  ....  .X.X XXXX ....  ....
4   ....  ....  ....  ....  .X.X XXXX ..X. ....
5   ....  ....  ....  ....  .X.X XXXX ....  ....
6   ....  ....  ....  ....  .X.X XXXX ....  ....
7   ....  ....  ....  ....  .X.X XXXX ....  ....
8   ....  ....  ....  ....  .X.X XXXX ....  ....
9   ....  ....  XXXX  ....  .X.X XXXX ....  ....
10  ....  ....  ....  ....  .X.X XXXX ....  .X..
11  ....  ....  XXXX  ....  .X.X XXXX ....  ....
12  ....  ....  ....  ....  .X.X XXXX ....  ....
13  ....  ....  ....  .X..  .X.X XXXX ....  ....
14  ....  ....  ....  ....  .X.X XXXX ....  ....
15  ....  ....  ....  ....  .X.X XXXX ....  ....
H0  ....  ....  ....  ....  .X.X XXXX X... ....
H1  ....  ....  ....  ....  .X.X XXXX ....  ....
H2  ....  ....  ....  ....  .X.X XXXX ....  ....
H3  ....  ....  ....  ....  .X.X XXXX ....  ....
H4  ....  ....  ....  ....  .X.X XXXX ....  ....
H5  ....  ....  ....  ....  .X.X XXXX ....  ....

---------------------------------------------------------------
```

Figure 3.3 - Example of Error Log Map

This example illustrates some of the more common memory subsystem failure modes. Interpretation of the failures is explained in Section 6 (Error Interpretation) of this ERS.

2.7 LOOPING ON A SINGLE TEST

We wish to loop on section 15, over 32K byte blocks 2 through 5 inclusive and not destroy the current contents of the error log.

Enter the following commands:

      TEST 15

      LOTM 2

      HITM 5

      LOOP

      ENPR

The diagnostic will loop until the ATTENTION key or control-y is depressed switch is changed. Output request for controller 0 will display results of step 15 looping and historical data that was in the error log RAM before the test was started. Had any controller test(s) been selected, historical error log RAM data would have been lost.

EXECUTION TIMES

3.0 INTRODUCTION

The following table contains information about standard and
default test section execution time and a brief description of
each test section.

```
    ----------Section
   |  --------Step
   |  |  -----Standard (Default) Set
   |  |  |  --Test Description          Execution Time
   |  |  |  |
-----------------------------------------------------------
 1  1  x  Self-read                     2 sec / system
    2  x  Memory Configuration Determin.
    3  x  Diagnostic Compatibility
    4  x  Initialization                operator
    5     not used
 2  6  x  Error Log 1s/0s               ¯1 sec / controller
 3  7  x  Error Log March 1s/0s         ¯1 sec / controller
 4  3  x  Error Logging/Correction      ¯1 sec / controller
 5  9  x  Error Detection               ¯1 sec / controller
 6 10  x  Read/Write 1s                 <1 sec / controller
 7 11     1s/0s                         2 sec / 32K bytes
 8 12  x  Data Patterns                 4 sec / 32K bytes
 9 13  x  Move Program                  <1 sec / 32K bytes
10 14  x  March                         8 sec / 32K bytes
11 15     Surround Disturb              223 sec / 32K bytes
12 16     Galpat - row/col              399 sec / 32K bytes
13 17     Shifting Diagonal             480 sec / 32K bytes
14 18  x  Pre-relocation Self-read      3 sec
15 19  x  March lower 32K bytes         10 sec / 32K bytes
16 20     Surround Disturb lower 32Kb   210 sec / 32K bytes
17 21     Galpat row/col lower 32Kb     380 sec / 32K bytes
18 22     Shifting Diagonal lower 32Kb  290 sec / 32K bytes
19 23  x  Address Paths                 <1 sec / controller
20 24     Test Lights                   operator
21 25     Refresh Tests                 operator
22 26     Multi-bit Error Test          <1 sec
23 27     Display Error Log             ¯2 sec / controller
```

Default is 28 seconds + 16 seconds / 32 KB

Figure 4.1 - Step Execution Times

4.0 INTRODUCTION

Possible error message for any portion of this diagnostic:

   "multi-bit error - cannot continue"

   data format used in the following error messages:

```
dddd dddd dddd dddd
^                 ^
|                 |
0                 15      <-- bit

dddd dddd dddd dddd dddddd
^                 ^ ^    ^
|                 | |    |
0                 15 H0  H5        <-- bit
```

4.1 TEST SECTION 1 — DIAGNOSTIC PREPARATION/VERIFICATION

The following tests are executed before the main portion of the
diagnostic to verify the diagnostic environment and insure that
the diagnostic area is minimally functional in that it contains
non-fatal errors.   This section is not suppressable and is not
loopable.

Step 1 - Self-read

            This step   checks   for   multi-bit   hardware   detectable
            errors   by   reading   the lowest 64K bytes including the
            diagnostic area (program, data, associated tables   and
            pointers).   Replacement   or   repair of the lowest 64K
            bytes in the first array board and   reloading   of   the
            diagnostic will be required to rerun the diagnostic in
            the event of an error in this step.

Possible error message:

      "multi-bit error in program area - cannot continue"

Step 2 - Memory configuration determination

            This determines what 32K byte memory blocks are present
            by writing/reading to/from the first word of each   pos-
            sible 32K byte block. A memory map is displayed show-
            ing system memory configuration. If PRONTO memory is
            installed, the following message will be displayed:

"controller is 30094-6000, use PRMDIAG"

The program then returns to DUS.

Step 3 - Diagnostic compatibility

This detects incompatibilities between the diagnostic and the memory subsystem by doing a status read on each controller and comparing it to the expected status for a 22-bit error correcting memory subsystem.

Possible error message:

"controller n not compatible with diagnostic controller type found is !dd"

(dd = five least significant bits of status word should be !10 n = {0/1})

Step 4 - Initialization (Diagnostic configuration)

This allows the user the option of configuring the diagnostic.

The following tests are performed on the controller(s) specified by the information obtained from Memory Configuration Determination (step 2) or the low and high memory test limits from Initialization (Diagnostic Configuration) (step 5). All controller tests, by nature of their testing technique, initialize the error log RAM.

4.2 TEST SECTION 2 — ERROR LOG RAM TESTING

Verifies that the error log RAM is minimally functional by writing/reading 0s and 1s to/from all cells

Step 6 - Error log 1'S/0'S

The error log is written to all ones and verified; then to all zeroes and verified.

Possible error message:

"controller n wrote k to error log read m"

(n = {0/1}   k = {0/1}   m = {1/0})

4.3   TEST SECTION 3 — EXTENDED ERROR LOG RAM CHECKING

Verifies the proper operation of the error log RAM by exercising it with a more rigorous test pattern (March)

Step 7 -   Error Log March 1s/0s

> This step marches 1s through 0s and 0s through 1s in the error log.   The error log is written to a background of 0s. In ascending order each cell is read for a 0 and then written to a 1 and verified until the whole error log contains 1s. The process is repeated for 0s with the current background of 1s in descending order until the whole error log contains 0s.

> Possible error messages:

"controller n wrote k to error log read m"

$$(n = \{0/1\} \qquad k = \{0/1\} \qquad m = \{1/0\})$$

"controller n - error log RAM fails March m"

$$(n = \{0/1\} \qquad m = \{0/1\})$$

The next 3 Test Sections require a 22-bit error free word at address !3FFF. This word is tested in each section.

4.4   TEST SECTION 4 — TEST CONTROLLER ERROR HANDLING

Verifies proper operation of error handling function of the controller(s).

Step 8 -   Error logging/correction

> This step verifies proper logging and correction of single bit errors by:

> (1)   Verifying 22 bit test word to be error free (Warning: this requires disabling of error correction and thus no errors can be tolerated in the executing program or stack areas).

> (2)   Creating all possible single bit errors (22)

> (3)   Verifying proper correction

> (4)   Checking error log for proper logging

Possible error messages:

"controller n wrote k to error log read m"

(n = {0/1}   k = {0/1}   m = {1/0})

"controller n did not set E bit in status word test bit bb"

(n = 0 or 1   bb = 0->15 or H0->H5)

"controller did not correct test bit properly in data word
    read EC on: dddd dddd dddd dddd
    expected: dddd dddd dddd dddd"

"error at bit nn of test word not logged at proper location"

(nn = bit not properly logged (0-15,H0-H5))

"controller changed data word when hamming/parity test bit was
 bad   read EC on: dddd dddd dddd dddd
            expected: dddd dddd dddd dddd"

"!aaaaa
        read EC off: dddd dddd dddd dddd dddddd
        expected: dddd dddd dddd dddd dddddd"

(aaaaa => {!3FFF/!23FFF})

4.5 TEST SECTION 5 — MULTIPLE-BIT ERROR DETECTION

Step 9 - Error detection

This step verifies proper operation of multiple-bit
hardware error detection function by:

(1) Verifying 22 bit test word to be error free
     (Warning: this requires disabling of error
     correction and thus no errors can be tolerated in
     the executing program or stack areas.

(2) Creating multi-bit hardware detectable errors (by
     writing all error syndromes not used for single
     bit errors to hamming and parity bits)

(3) Verifying proper detection

possible error messages:

"controller n wrote k to error log read m"

(n = 0/1}   k = {0/1}   m = {1/0})

"controller n multi-bit error not detected - syndrome = !dd"

     (n = {0/1}   dd = syndrome in error)

"controller n did not set P bit in status word - syndrome = !dd"

     (n = 0 or 1, nn = syndrome in error)

"!aaaaa read EC off: dddd dddd dddd dddd dddddd expected: dddd dddd dddd dddd dddddd"

     (aaaaa => {!3FFF/!23FFF})

4.6   TEST SECTION 7 — WRITE 1's AND 0's

Step 11 -   1s/0s

ls/0s are written sequentially in ascending order   into
the memory block under test. The data written is read
and verified in the same fashion.

4.7   TEST SECTION 8 — RAM BIT ERROR DETECTION

Step 12 - Data Patterns

This test writes all possible data patterns to one word
in each 32K byte block. This will cause completely bad
RAMs to be logged even if there are two-bit failures
within a 32K byte block.

4.8   TEST SECTION 9 — MEMORY LOAD/STORE USING MOVE

Step 13 - Move program

This uses the lowest 32K bytes as data and copies it to
and verifies it in all upper 32K byte blocks using the
MOVE instruction. The MOVE instruction exercises memory
significantly faster than the standard load and store.

4.9   TEST SECTION 10 — MARCHING READ/WRITE 1's/0's

Step 14 - March

The memory block under test is written to   1s/0s.
Starting at the first address the word is read and ver-
ified for 1s/0s and then written to 0s/1s. This   se-
quence is continued to the last address of the test
block.   Starting at the last address the word is read
and verified for 0s/1s and then written to 1s/0s.   This
sequence is continued in descending order until   the
first address of the test block.

4.10 TEST SECTION 11 — MEMORY BLOCK TESTING

Step 15 - Surround Disturb

This step will skip non-existent memory.

Starting at the first address of the block under test the word is written to 0s/1s. The word's 8 nearest neighbors (5 for edge and 3 for corner words) are written to 1s/0s 10 times. The test word is verified for 0s/1s. The sequence is continued for all other words in the block under test. If the word under test is changed an error message will be printed out with the address and the bad data word.

Possible error message:

```
"address !dddddd
      read EC on:   dddd dddd dddd dddd
      expected:     dddd dddd dddd dddd"
```

TEST SEQUENCE ILLUSTRATION:

```
x    x    x    x    x      * x    x    x      ************
                          *                  *
                          *                  *  0    1    x
                          *                  *
x    1    1.   1    x      * 1    1    x      *
                          *                  *  1    1    x
                          *                  *
x    1    0    1    x      * 0    1    x      *
                          *                  *  x    x    x
                          *
x    1    1    1    x      * 1    1    x         3 nearest
                          *                      neighbors
                          *
x    x    x    x    x      * x    x    x
```

    8 nearest neighbors   5 nearest
                               neighbors

    x => don't care   1 => 1s (surrounding pattern)
    0 => 0s (test word)   * => edge

Figure 4.2 - Surround Disturb Test Sequence Illustration

4.11 TEST SECTION 12 — PERFORM READ/WRITE TRANSITIONS

Step 16 - GALPAT - row/col test

The memory block under test is written to 1s/0s. Starting at the first address of the block under test the word is verify for 1s/0s and then written to the complement (0s/1s). A sequence of read test word, read word´1, read test word, read word´2, ...etc is performed, with word´ signifying other words in the same row or column as the test word. All possible read/write transitions for words in the same column or row as the test word, and the test word are performed.

4.12 TEST SECTION 13 — DIAGONAL SHIFTING

Step 17 - Shifting diagonal

The test starts by writing and verifying a diagonal stripe of 1s/0s in a field of 0s/1s in the memory block under test (iteration 1). The block under test is then rewritten and verified with the diagonal shifted to the left one location (iteration 2). For the 16 location memory block in the example, the process requires 4 iterations to completely shift the diagonal through the block under test. For a 32K byte memory block the test will require 4K iterations.

```
1  0  0  0          0  0  0  1
0  1  0  0          1  0  0  0
0  0  1  0          0  1  0  0
0  0  0  1          0  0  1  0

  iteration 1          iteration 2


0  0  1  0          0  1  0  0
0  0  0  1          0  0  1  0
1  0  0  0          0  0  0  1
0  1  0  0          1  0  0  0

  iteration 3          iteration 4
```

Figure 4.3 - Test Sequence Illustration Shifting Diagonal

4.13 TEST SECTION 14 — VERIFY LOWEST 64K BYTES OF MEMORY

Verifies destination program area, relocates the diagnostic to allow the running of steps 19 through 22 lowest 64K bytes of memory.

Step 18 – Reads destination program area and relocates the diagnostic if possible.

4.14 TEST SECTION 15 — MARCHING IN LOWER 64K BYTES

Step 19 – March Lower 64K Bytes (See Test Section 10)

4.15 TEST SECTION 16 — MEMORY BLOCK TESTING (LOWER 64K)

Step 20 – Surround Disturb Lower 64K Bytes (See Test Section 11)

4.16 TEST SECTION 17 — READ/WRITE TRANSITIONS (LOWER 64K)

Step 21 – GALPAT row/col Lower 64K Bytes (See Test Section 12)

4.17 TEST SECTION 18 — DIAGONAL SHIFTING (LOWER 64K)

Step 22 – Shifting Diagonal Lower 64K Bytes (See Test Section 13)

4.18 TEST SECTION 19 — VERIFIES ADDRESS UNIQUENESS

Step 23 – Address paths

>   This step verifies unique addressability to 32K byte blocks by walking 0s through 1s of the last 2 bytes of each 32K byte block.

>   Possible error message:

"writing to 32K byte block dd affects 32K byte block ee"

(dd and ee = 32K byte block numbers)

4.19 TEST SECTION 20 — TEST LIGHT OBSERVATION

Step 24 – Test light tests

>   This creates a single bit error, a multi-bit hardware detectable error and goes into active mode tc allow test light observation. Testing of the Reset switch is possible in this step by depressing the switch and observing the extinguishing of test lights D and E.

Light(s) D and/or E may re-light if a real one or
multi-bit error occurs. To exit this test section,
depression of ATTN or CTRL-Y is required.

Test Lights

```
+-+
|o|    A=> activity (read/write/refresh)
|o|    B=> reserved
|o|    C=> reserved
|o|    D=> single bit error
|o|    E=> multi-bit hardware detectable error
+-+
```

4.20 TEST SECTION 21 — REFRESH

Step 25 - Refresh test
This test writes binary test patterns of alternating 1s
and 0s and alternating 0s and 1s to memory and halts.
Operator intervention is required to resume the test to
verify no loss of data.

4.21 TEST SECTION 22 — MULTI-BIT ERROR TEST

Step 26 - Multi-bit Error Test

This step requires that the diagnostic area of memory
be error free (lowest 64K bytes) because error correc-
tion and error detection are both disabled. This test
is like step 7 (1s/0s) but it also verifies the 22-bit
word.

Possible error message:

"!aaaaa
read EC off: dddd dddd dddd dddd dddddd
expected: dddd dddd dddd dddd dddddd"

   (aaaaa => the failing address)

4.22 TEST SECTION 23 — OUTPUT ERROR LOG

Step 27 - Output error log

This routine outputs the contents of the error log in
the form of an error log map and is invoked also by
using the LEL n command.

Possible error message:

--

"only one controller on system"

ERROR INTERPRETATION

5.0 INTRODUCTION

The following paragraphs provide an summary explanation for error
action to be taken in the instance that an error occurs in the
diagnostic execution.

5.1 DIAGNOSTIC DOES NOT START

If the diagnostic does not "come up" (doesn't output heading and
configuration map):

(1) try CPU Self-Test to check that at least memory minimally
    functional (checks first 128K bytes)

(2) self-test O.K. => most likely a non-memory hardware or diag-
    nostic medium problem

(3) If CPU self-test fails insert a known good memory array 0
    and/or memory controller board and retry CPU self-test.

(4) and/or change CTRL ADDR paddle switch (if second controller
    in system) to achieve "good" memory in lowest 32K bytes

(5) If still bad CPU self-test, pull unnecessary memory boards
    and retry diagnostic.

If no fatal errors, display appropriate error log map(s) to find
RAM failures

5.2 COMMON SUBSYSTEM FAILURE MODES

This example illustrates some of the more common memory subsystem
failure modes.

```
-----------------------------------------------------------------
       0     1     2     3     4     5     6     7    128K byte block
  0   ....  ....  ....  ....  .X.X  XXXX  ....  XXXX
  1   ....  ....  ....  ....  .X.X  XXXX  ....  XXXX
  2   ....  XX..  ....  ....  .X.X  XXXX  ....  ....
  3   ....  ....  ....  ....  .X.X  XXXX  ....  ....
  4   ....  ....  ....  ....  .X.X  XXXX  ..X.  ....
  5   ....  ....  ....  ....  .X.X  XXXX  ....  ....
  6   ....  ....  ....  ....  .X.X  XXXX  ....  ....
  7   ....  ....  ....  ....  .X.X  XXXX  ....  ....
  8   ....  ....  ....  ....  .X.X  XXXX  ....  ....
  9   ....  ....  XXXX  ....  .X.X  XXXX  ....  ....
 10   ....  ....  ....  ....  .X.X  XXXX  ....  .X..
 11   ....  ....  XXXX  ....  .X.X  XXXX  ....  ....
 12   ....  ....  ....  ....  .X.X  XXXX  ....  ....
 13   ....  ....  ....  .X..  .X.X  XXXX  ....  ....
 14   ....  ....  ....  ....  .X.X  XXXX  ...;. ....
 15   ....  ....  ....  ....  .X.X  XXXX  ....  ....
 H0   ....  ....  ....  ....  .X.X  XXXX  X...  ....
 H1   ....  ....  ....  ....  .X.X  XXXX  ....  ....
 H2   ....  ....  ....  ....  .X.X  XXXX  ....  ....
 H3   ....  ....  ....  ....  .X.X  XXXX  ....  ....
 H4   ....  ....  ....  ....  .X.X  XXXX  ....  ....
 H5   ....  ....  ....  ....  .X.X  XXXX  ....  ....

-----------------------------------------------------------------
```

GLOSSARY OF TERMS

!	Signifies hexadecimal when it preceeds a number
0s	=!0000, except in references tc error log
1s	=!FFFF, except in references to error log
Block	a contiguous portion of memory
Correctable error	is a single bit error in memory array
CPU	Central Processor Unit
EC	Error Correcting (or error correction)
Fatal Error	Typically a multi-bit error detected in the error correction enabled mode that disallows continuation of the diagnostic
Hard Error/Failure	Non-intermittent RAM failure
K	When "K" follows a decimal number it signifies 1024 decimal
Multi-Bit Hardware Detectable Error	All 2-bit and selected >2-bit errors
RAM	Dynamic Random Access semiconductor Memory device
Soft Error/Failure	Intermittent RAM failure usually pattern/ sequence sensitive

Status Word

Status word returned by the memory subsystem controller (see hardware ERS for details).

MEMORY STATUS WORD

```
                      1 1 1 1 1 1
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
  +-------------------------------+
  |              |    |P|E|1 0 0 0 0|
  +-------------------------------+
               ^                 ^
  Hamming Bits          Controller Type
```

P-bit   Multi-bit hardware detectable error has occurred since last controller reset

E-bit   Correctable error has occurred since last controller reset

Controller is reset by MCS, PON signal, System RESET, depressing S1 on Controller.

HP 3000 Computer System

PRONTO MEMORY DIAGNOSTIC
MANUAL

LIST OF EFFECTIVE PAGES

The List of Effective Pages gives the date of the current edition and of any pages changed in updates to that edition. Within the manual, any page changed since the last edition is indicated by printing the date the changes were made on the bottom of the page. Changes are marked with a vertical bar in the margin. If an update is incorporated when an edition is reprinted, these bars are removed but the dates remain.

All pages original issue . . . . Mar 1982

PRINTING HISTORY

| CONTENTS |

.

```
+-------------------------------------------------------+ +---------+
|                                                       | | SECTION |
|                          GENERAL INFORMATION | |---------|
|                                                       | |    I    |
+-------------------------------------------------------+ +---------+
```

1.0 INTRODUCTION

The HP PRONTO Memory Diagnostic, hereafter called the diagnostic,
is a diagnostic designed to verify a memory subsystem which con-
tains one or two Memory Control Logging (MCL) PCA(s) and from one
to sixteen Semiconductor Memory Array (SMA) PCAs. Each SMA
represents one Mega-Bytes of memory if 64K memory chips are used
or 256 KiloBytes of memory if 16K memory chips are used.

This diagnostic will test all control functions, check the error
logging array, force single error detection and correction,
detect multiple errors, and test all memory.

At the beginning of this diagnostic the Error Logging Array (ELA)
is read and the user is informed that if the ELA has not been
initialized since power was turned on, the ELA contains no
significant information. The user has the option, at that time,
to view the ELA or not. If test section 2 (ELA verification) or
test section 3 (Error correction/detection verification) is
executed, the ELA will be cleared of any previous information.
After each execution of this diagnostic the ELA is checked for
errors. If any are detected the user is notified and they may
review them if they wish to.

This diagnostic is a tool that can be used to find weak memory IC
chips, to reduce system crashes, and to verify proper memory cor-
rection and logging.

1.1 REQUIRED HARDWARE

a. An HP 3000 Series 40, 40SX, 44, or 30 with a minimum 256K
   bytes of memory.

b. A channel and loading device for retrieving the diagnostic
   from a diagnostic storage medium (flexible disc, cartridge
   tape, or tape).

c. A communication link between the user and the diagnostic (must
   be a terminal or teletype supported by Diagnostic/Utility Sys-
   tem).

1.2 REQUIRED SOFTWARE

a. Diagnostic/Utility System (DUS) flexible disc, cartridge tape,
   or tape containing the HP Stand-Alone Pronto Memory diag-
   nostic.

1.3 MESSAGES

Two types of messages are output by the diagnostic: error and information. Error messages are used to inform the operator of a failure of the memory subsystem to respond properly to a test. Information messages are used to inform the operator of the progress of the diagnostic or to instruct the operator to perform an operation. Information messages, with the exception of the diagnostic heading, interrupt, pause prompt, post diagnostic dialogue, and diagnostic "heartbeat" are suppressable at diagnostic configuration time. Error messages are not suppressable.

Error message format:

error in step nn:
xxxxxxxxxxxxxx error message text xxxxxxxxxxxxxx

where: nn = step number in which error was detected

1.4 LIMITATIONS

a. Memory configuration errors cannot be directly detected (e.g. two array boards or two controllers with the same switch settings).

b. Program restart after diagnostic failure may require reloading of the diagnostic.

c. No attempt is made to test any SMA switches.

d. There is no facility to differentiate between hard and soft errors.

e. Tests are done on a logical level and do not attempt to test RAMs with actual physical patterns. Thus a checkerboard pattern written logically to a RAM (with respect to RAM addresses) may not actually be a checkerboard pattern in the RAM due to address scrambling and data inversion within the RAM.

f. If the lowest 128K bytes of memory has correctable errors, the memory controller tests (Test Section 3) most likely will not pass. The lowest 128K bytes of memory contains the program code. The memory controller test turns off error correction. If the program area of memory has a single bit error, and it is read with error correction disabled, the data read will have this error. Most likely, this error will cause an illegal machine instruction to be seen by the CPU. The CPU will then issue a system halt.

```
+---------------------------------------------------------+ +---------+
|                                                         | | SECTION |
|                             OPERATING INSTRUCTIONS  |   | |---------|
|                                                         | |   II    |
+---------------------------------------------------------+ +---------+
```

2.0 INTRODUCTION

All responses made to the diagnostic or the Diagnostic/Utility
System prompts should be followed by pressing the RETURN key.

Basic diagnostic procedure overview:

```
begin
  |
  |
  v
2.1 Coldload Diagnostic/Utility System
  |
  |
  v
2.2 Invoke diagnostic
  |
  |
  v
2.3 Configure Diagnostic
  |
  |
  v
2.4 Diagnostic executes per selected and/or default options
  |
  |       Diagnostic may be reconfigured during program pauses.
  |       Reconfiguration will not affect diagnostic execution
  |       until the diagnostic makes its normal check for con-
  |       figuration options (e.g., deletion of a test section
  |       that is currently being executed will not cause that
  |       test section to be exited, but rather, the test sec-
  |       tion will complete and not be executed the next time)
  |
  v
end of diagnostic and return to Diagnostic/Utility System
```

2.1 HOW TO COLDLOAD DIAGNOSTIC/UTILITY SYSTEM

a. Install a Diagnostic/Utility Flexible Disc, Cartridge Tape, or
   Tape.

b. Set Control Panel COLD LOAD CHANNEL and DEVICE switches to
   the channel and device number of the cold load device.

c. Press Control Panel HALT.

d. Press Control Panel SYSTEM RESET. This step is excluded for HP 3000 Series 44 systems.

e. Press Control Panel LOAD. (Control Panel will switch from the HALT state to the RUN state. The Diagnostic/Utility System will respond with the following message within 50 seconds.

Diagnostic/Utility System Revision nn.nn
Enter Your Program Name (type HELP for program information)
:

2.2 HO TO INVOKE THE DIAGNOSTIC

a. Respond to the Diagnostic/Utility System prompt with:

PRMDIAG

2.3 HOW TO CONFIGURE THE DIAGNOSTIC

Commands available allow the user to configure or reconfigure the diagnostic according to his needs if they differ from the existing or default mode. These commands may be entered after diagnostic interruption has been achieved.

Command	Parameter(s)	Description
EEOPP		Enable End of Program Pause
EEPS		Enable Error PauSe
ENPR		Enable Non-error PRint
EXIT		EXIT and return to DUS
GO		GO, resume diagnostic execution
HITM	64K byte mod	Diagnostic high test limit
IEL		Initialize Error Log(s)
LC		List Commands
LDS		List Diagnsotic Status
LEL	controller #	List Error Log
LOOP		Loop on current test list
LOOPOFF		Do not loop on current test list
LOTM	64K byte mod	Diagnostic low test limit
LT		List Tests available
MFG		ManuFacturinG test
RUN		Restart diagnostic with current configuration
SEOPP		Suppress End Of Program Pause
SEPS		Suppress Error Pa\\\
SNPR		Suppress Non-error PRint
TEST	test list	Specify TEST(s) to be executed

LISTING COMMANDS (LC)

lists commands avaliable and gives abbreviated description and
parameters where applicable

LIST DIAGNSOTIC STATUS (LDS)

List the following information:
  1) list test sections that have been enabled
  2) ENPR flag true or false
  3) EEPS flag true or false
  4) EEOPP flag true or false
  5) LOOP flag true or false
  6) list low and high test moduale parameters
  7) list number of diagnostic passes executed

EXAMPLE:

A "1" under a test section indicates the test is enabled. A "0"
indicates that the test section is disabled. The following exam-
ple indicates the standard default settings for a system with 2
Mbytes of memory present.

```
PRMDIAG  02.00      1/29/82
test(s) selected:   1  2  3  4  5  6  7  8  9
                    1  1  1  1  1  1  0  1  1
loop:  false
halt on error: true
suppress non-error messages:   false
lo test mod: 00
hi test mod: 31
halt at EOP: false
pass: 00000
```

LIST TESTS   (LT)

List test section available and gives an abbreviated description
of each test section.

CONTINUING DIAGNOSTIC EXECUTION (GO)

Allows the user to continue/resume diagnostic execution

EXITING DIAGNOSTIC (EXIT)

Allows the user to exit diagnostic and return to the
Diagnostic/Utility System

MANUFACTURING TEST   (MFG)

This test is used in manufacturing when failure conditions re-
quired that the controller under test be a second controller. It
sets the low memory test module to 32 and the high memory test
module to the highest test module present in the system.

RESTARTING THE DIAGNOSTIC (RUN)

Allows the user to, at the point of diagnostic interruption, re-
start the diagnostic without having to re-invoke the diagnostic.

LISTING CONTENTS OF AN ERROR LOG (LEL n)

(where n is the controller number of interest) allows the user to
list the contents of a specific controller error log.   If the
second controller (controller number 1) is not present, a message
to that effect is output.

INITIALIZING THE ERROR LOG(S) ON A SYSTEM (IEL)

Allows the user to initialize all error logs known to the diag-
nostic.

SPECIFYING OR CHANGING MEMORY TEST LIMITS (LOTM n / HITM n)

(where n is the 64K byte module to be used as the lower or upper
limit)   for either limit n must not be less than 0 nor greater
than the maximum memory found by the diagnostic in Memory Confi-
guration Determination.   A check is made at execution time to
determine that the hi test limit is greater than or equal to the
lo test limit.

HALTING OR CONTINUING AFTER ERROR OCCURS (EEPS and SEPS)

Allows the user to specify action to be taken by the diagnostic
after it has detected an error, halt (EEPS) or continue (SEPS).

LOOPING OR NOT LOOPING ON CURRENT LIST OF TESTS TO BE EXECUTED
(LOOP and LOOPOFF)

Allows the user to loop (LOOP) or not loop (LOOPOFF) on current
list of tests to be executed.

SUPPRESSING OR ALLOWING NON-ERROR MESSAGES TO BE OUTPUT (SNPR and
ENPR)

Allows the user to specify the output (ENPR) or suppression
(SNPR) of informational messages.

SUPPRESSING OR ALLOWING END OF PROGRAM PAUSE (EEOPP AND SEOPP)

Allows the user to specify the action to be taken by the diagnostic at the end of the diagnostic, pasue (EEOPP) or return to DUS (SEOPP).

SPECIFYING OR CHANGING SECTIONS TO BE EXECUTED (TEST test list)

(a test list is a list of sections) allows the user to replace or modify the list of sections to be executed according to directions specified in the test list

A plus (+) in front of the test list will cause the sections specified in the test list to be added to the current list of sections to be executed.

A minus (-) in front of the test list will cause the sections specified in the test list to be deleted from the current list of sections to be executed. If the resulting list of sections to be executed is null the list of sections to be executed will revert to the default set.
A test list without a plus or minus will cause the current list of sections to be executed to be replaced by the test list.

EXAMPLES

for each of the following examples assume current list of sections to be executed is: 1,5,6,7

(1) TEST +2,3,4 or TEST +2/4 will change the list of sections to be executed to: 1,2,3,4,5,6,7

(2) TEST -5,6,7 or TEST -5/7 will change the list of sections to be executed to: 1

(3) TEST -1,5,6 will change the list of sections to be executed to: 7

(4) TEST -1,5,6,7 will cause the list of sections to be executed to be replaced by the default set

(5) TEST 1/7 will change the list of sections to be executed to: 1,2,3,4,5,6,7

(6) TEST 4 will change the list of sections to be executed to: 4

2.4 DIAGNOSTIC HEADING AND SYSTEM MEMORY CONFIGURATION MAP

The diagnostic will output a heading, perform Section 1 step 11
(Low Memory test), perform Section 1 step 12 (Diagnostic Compat-
ibility test), and output the Memory Configuration Map. Before
the prompt, the diagnostic may output a message for each control-
ler on the system that has been found to have logged error(s).
If the diagnostic fails to reach the point of outputting a prompt
most likely hardware requirements have not been satified or a
procedural error has occurred.

The Memory Configuration Determination step determines what 64K
byte blocks of memory are present and displays them in a config
uration map.

Example of memory configuration map:

```
---------------------------------------------------------------

 System memory configuration

controller 0

    0    1    2    3    4    5    6    7     Logical # of SMA
    1111 1111 1111 ?111 11.1 1111 1111 1111

controller 1

    0    1    2    3    4    5    6    7     Logical # of SMA
    1111 11?1 .... .... .... .... .... ....

                  .  => absent
                  1  => 16K chip array present
                  6  => 64K chip array present
                  ?  => present with Detectable Double-Bit
                        error(s)

 type GO to continue (LC to list commands)
 >

---------------------------------------------------------------
```

This example shows a system containing 2 controllers (0 & 1) with
2 mega bytes of memory on controller 0 of which there is a
problem with the 4th and 5th array boards (arrays 3 and 4) and
512K bytes of memory on controller 1 of which there is a problem
with the 2nd array board.

NOTE: Systems with one controller will only output memory found
for controller 0.

2.5 DIAGNOSTIC EXECUTES PER SELECTED AND/OR DEFAULT OPTIONS

```
begin
 |
memory configuration map; then diagnostic pause
 |
 |<---------------------------------------------------------
 v                                                         /\
execute selected controller test sections                  |
 |                                                          |
 v                                                          |
execute selected >=64K byte RAM test sections [*]           |
 |                                                          |
 v                                                          |
check error log and notify user id errors detected          |
 |                                                          |
 v                                                          |
execute selected special test sections                      |
 |                                                          |
 v                                                          |
looping?-------------------------------------------------->
 |            yes
 | no
 |
 |            ------------------------------------------
 |           |                                          |
 |           |  [*] => operator intervention requested  |
 |           |        (ATTENTION or control-Y) returns   |
 |           |        to point of interruption if exit   |
 |           |        not requested.                     |
 |            ------------------------------------------
 |                                |
 |                                | exit request made
 |                                v
 |<---------------------------
 |
 v
end of diagnostic execution per selected
and/or default options
```
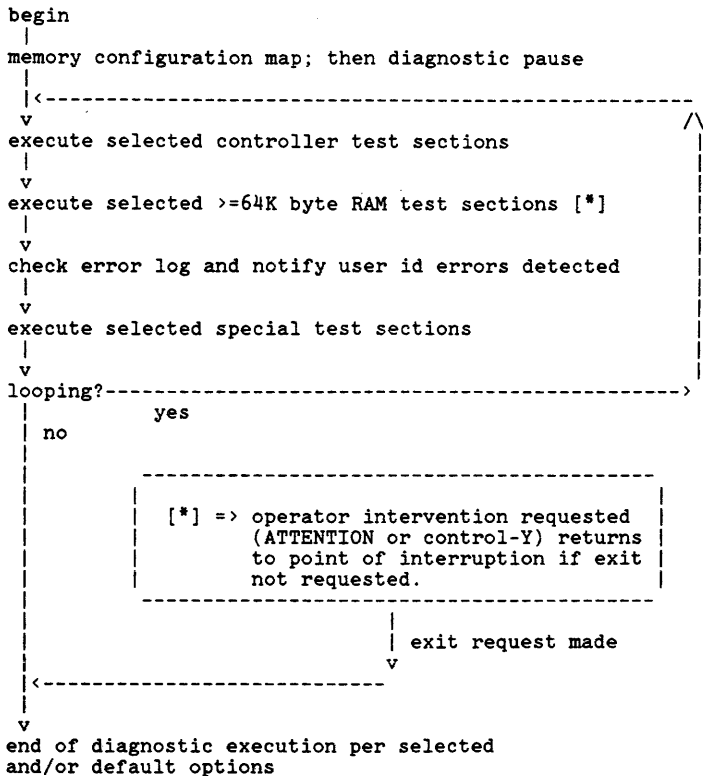
2.6 ERROR DETECTION

The Error Log Array (ELA) in the memory controller will be
checked for errors after sections 1, 6, 7 and 8. If an error is
detected after the completion of any of these sections, the diag-
nostic will notify the user that an error occurred. The error log
of the controller containing the error(s) can then be displayed
by entering the command LEL (0 or 1).

An example of an erro log display is as follows:

```
----------------------------------------------------------------
Controller X

    0    1    2    3    4    5    6    7     Logical # of SMA
    0123 0123 0123 0123                      Row # at SMA PCB
                                             SMA with 64K chips
    256K 256K 256K 256K                      SMA with 16K chips
----------------------------------------------------------------

D0     .... ..X. .... ....

D5     .... ..X. .... ....

C2     .... .... X... ....

DBL/   .... ..X. .... ....
MLT
```

Would you like to see details of double/mults error? (yes/no)
>YES

```
----------------------------------------------------------------
Syndrome 0    1    2    3    4    5    6    7  Logical # of SMA
         0123 0123 0123 0123                   Row # at SMA PCB
                                               SMA with 64K chips
         256K 256K 256K 256K                   SMA with 16K chips
----------------------------------------------------------------

Double Errors:
Syndrome
% 106    .... ..X. .... ....
----------------------------------------------------------------

Multi Errors:
Syndrome
% 050    .... .... .X.. ....
----------------------------------------------------------------
```

This example shows that SMA number 1 had two single bit errors logged (Data bits 0 and 5) and a double bit error that probably included data bits 5 and 20 in row two. Refer to appendix A for a list of other possible double bits pairs associated with syndrome %106. Any multiple bit error syndrome may be caused by a large number of possible multi-bit error combinations. Thus, no multiple error combinations are listed. The default answer to the request to display double multi-bit errors is "NO".

Refer to Appendix B for locating a defective chip on an SMA.

2.7 LOOPING ON A SINGLE TEST

We wish to loop on section 4, over 64K byte blocks 2 through 5 inclusive and not destroy the current contents of the error log.

Enter the following commands:

    TEST 4

    LOTM 2

    HITM 5
    LOOP

The diagnostic will loop until control-y is depressed. Output request for controller 0 (LEL 0) will display results of section 4 looping and historical data that was in the error log RAM before the test was started. Had any controller test(s) been selected, historical error log RAM data would have been lost.

```
+----------------------------------------------------+ +---------+
|                                                    | | SECTION |
|                              TEST DESCRIPTIONS     | |---------|
|                                                    | |   III   |
+----------------------------------------------------+ +---------+
```

3.0 INTRODUCTION

The PRMDIAG is divided into 9 test sections. Each test section
is devided into several test steps. Test section 1 sets up test-
ing. Test sections 2,3, 6 and 8 test Memory Control/Logging
(MCL). Test sections 4 and 5 check Semiconductor Memory Arrays
(SMA). Test section 7 is interactive and requires operator
intervention. In the default mode Test section 7 is skipped.
Test section 9 checks the error log for any errors before the
diagnostic returns control to DUS operating system. In Section
III, any messages printed to the terminal during diagnsotic exec-
ution are enclosed within quotation marks(").

3.1 TEST SECTION 1

Tests low memory and diagnostic compatability. Reads ELA and
optionally displays Error Log Map. Checks memory configuration
and MCL error logging capability.

Step 11 - Low memory test

                This step checks for double-bit hardware errors by
                reading the lowest 128K bytes of memory, which includes
                the area where the diagnostic is resident. Replacement
                of the first SMA may be required if this step fails.

                Possible error message:

                "Double error in program area - can't continue"

Step 12 - Diagnostic compatibility
                This step checks for installation of proper MCL(s) and
                for legal configuration of MCL(s) and SMA(s). If
                either or both is incorrect, the following error mes-
                sage is issued and the diagnostic is aborted.

                31202-60001 Memory Controller Detected. Run MEMDIAG

                The following message is issued for operator interac-
                tion if proper memory is installed.

        Do you want to display the contents of the memory error log
        (yes/no) ?

        The default answer is "NO".

NOTE: Content of is not valid if initialization has not
occurred since last power on.

Step 13 - Memory Configuration

This step determines what 64K byte memory blocks are
present, by writing to and reading from the first word
of each 64K byte block found. A memory map is dis-
played showing the system memory configuration.

Step 14 - Reads memory status and confirms proper controller is
installed.

Step 15 - Low memory test for single bit errors

This test step does a preliminary test of the ELA in
the low memory area. Then, the program area of memory
is read amd single bit errors are noted. Before the
preliminary ELA check is performed, the diagnostic asks
the question:

O.K to clear memory error log (ELA)??(Y/N)
>

This allows the operator to elect not to clear the
error log and late reconfigure the diagnostic to
eliminate any ELA checks.

The default answer to this question is "YES".

If the operator responds with "YES", then the first 256
locations in the ELA are written with ones and verifed.
Then the first 256 locations are written with zeros and
verified. This completes a preliminary ELA test.

If the operator responds with "NO", then the
preliminary ELA check is not done and the first 256
locations of the ELA are not cleared. Thus, any
previously logged errors in this area of the ELA will
show up as RAM failures in the next part of the test.

Possible error messages during the preliminary ELA
checks are:

"Controller 0"
"Address: !XXXXX"
"Wrote 'Y' to error log read 'Z'"

Where: XXXXX = ELA addresss where the error occured
        Y = 0/1
        Z = 1/0

3-2

During the read of the program area for single bit
errors, the possible error messages include:

"Error at SMA 0; Row 'X'; Bit Location 'WXY'."
"SYNDROME CODE % ZZZ"

or

"Double/Multi Bit Error Detected" ›
"Error at SMA 0; Row 'X';"
"SYNDROME CODE % ZZZ"

Where: X = 0 or 1
        W = D or C (Data bit or Check bit)
        YY = 0-31 for data bits
            0-6 for checks bits
        ZZZ = Octal syndrome code

If an error is detected, the program will then print
out a message indicating that corrective action should
be taken to insure proper operation of the diagnostic.

Step 16 - Checks the error log for any errors. If any errors have
been logged, the diagnsotic prints out the following
message:

"possible errors have been detected by controller 'X'"
Where: X = 0 or 1

3.2 TEST SECTION 2

This section verifies that the Error Logging Array (ELA) is min-
imally functional, by writing/reading zeros (0) and ones (1) to
and from all cells.

Step 21 - ELA Check with Ones/Zeros

The ELA is written with all ones and verified. Then it
is written with all zeros and verified. An error mes-
sage would be of the form:

"Controller wrote 'X' to ELA address EEEE AAAAAAA and read 'Z'"

Where: X = 0/1
        Z = 1/0
        EEEE AAAAAAA = ELA address

Step 22 - ELA Check with marching 1/0

> This step marches ones through zeros and zeros through ones in the ELA. The ELA is written to a background of zeros. In ascending order each cell is read for a zero and then written to a one and verified until the whole ELA contains ones. The process is repeated for zeros with the current background of ones in descending order until the whole ELA contains zeros. An error message would have the same form as the previous step.

Controler X
Wrote X to error log read Z
    Where: X = 0/1
           Z = 1/0

Controller X
Marching zeros/ones Test to ELA X fails
    Where: X = 0/1

3.3 TEST SECTION 3

This section verifies both the single error detection, correction and logging into the ELA and double error detection into traps in segment 1.

Step 31 - Single Bit Error Test

> This step verifies proper logging and correction of single bit errors by:
>
> a. Verifying the 39 bit test word to be error free.

NOTE

> This requires disabling of error correction. Therefore; no errors can be tolerated in the executing program or stack areas.

> b. Creating all 39 possible single bit errors.
>
> c. Verifying proper error correction.
>
> d. Checking error log for proper error logging.
>
> Error messages would be of the following form:

Step 32 - Double Bit Error Test

       This step verifies that all possible double bit syn-
diome error codes can be generated by the memory con-
troller(s). Trap servicing this kind of error returns
control from segment 1 (traps) back to segment 3 (Pro-
gram).

       Error messages would be of the form:

Controller X
Double bit error not detected; Syndrome = YYY

       Where: X = 0/1
               Y = The octal syndrome error code. Refer to
                     Appendix A for further information

3.4 TEST SECTION 4

This section verifies memory cells, by writing and reading words
to each 64K byte block and then verifying them.

Step 41 - Alternate One's and Zero's Test

       This step writes alternate one's and zero's (%125252)
sequentially, in ascending address order, into the mem-
ory block then reads them back. The complement pattern
(%52525) is then written and read back.

Step 42 - Data Patterns

       This step writes all possible data patterns to one ad-
dress of each 64K byte block; to log an error in the
ELA when an IC chip is weak.

Step 43 - Move Data

       This step uses the lowest 32K bytes as data and copies
it to and verifies it in all upper 32K byte blocks,
using the MOVE instruction. The MOVE instruction exer-
cises memory significantly faster than the standard
LOAD and STORE.

Step 44 - March Ones/Zeros

This step writes a background of ones into a 64K byte
memory block.   Then reads and verifies every word
starting at the lowest address.   Then the data is re-
written to be all zeros, before moving on to the next
address. When the end of the block is reached all
words contain zeros. Then this step executes the same
with zeros background and ones data, starting at the
highest address. A tested 64K memory block has a back-
ground of ones, when this step is completed.
In addition, if a double-bit error is encountered while
step 44 is ececuting, the diagnostic will march a "1"
through the memory location with the double bit error.
It will then march a "0" through tyhe same location.
This will cause most double bit errors to appear in the
error log as two single bit errors.

Error messages would be of the form:

"Address: !XXXXX"
"Controller 'Q'; SMA 'R'; Row 'S;   TTTTT Array"

"Expected: YYYY YYYY YYYY YYYY"
"Read EC on: ZZZZ ZZZZ ZZZZ ZZZZ"

Where: XXXXX = Address in hex of the error word.
          Q = 0 or 1

R = SMA thumbwheel number (0-7)
S = Row number (0-3) on SMA
TTTTT = Either 256KB or 1MB (depending on
        array type)
Y = Expected data in binary.
Z = Actual data in binary.

3.5 TEST SECTION 5

This section checks the lowest 64K bytes of memory.

Step 51 - Read

This step reads the lowest 128K bytes of memory.

Step 52,53,54 - Relocate Program

> These steps will relocate the program from bank 0 into
> bank 1. They will then perform a marching of ones and
> zeros throughout bank 0 (the lowest 128K bytes of mem-
> ory). If an error is detectyed, the program sets an
> error flag. At the end of the marching test, the diag-
> nostic will relocate itself back into bank 0. If an
> error has been detected, the diagnostic will print an
> error message with following format:

> "Address: !XXXXX"
> "Controller 'Q'; SMA 'R"; Row 'S';   TTTTT Array"

> "Expected: YYYY YYYY YYYY YYYY"
> "Read EC on: ZZZZ ZZZZ ZZZZ ZZZZ"

> Where: XXXXX = Address in hex of the error word.
>   Q = 0 or 1
>   R = SMA thumbwheel number (0-7)
>   S = Row number (0-3) on SMA
>   TTTTT = Either 256KB or 1MB (depending on
>   array type)
>   Y = Expected data in binary
>   Z = Actual data in binary

Due to limitations imposed by the relocation of the diagnostic,
only one error will be reported. This error is the last error
encountered by the march test. Upon successful completion, the
program will continue.

3.6 TEST SECTION 6

> This section executes a special test to controller(s), then
> resets them.

Step 61 - Address Paths

> This step verifies unique addressability to 64K byte
> blocks, by walking zeros through ones in the last 2
> bytes of each 32K byte block. An error message would
> be of the form:

"Writing 'DATA NOT' to mm 32K byte block affects data at nn
32K byte block."

> EXPECTED: XXXX XXXX XXXX XXXX
> READ EC on: XXXX XXXX XXXX XXXX
> Where: mm and nn = 32K byte block numbers.
>   X = Expected data
>   Y = Actual data

3.7 TEST SECTION 7

This is the interactive section, requiring operator intervention.
This section is skipped, when the test selection is in default
mode.

Step 71 - Test Light - test

> This step creates a single bit error, a double-bit
> hardware detectable error, and initialized write activ-
> ity. Then, the step goes into an active mode to allow
> the operator to observe the test lights. Testing of
> the RESET switch is possible in this step by pressing
> the switch and observing the extinguishing of test
> lights P and E, or it will be done programatically.
>
> The user will be instructed to check that lights P and
> E are ON, then told to press the console RETURN key.
> The program will then extinguish lights P and E. At
> the same time, the program will light both lights A and
> I. The user will be instructed to check that lights P
> and E are OFF and that lights A and I are ON. The user
> will then be told to press the console RETURN key and
> observed that the I light is now extinguished.

> NOTE: Test light A is always ON to some extent. The
>   test allows light A to be brighter, thus easier
>   to observe. It is never fully extinguished.

> Test Lights
>
> ```
> +-+
> |o|  A --> activity (Read/Write/Refresh)
> |o|  I --> initialize write
> |o|  C --> second controller
> |o|  E --> single bit correctable error
> |o|  P --> parity (double bit) error
> +-+
> ```

Step 72 - Refresh test

> This step writes a binary test pattern of alternating
> ones and zeros into memory. The user will be asked to
> wait two seconds before proceeding. Operator interven-
> tion is required to resume the test to verify no loss
> of data occurred.

3.8 TEST SECTION 8

This section verifies that the controller can perform initialized
writes.   The controller uses the initialized write function to
initialize memory at eery power-on (that is not the result of a
powerfail-recovery).

Step 81 - Initialized Write Check

This step uses the least significant word of a two word
address to write to the upper half of bank 1 in
memory using initialized writes.   The test then reads
this portion of memory and assures that the data was
written correctly to both words of memory.   If the low
test moduale specified is greater than 31, only the
upper controller is tested.

Error messages would be of the form:

"Controiller 'R'"
"Initialized Write Error"
"Attempted to write Word 'Z' address into both Word A
and Word B"
"Word A address:=!YYYYY"
"Word B address:=!XXXXX"

"Expected: **MMMM MMMM MMMM MMMM**"
"Word A: **NNNN NNNN NNNN NNNN**"

   Where: R = 0 or 1
             Z = A or B
             YYYYY = Address of Word A in Hex
             XXXXX = Address of Word B in Hex
             M = Expected data in binary
             N = Word A data in binary
             P = Word B data in binary

The last error message depends on what words agreed
with each other.   The last message is of one of the
following forms:

"Word A not equal to Word B"
         OR
"Word A equal to Word B but not equal to Expected"

3.9 TEST SECTION 9

This test section checks the error log for any logged errors be-
fore the diagnostic returns control to the Diagnsotic/Utility
System.   Error messages are of the following form:

   "possible errors have been dtected by controller 'X'"

   Where: X = 0 or 1

```
+----------------------------------------------------+ +----------+
|                                                    | | APPENDIX |
|        ERROR LOG ARRAY SYNDROME ERROR CODES        | |----------|
|                                                    | |    A     |
+----------------------------------------------------+ +----------+
```

                            .

Error Log Array Syndrome Error Codes

This list contains all possible syndrome error codes returned
from the memory controller when the memory error log is read.
There are thirty nine (39) possible single bit errors and sixty
three (63) double bit error pairs. An error code syndrome of 7.32
means that a double bit error occurred and that the two bits
involved are either bits 2 and 23, 3 and 37 .... or 15 and 26.

OCTAL SYNDROME ERROR CODE

```
177   NO ERROR
176   38
175   37
174   0/36 1/25 4/35  5/34  6/33  7/32  8/16  9/17 10/18 11/19
      12/22 13/23 15/21 37/38
173   36
172   0/37 2/22 4/16  5/17  6/18  7/19  8/35  9/34 10/33 11/32
      15/27 36/38
171   0/38 2/12 3/14  4/ 8  5/ 9  6/10  7/11 16/35 17/34 18/33
      19/32 20/26 21/27 36/37
170   0
167   35
166   0/16 1/17 4/37  6/20  7/21  8/36  9/25 10/26 11/27 12/28
      13/29 14/30 15/32 35/38
165   0/ 8 1/ 9 4/38  7/15 16/36 17/25 10/26 19/27 20/33 21/32
      22/28 23/29 35/37
164   4
163   0/ 4 1/ 5 8/38 11/15 16/37 18/20 19/21 24/31 25/34 26/33
      27/32 35/36
162   8
161   16
160   0/35 1/34 2/28  3/30  4/36  5/25  6/26  7/27  8/37 10/20
      11/21 15/19 16/38
157   34
156   0/17 1/16 2/18  5/37  6/22  7/23  8/25  9/36 12/33 13/32
      14/31 15/29 34/38
155   0/ 9 1/ 8 2/10  5/38  6/12  7/13 16/25 17/36 20/20 21/29
      22/33 23/32 34/37
154   5 .
153   0/ 5 1/ 4 2/ 6  9/38 10/12 11/13 17/37 18/22 19/23 24/30
      25/35 26/28 27/29 34/36
152   9
151   17
150   0/34 1/35 2/33  3/31  4/25  5/36  9/37 10/22 11/23 12/18
      13/19 17/38
147   0/ 1 4/ 5 8/ 9 13/15 16/17 20/22 21/23 25/36 28/33 29/32
      30/31 34/35
146   NOT A SINGLE OR DOUBLE BIT ERROR
145   NOT A SINGLE OR DOUBLE BIT ERROR
144   0/25 1/36 2/26  3/24  4/34  5/35  6/28  7/29  8/17  9/16
      12/20 13/21 15/23
```

```
                        LEGEND:
                   0-31 = D0-D31
                     32 = C0
                     33 = C1
                     34 = C2
                     35 = C3
                     36 = C4
                     37 = C5
                     38 = C6
```

OCTAL   SYNDROME ERROR CODE

```
143   25
142   1/37 2/20 4/17  5/16  8/34  9/35 10/20 11/29 12/26 13/27
      14/24 25/38
141   1/38 4/ 9 5/ 8 16/34 17/35 18/28 19/29 22/26  23/27 25/37
140   1
137   33
136   0/18 2/17 3/19  4/20  5/22  6/37  8/26 10/36 12/34 13/31
      14/32 1530 33/38
135   0/10 2/ 9 3/11  5/12  6/38  7/14 16/26 18/36 20/35 21/30
      22/34 23/31 33/37
134   6
133   0/ 6 2/ 5 3/ 7  9/12 10/38 11/14 16/20 17/22 18/37 24/29
      25/28 26/35 27/30 33/36
132   10
131   18
130   0/33 1/28 2/34  3/32  4/26  6/36  8/20  9/22 10/37 12/17
      14/19 18/38
127   1/ 2 4/ 6 8/10 14/15 16/18 20/37 26/36 28/34 29/31 30/32
      33/35
126   NOT A SINGLE OR DOUBLE ERROR
125   20
124   0/26 2/25 3/27  4/33  5/28  6/35  7/30  8/18 10/16 14/21
      20/38
123   26
122   0/20 1/22 3/21  4/18  6/16  8/33  9/28 10/35 11/30 12/25
      13/24 14/27 26/38
121   1/12 3/15 4/10  6/ 8 16/33 17/28 18/35 19/30 20/36 22/25
      23/24 26/37
120   NOT A SINGLE OR DOUBLE BIT ERROR
117   0/ 2 5/ 6 9/10 12/38 13/14 17/18 22/37 24/27 25/26 28/35
      29/30 31/32 33/34
116   12
115   22
114   1/26 2/36 4/20  5/33  6/34  7/31  9/18 10/17 12/37 14/23
      22/38
113   NOT A SINGLE OR DOUBLE BIT ERROR
112   0/22 1/20 2/37  3/23  5/10  6/17  8/20  9/33 10/34 11/31
      12/36 15/24
111   0/12 2/38 5/10  6/ 9 16/28 17/33 18/34 19/31 20/25 21/24
      22/36
110   2
```

```
                              LEGEND:
                        0-31 = D0-D31
                          32 = C0
                          33 = C1
                          34 = C2
                          35 = C3
                          36 = C4
                          37 = C5
                          38 = C6
```

OCTAL SYNDROME ERROR CODE

```
107   28
106   1/18 2/16 4/22  5/20  9/26 10/25 11/24 12/35 13/30 14/29
      15/31 28/38
105   1/10 2/ 8 4/12 17/26 18/25 19/24 20/34 21/31 22/35 23/30
      28/37
104   NOT A SINGLE OR DOUBLE BIT ERROR
103   1/ 6 2/ 4 8/12 16/22 17/20 24/32 25/33 26/34 27/31 28/36
102   NOT A SINGLE OR DOUBLE BIT ERROR
101   NOT A SINGLE OR DOUBLE BIT ERROR
100   0/28 1/33 2/35 3/29 5/26 6/25 7/24 8/22 9/20 12/16
077   32
076   0/19 3/10 4/21 5/23 7/37 8/27 11/36 12/31 13/34 14/33 15/35
      32/38
075   0/11 3/10 4/15 5/13 6/14 7/38 16/27 19/36 20/30 21/35 22/31
      23/34 32/37
074   7
073   0/ 7 3/ 6 8/15 9/13 10/14 11/30 16/21 17/23 19/37 24/28
      25/29 26/30 27/35 32/3
072   11
071   19
070   0/32  1/29  2/31  3/33  4/27  7/36  8/21  9/23 11/37 13/17
      14/18 15/16 19/38
067   4/ 7  8/11 15/38 16/19 21/37 27/36 28/31 29/34 30/33 32/35
066   15
065   21
064   0/27  2/24  3/26  4/32  5/29  6/30  7/35  8/19 11/16 14/20
      15/37 21/38
063   27
062   0/21  1/23  3/20  4/19  7/16  8/32  9/29 10/30 11/35 12/24
      13/25 14/26 15/36 27/3
061   0/15  1/13  4/11  7/ 8 16/32 17/29 18/30 19/35 21/36 22/24
      23/25 27/37
060   NOT A SINGLE OR DOUBLE BIT ERROR
057   2/ 3  5/ 7  9/11 12/14 13/38 17/19 23/37 24/26 25/27 28/30
      29/35 31/33 32/34
056   13
055   23
054   1/27  4/29  5/32  6/31  7/34  9/19 11/17 13/37 14/22 23/38
053   NOT A SINGLE OR DOUBLE BIT ERROR
052   0/23  1/21  3/22  5/19  7/17  8/29  9/32 10/31 11/34 13/36
      15/25
```

LEGEND:
0-31 = DO=D31
  32 = C0
  33 = C1
  34 = C2
  35 = C3
  36 = C4
  37 = C5
  38 = C6

OCTAL SYNDROME ERROR CODE

```
051   0/13  1/15  2/14  3/12  5/11  7/ 9 16/29 17/32 18/31 19/34
      20/24 21/25 23/36
050   NOT A SINGLE OR DOUBLE BIT ERROR
047   29
046   1/19  4/23  5/21  9/27 10/24 11/25 12/30 13/35 14/28 15/34
      29/38
045   1/11  4/13  5/15 17/27 18/24 19/25 20/31 21/34 22/30 23/35
      29/37
044   NOT A SINGLE OR DOUBLE BIT ERROR
043   1/ 7  8/13  9/15 16/23 17/21 24/33 25/32 26/31 27/34 29/36
042   NOT A SINGLE OR DOUBLE BIT ERROR
041   NOT A SINGLE OR DOUBLE BIT ERROR
040   0/29  1/32  2/30  3/28  5/27  6/24  7/25  8/23  9/21 13/16
      15/17
037   0/ 3  6/ 7 10/11 12/13 14/38 18/19 20/21 22/23 24/25 26/27
      21/26
036   14
035   NOT A SINGLE OR DOUBLE BIT ERROR
034   1/24  3/36  4/30  5/31  6/32  7/33 10/19 11/10 12/23 13/22
      14/37 15/20
033   NOT A SINGLE OR DOUBLE BIT ERROR
032   2/23  3/37  6/19  7/18  8/30  9/31 10/32 11/33 14/36 15/26
031   0/14  2/13  3/38  6/11  7/10 16/30 17/31 18/32 19/33 20/27
      21/26
030   3
027   30
026   3/16  6/21  7/20  9/24 10/27 11/26 12/29 13/28 14/35 15/33
      30/38
025   3/ 8  4/14  6/15 17/24 10/27 19/26 20/32 21/33 22/29 23/28
      30/37
024   NOT A SINGLE OR DOUBLE BIT ERROR
023   3/ 4  8/14 10/15 18/21 19/20 24/34 25/31 26/32 27/33 30/36
022   NOT A SINGLE OR DOUBLE BIT ERROR
021   NOT A SINGLE OR DOUBLE BIT ERROR
020   0/30  1/31  2/29  3/35  5/24  6/27  7/26 10/21 11/20 14/16
      15/18
017   31
016   2/19  3/17  6/23  7/22  8/24 12/32 13/33 14/34 15/28 31/38
015   2/11  3/ 9  5/14  6/13  7/12 16/24 20/29 21/28 22/32 23/33
      31/37
014   NOT A SINGLE OR DOUBLE BIT ERROR
```

```
                                     LEGEND:
                                0-31 = D0-D-31
                                  32 = C0
                                  33 = C1
                                  34 = C2
                                  35 = C3
                                  36 = C4
                                  37 = C5
                                  38 = C6
```

OCTAL SYNDROME ERROR CODE

```
013   2/ 7   3/ 5   9/14 10/13 11/12 18/23 19/22 24/35 25/30 26/29
      27/28 31/36
012   NOT A SINGLE OR DOUBLE BIT ERROR
011   NOT A SINGLE OR DOUBLE BIT ERROR
010   0/31   1/30   2/32   3/34   4/24 10/23 11/22 12/19 13/18 14/17
007   1/ 3 12/15 20/23 21/22 24/36 28/32 29/33 30/34 31/35
006   NOT A SINGLE OR DOUBLE BIT ERROR
005   NOT A SINGLE OR DOUBLE BIT ERROR
004   0/24   2/27   3/25   4/31   5/30   6/29   7/28 12/21 13/20 15/22
003   24
002   2/21   8/31   9/30 10/29 11/28 12/27 13/26 14/25 24/38
001   1/14   2/15 16/31 17/30 18/29 19/28 22/27 23/26 24/37
000   NOT A SINGLE OR DOUBLE BIT ERROR
```

```
                            LEGEND:
                        0-31 = D0-D31
                          32 = C0
                          33 = C1
                          34 = C2
                          35 = C3
                          36 = C4
                          37 = C5
                          38 = C6
```

HP 3000 Series 44 CMP Maintenance Mode

CMP Maintenance Mode Manual

HEWLETT
PACKARD

The List of Effective Pages gives the date of the current edition and of any pages changed in updates to that edition. Within the manual, any page changed since the last edition is indicated by printing the date the changes were made on the bottom of the page. Changes are marked with a vertical bar in the margin. If an update is incorporated when an edition is reprinted, these bars are removed but the dates remain.

All pages in this manual are original issue dated Jan 1981.

PRINTING HISTORY

New editions are complete revisions of the manual. Update packages, which are issued between editions, contain additional and replacement pages to be merged into the manual by the customer. The date of the title page of the manual changes only when a new edition is published. When an edition is reprinted, all the prior updates to the edition are incorporated.

First Edition. . . . . Jan 1981

CONTENTS

ILLUSTRATIONS

1.0   INTRODUCTION

This manual describes the Control Maintenance Processor (CMP) functions and user interaction when the system is in Maintenance Mode.

The CMP contains the microcode which allows the system or remote console to be used as a troubleshooting aid via the maintenance display. The console screen becomes a maintenance panel which displays the contents of many registers and the state of principal signals in the computer.

1.1   REQUIRED HARDWARE

The system configuration required to use all of the maintenance display functions is shown in figure 1-1.
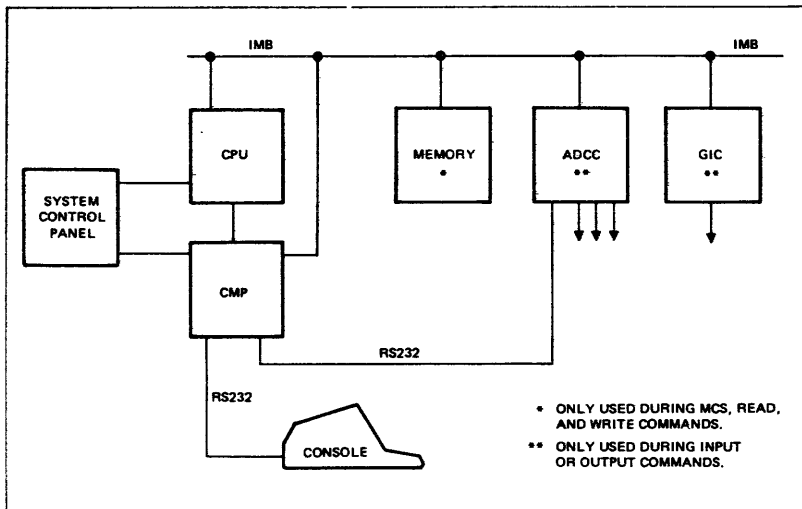
Figure 1-1.   System Architecture For Maintenance Mode

1.2 OPERATING INSTRUCTIONS

To cause the maintenance display to appear when the CPU is halted
or microhalted, enter the DISPLAY command. If the CPU is running
preceed the DISPLAY command with a Control B character. If the
message "DISABLED" appears, set the MAINTENANCE MODE switch on
the system front panel to ON and re-enter the DISPLAY command.

Note

While entering the DISPLAY command when the system
is running, any characters about to be sent from
the ADCC to the console may not be received.

Once the maintenance display is invoked, any maintenance display
commands entered must be preceeded by a Control B character if
the CPU is microrunning and program running.

There are two modes of operation the maintenance display can be
used in; Hardware (HW) or Software (SW). Under HW mode, entering
commands that require the CPU to be halted will cause the CPU to
be microhalted.   Under SW mode, the CPU will be program halted
when a command to set a CPU register is entered.

A list of applicable commands and their descriptions can be found
in Section II.

1.3 MINI OPERATING INSTRUCTIONS

a. Set the MAINTENANCE MODE switch, on the system front
    panel, to ON.

b. Enter the DISPLAY command
               OR
    Enter a Control B character and then the DISPLAY command
    if the CPU is running.

c. Enter either the HW or SW command to select the desired
    mode. Default is HW.

d. If desired, enter the OCTAL command to change the numer-
    ical items on the Maintenance Display to octal. The de-
    fault is HEX.

e. Enter appropriate Maintenance Display command(s).

2.0 INTRODUCTION

This section contains an example of the Maintenance Display with
descriptions of each item on the display.

2.1 MAINTENANCE DISPLAY FORMAT

An example of the Maintenance Display is shown in figure 2-1.
Some of the flags not normally present are shown in the example
for completness. Inverse video and underline enhancements are
used in the display. Flags which are set appear in uppercase,
are underlined, and in inverse video.

```
RA    2024       PB    000C              MAINTENANCE DISPLAY 1.0              MIR   EC8A0571AFFF
RB    1A4F        P    0AB3        DL    1BDA     ABNK    0000        CSAR  1861
RC    31FA       PL    10AF        DB    1C58     BBNK    0001        UBUS 0000
RD    0006      PBNK   0001        Q     2A91     DBNK    0001     RUN        MHALT
R4    31FA    R16   0000   R28   0008   SM    3554     SBNK    0001     PARITY     UPDATE OFF
R5    0000    R17   269B   R29   0020   Z     3AA9                      TIMEOUT    DISABLED
R6    297F    R18   0001   R30   FFFF   SWCH  0231
R7    297F    R19   0503   R31   36B8   STA   44DE              m l t r o C e l
R8    000F    R20   10AF   R32   0023   SIR   200E          CSRQ irq nrdi datn disp ICS SS.
R9    00FF    R21   0200   R33   0000   SRR   1A5B
R10   0080    R22   0000   R34   A000   CIR   2138
R11   10A3    R23   36E8   R35   0002   CTR   0003                    f1 f2 f3 f4
R12   2429    R24   0020   R36   FFFF   SPO   FFFF         MODE: HW
R13   3555    R25   0024   R37   000A   X     0000         STOPBP    STOP
R14   0000    R26   0020   R38   FFCF   SR    0004         IMB: 013555
R15   0000    R27   3555   R39   4000.  OPND  0006         BP  000000    BPTYPE: CSARBP
```

147026·01

Figure 2-1.   Maintenance Display Format

2.2 MAINTENANCE DISPLAY DESCRIPTION

The following is a description of each item in the display:

RA, RB, RC, RD - These registers are usually the Top Of Stack (TOS) registers. Their values are shown regardless of whether the data is valid or not.

R4 thru R39 - These registers are scratch pad registers used by the CPU microcode for intermediate results. Register R14 will always be zero and is used by the CPU as a source of zeros.

PB, P, PL, PBNK - These registers point into the current code segment. PBNK holds bank bits for the PB, P, and PL registers.

DL, DB, Q, SM, Z - These are data area registers. They use the DBNK and SBNK registers for their bank bits. The SM register is the pointer to memory TOS and will have the value of S minus SR.

SWCH - This register contains 16 bits. They are externally supplied and include bits from the system control panel. The bit configuration is:

```
0-3      zero
4        Selftest      Switch on CPU board
5                      Not used
6        <>PBUS        Processor has IMB if PBUS is true (=0)
7        Head 2        Normal/Split System Disc switch
8-9      ED0,1         00 = NOP
                       01 = cold load
                       10 = dump
                       11 = warm start
10-12                  Start or dump device
13-15                  start or dump device
```

STA - This is the status register. It is displayed as a 16-bit register with the lower eight bits being the segment number. The upper eight bits are also shown, as individual flags, to the right of the register value. The meaning of the flags are:

FLAG	MEANING IF INVERSE VIDEO	MEANING IF NORMAL VIDEO
M	Privileged Mode	User Mode
I	Interrupts Enabled	Interrupts Disabled
T	User Traps Enabled	User Traps Disabled
R	Right Stackop Pending	Right Stackop Not Pending
O	Overflow Flag Set	Overflow Flag Not Set
C	Carry Flag Set	Carry Flag Not Set
E	Equal Flag	Not Equal
L	Less Than	Not Less Than

SIR - This is the status and interrupt register. To the right of
it is shown what several of its bits represent. They are as fol-
lows:

    CSRQ - Channel Service Request asserted by an I/O device.
    IRQ - Interrupt Request asserted by an I/O device.
    NRDI - Non-responding device interrupt. The IMB handshake
           did not complete.
    DATN - Disable Attention. Disable interrupts to microcode.
    DISP - Dispatcher flag
    ICS - Interrupt Control Stack flag.
    SS   - Split Stack mode.

OPND - Memory operand register.

SP0 - This is a 16-bit scratchpad register used in 32 bit shift-
ing.

SR - This is a four bit register specifying how many of the TOS
registers are valid.

SRR - Microcode subroutine return register.

X - This is the 16 bit index register.

ABNK,BBNK - General purpose bank registers.

CIR - The current instruction register. This usually contains
the current instruction.

CTR - A loop counter register used by microcode.

MIR - The Micro Instruction Register. This shows the value of
the next microinstruction to be executed.

CSAR - Control Store Address Register. This shows the address of
the next microinstruction to be executed if updates are enabled.
If updates are disabled it shows the instruction in rank 0 of the
pipeline.

RUN - Indicates that the CPU run/halt flip flop is set to run.

HALT - Indicates that the CPU run/halt flip flop is set to halt.

MICROHALT - Indicates that the CPU is frozen and is not executing
microinstructions.

MICRORUN - Indicates that the CPU is executing microinstructions.

PARITY - This flag indicates that a parity error has occured on
the IMB. To reset the indicator, type the PARITY command.

UPDATE OFF — This indicates that no screen updates are occuring and consequently the information on the display may not be valid.

TIMEOUT DISABLED — This indicator is present when IMB timeouts will cause the CPU to freeze, otherwise a timeout interrupt will occur in the CPU.

UBUS — This shows the output of the ALU and Shifter of the CPU, when updates are enabled.

F1 thru F4 — These are general purpose flags in the CPU and, if set, will appear in inverse video and upper case.

MODE: — HW (hardware) or SW (software). Indicates the mode of the CMP display updates. Refer to the HW and SW command descriptions for more information.

STOPBP — This function, when enabled, will microhalt the CPU whenever a microinstruction which contains a STOP function is executed. If the CPU is microhalted due to a STOPBP, the STOP indicator will appear in inverse video.

IMB (address) — This field shows the last I/O or memory operation performed on the Intermodule Bus. The upper three bits of the 24-bit number displayed are opcode bits on the IMB. The meanings of the opcode bits are:

    000 — Memory read
    001 — Memory read/write ones
    010 — Memory write
    101 — Memory control/read status
    100 — I/O read
    110 — I/O write

BP (address) — The address shown is the breakpoint address. If a breakpoint has been reached, the BP will be in inverse video and blinking.

BP TYPE — The breakpoint type can be READBP (memory read), WRITE-BP (memory write), INPUTBP (I/O read), OUTPUTBP (I/O write), or CSARBP (control store read).

2.3 DISPLAY UPDATES

Display updates are enabled when the maintenance display is first called up. Updates may be disabled or enabled anytime by entering the UPDATE command. When updates are disabled, the indicator "UPDATE OFF" will be shown in the upper right quarter of the display. No indication is given when updates are enabled.

Host of the indications shown on the maintenance display are only updated in microhalt unless the CPU is in SW mode. However, the following are updated regardless of the state of the CPU:

    RUN
    MHALT
    HW/SW
    BP address
    BP function
    MIR
    CSAR

The remainder of the display is updated after every command if the CPU is microhalted and whenever a microhalt occurs. If the display is in SW mode, the display will be updated if a program halt occurs even if the CPU is in microrun.

When updates are enabled and the display is in HW mode, the CMP may be unable to restore the CPU to its original state after a screen update.

It is possible to microstep through sections of code with updates disabled. The CMP will not force any microinstructions or read the CPU UBUS in this state. Only the CSAR and MIR are updated on the display in this mode.

SW mode allows the display of the CPU state without affecting the execution of code. However, the user cannot explicitly micro-halt or microstep the CPU because, in order to prevent changes to the state of the CPU, the CMP manages the microhalt and microstep functions.

3.0 INTRODUCTION

This section provides the user with Maintenance Display command
descriptions and syntax.   There are five types of commands used
with the Maintenance Display:

    TOGGLE FLAG
    REGISTER STORE
    LONG
    SPECIAL
    DEBUG

Only REGISTER STORE and LONG commands have   optional parameters
that may be used with them.   The following conventions apply to
those types of commands:

    Upper and lower case letters are treated identically.

    Keywords must be separated by one or more blanks or special
    characters.

    An equal sign (=) is interpreted as a blank.

    Octal numbers must have a period (.) separating the bank
    bits from the lower 16 bits.

3.1 MAINTENANCE DISPLAY COMMAND DESCRIPTIONS

Each of the five types of commands are described in the following
paragraphs.

3.1.1 Toggle Flag Commands

Toggle Flag commands are used to toggle their respactive flags to
the desired state.   In some cases some of the commands are not
shown on the maintenance display.   The following list contains
the command name and description.   For more detailed information
about how to use the Toggle Flag breakpoint commands (CSARBP,
READBP, WRITEBP, OUTPUTBP, or INPUTBP) refer to paragraph 3.2 of
this section.

RUN   - This command sets the CPU RUN/HALT flip flop to the
             RUN state.   This allows the CPU to execute program
             instructions.

HALT   - This command sets the CPU RUN/HALT flip flop to the
            HALT state. This prevents the CPU from executing
            program instructions.

MRUN   - This command sets the CPU to the microrun state.
            This allows the CPU to execute microinstructions.

MHALT   - This command sets the CPU to the microhalt state.
            This prevents the CPU from executing microinstruc-
            tions. Whenever the CPU is microhalted, the CPU
            timer and timeouts are disabled.

HARDCOPY - This command causes the maintenance display to be
            printed without using the cursor addressing and
            memory lock functions present on most CRT terminals.
            The display will be reprinted completly after every
            screen update.

PARITY   - This command resets the parity indicator on the CMP.

UPDATE   - This command is used to enable/disable updates. While
            display updates are disabled, the CMP Maintenance
            Display will not necessarily reflect the state of the
            CPU. Even when updates are disabled, the CSAR and MIR
            will continue to be updated.

HW   - This command puts the display in hardware mode and is
            used when it is necessary to analyze microcode execut-
            tion. It is the default mode.

SW   - This command puts the display in software mode and is
            used when it is necessary to debug programs. It
            causes all CPU breakpoints to set the CPU to halt.
            This allows program stepping and setting breakpoints
            without destroying the state of the CPU when updating
            the screen. The following commands are not allowed in
            SW mode: CSARBP, MHALT, MRUN, MS, and STOPBP.

STOPBP   - This command causes the CPU to be microhalted if
            microcode executes an instruction with STOP in the
            store field. If updates are enabled, the CPU will
            microhalt three instructions after the one with STOP
            in it. If updates are disabled, the third instruction
            after the one with STOP in it will be in rank 1.

TIMEOUT   - This command disables CPU IMB timeouts from occurring.
            If an IMB handshake fails to complete, the CPU will
            freeze.   Enter the TIMEOUT command to allow the IMB
            handshake to complete. The default mode is timeouts
            enabled. If timeouts are disabled, the TIMEOUT DIS-
            ABLED message will appear on the Maintenance Display.

CSARBP   – This is the Control Store Adress Breakpoint command.
          It is used to set a microcode address breakpoint.

READBP   – This command is used to set a memory read breakpoint.

WRITEBP   – This command is used to set a memory write breakpoint.

OUTPUTBP – This command is used to set an I/O write breakpoint.

INPUTBP   – This command is used to set an I/O read breakpoint.

OCTAL   – This command causes the numerical fields of the dis-
          play to change from HEX to OCTAL.

HEX   – This command causes the numerical fields of the dis-
          play to change from OCTAL back to HEX which is the
          the default condition when the Maintenance Display. is
          invoked.

3.1.2 Register Store Commands

Register Store commands are used to set the indicated storage
register to the numerical value used in the expression or to the
value found in the register specified in the expression. A Reg-
ister Store command is of the form:

<storage register> <expression>

The storage registers that may be specified in a Register Store
command are: RA, RB, RC, RD, R4 through R13, R15 through R39,
PB, P, PL, PBNK, DL, SM, Q, Z, DBNK, STA, SIR, CTR, SP0, X, BP,
or CSAR.

The only exceptions to setting the indicated register to the
value of the expression are BP and CSAR:

    – If BP is specified in a Register Store command it will
      cause the expression to be transferred to the Breakpoint
      register on the CMP.

    – If CSAR is specified in a Register Store command it will
      cause a JUMP microinstruction to be executed, to set the
      CSAR to a new value.

The expression part of a Register Store command may be a numer-
ical value, another storage register, the sum/difference of other
storage registers, or an unsetable register. Unsetable registers
are: CIR, IMB, MIR, R14, SR, SRR, SWCH, or OPND.

Some examples of Register Store commans are:

    RC 31FA   - This would set register RC to 31FA (HEX).

    RD PBNK   - This would set register RD to the value
                  contained in PBNK register.

    R12 R4-R21 - This would set register R12 equal to the
                  difference between R4 and R21.

    R8 SRR   - This would set register R8 to the value
                  contained in the unsetable register SRR.

3.1.3 Long Commands

Long commands are used to read from or write to memory or I/O.
The READ command reads from system memory. The INPUT command
reads from system I/O. The WRITE command performs writes to the
system memory. The OUTPUT command causes writes to the system
I/O. For write operations, only the lower 16 bits of the data
item are used. For read operations, a maximum of 256 words will
be displayed. The READ, INPUT, and MCS commands will alter the
OPND register in the CPU.

READ (expression 1) (expression 2) - This command performs a sys-
tem memory read from the address specified in expression 1.
Expression 2, which is optional, gives the number of words to
display. If expression 2 is omitted, only one word will be dis-
played.

INPUT (expression 1) (expression 2) - This command performs a
system I/O read from the address specified in expression 1.
Expression 2, which is optional, gives the number of words to
display. If expression 2 is omitted, only one word will be dis-
played. Thhe lower 16 bits of expression 1 are placed on the IMB
as an I/O read address. The address is interpreted as follows:

    A0 - A3   IMB I/O command;
                      0000 = Read Channel Command
                      0010 = Obtain Interrupt Information
                      0100 = Obtain Service Information
                      1000 = Interrupt Poll (global command)
                      1010 = Channel Roll Call (global command)
                      1100 = Service Poll 1 (global command)
                      1110 = Service Poll 2 (global command)
    A4 - A7   K-field, I/O device register number
    A9 - A12   Channel Number (for non-global commands)

MCS (expression 1) (expression 2) - This command is used to read
memory status and read or write to the memory error log. Expres-
sion 1 is the address to be accessed. Expression 2, which is
optional, gives the number of words to display. If expression 2
is omitted, only one word is displayed.

WRITE (address) (data) (data).... - This command causes the data specified to be written to system memory, starting at the address specified. The write is always followed by a memory read of address 0 to check for CPU IMB interface lockup.

OUTPUT (address) (data) (data)... - This command causes the data specified to be written to I/O devices, starting at the address specified. The output is followed by a memory read from address zero to check for CPU IMB interface lockup. The lower 16 bits of the address have the following meaning:

```
A0 - A3    IMB I/O command
             0000 = Write Channel Command
             0010 = Initialize Channel
             0100 = Start I/O Program
             0110 = Halt I/O Program
             1000 = Set Mask (global command)
             1010 = I/O Clear (global command)
A4 - A7    K-field, I/O register number
A9 - A12   Channel Number (for non-global commands)
A13 - A15  Device Number (for non-global commands)
```

3.1.4   Special Commands

HELP   - This command causes the list of valid commands to be displayed.

EXIT   - This command causes the maintenance display firmware to be de-activated. The maintenance display will be allowed to roll off the screen. All breakpoints will be disabled and the CPU set to microrun.

RESET   - This command causes a system reset. If the CPU is microhalted in HW mode, then a screen update will be performed following reset, and then another reset will be performed. The system reset signal is asserted for approximately 120 milliseconds on each reset, and the CMP will microstep the CPU during the reset if the CPU is microhalted.

MS (microstep) - This command causes one microinstruction to be executed. One major CPU clock cycle occurs. The command is ignored unless the CPU is microhalted. Updates must be disabled to microstep since updates will affect the internal state of the CPU. Timer interrupts and IMB timeouts are disabled while microstepping.

PS (programstep) - This command causes one program instruction to
be executed. The command is ignored if the
CPU run/halt flip flop is set to run or if the
CPU is microhalted. This command performs the
following:

(1) It microhalts the CPU in the halt loop.
(2) It forces the following 2 lines of microcode:

SIR 0401 IORL - SIR   Set RUN and DISABLE in SIR
SIR 0201 IORL - SIR NEXT Set RUN, clear DISABLE

(3) It activates the Toggle-Run signal from the
system control panel to the CPU.
(4) It sets the CPU to microrun.

Instructions that are interruptable (i.e., MOVE instructions and
many of the longer decimal or cobol instructions) will be inter-
rupted by the system control panel. Consequently, many PS's will
be required to complete the instruction.

Note

Loading the MS and PS commands into the
softkeys of a 264X terminal is suggested,
when possible.

3.1.5 DEBUG Commands

DEBUG commands are available for maintenance personnel to test or
analyze microcode on the CMP. They are not listed by the HELP
command.   Each command must be preceeded by a control Z. The
DEBUG commands are:

P <hex number> - This command causes the MC5 processor on the CMP
to start executing code at the address specified
by the HEX number.

R <hex number> - This command does a memory read from the HEX
address specified.   The value stored at that
memory address is printed.   Note that reading
from the UARTS is done with the R command since
the UARTS exist in the memory space.

W <hex number> <hex number> - This command does a memory write to
the address specified by the first HEX number.
The second HEX number represents the data to be
written into that location.

I -   This command reads and prints the values found
in all eight CMP external registers.

O <hex number> <hex number> - This command performs an I/O write to the external register (0-7) specified by the first HEX number. The second HEX number represents the data to be written.

S - This command sets CMP memory locations !700-!7FF to zero. The first failure detected during SELFTEST or IOMAP will cause the current stack area, starting at location 0, to be moved to the stack buffer area starting at location !700. Only the first failure will be logged. The S command must be re-entered to log the next failure.

3.2 USING BREAKPOINT TOGGLE FLAG COMMANDS

If any one of the following commands is entered and that breakpoint is already displayed in the BREAKPOINT TYPE field of the display, that breakpoint will be disabled:

    CSARBP
    READBP
    WRITEBP
    OUTPUTBP
    INPUTBP

If any one of the above commands is entered and that breakpoint is not presently displayed in the BREAKPOINT TYPE field of the display, a breakpoint halt will be enabled if the address specified by the breakpoint occurs.

In order to use breakpoints, there must first be a breakpoint address and one of the breakpoint commands enabled. Only one type of breakpoint can be enabled at a time. To set the breakpoint address, enter the BP (address) command. To enable the desired breakpoint, enter CSARBP for a microcode address breakpoint, READBP for a memory read breakpoint, WRITEBP for a memory write breakpoint, OUTPUTBP for an I/O write breakpoint, or INPUTBP for an I/O read breakpoint. To disable a breakpoint, re-enter the name of the present breakpoint.

CSARBP can only be used in hardware (HW) mode. Unless the user wishes to restart the section of code from the beginning, care should be taken to ensure the CPU is not halted in a section of code which cannot be restarted. It is better to have updates disabled when using CSARBP and, if necessary, microstep to get to a safe section of code. If the CPU is in the middle of a JUMP and updates are enabled, the execution of microcode will be affected When updates are enabled, the CPU will be stopped three instructions following the breakpoint address. With updates disabled, the CPU will be stopped with the third instruction following the breakpoint instruction in rank 1 and the fourth instruction after the breakpoint instruction showing in rank 0 (CSAR on display).

From a hardware standpoint, memory breakpoints (READBP/WRITEBP) are very simple; the CPU is microhalted whenever the memory at the breakpoint is accessed. Unfortunately, the use of the breakpoint when debugging is not so simple, due to the way the CPU operates. If a breakpoint is set on an instruction following a conditional branch, the instruction following the conditional branch is read from memory, regardless of whether the branch will be taken or not. This is because the CPU fetches the next sequential instruction before it finishes executing the previous instruction. To resolve this problem, set the breakpoint two instructions following the conditional branch.

In HW mode, when a breakpoint on an instruction is reached, the previous instruction is still being executed. For example, if the instruction preceeding the breakpoint instruction is a store to memory, the store may not have occurred yet. If you wish to finish execution of the previous instruction, set the CPU to program halt, then set the CPU to microrun. The CPU will halt having finished the previous instruction but not yet having started execution of the breakpoint instruction. However, if the breakpoint is still set on that instruction, setting the CPU to RUN will immediately cause another breakpoint halt since the instruction will be read from memory again.

In SW mode, The instruction which was being executed when the breakpoint occurred will usually be completed. The interruptable instructions such as MOVES, decimal instructions, and cobol instructions may be interrupted. SW mode is the preferred method for debugging software, since the state of the CPU is not destroyed and the current instruction is usually completed.

Remember that the breakpoint works regardless of who accesses the memory or why the memory is accessed. Though your code may not be reading or writing to a memory location, the channel program processor (part of the CPU) or the GIC may read or write to those locations, and these will cause a breakpoint to occur.

The following error messages may be received when using the CMP
MAINTENANCE MODE. Generally, serious error messages are display-
ed in inverse video.

CAN'T MHALT – CPU cannot be microhalted. Probably due to clock
failure on the CPU or microstep logic on the CMP.

CAN'T MS – CPU is frozen. The freeze may be an internal CPU
freeze or CMP failure in its breakpoint or microstep
logic.

CAN'T PS – The CPU cannot be program stepped because it is micro-
halted or program running.

CAN'T SET – The register specified cannot be set. Unsettable
registers include CIR, R14, and SWCH.

GO TO HW – The command entered is not valid unless HW mode is
enabled. Enter the HW command before attempting the
operation again.

INVALID – USE HELP – The command entered was not a valid CMP com-
mand in the current mode. HELP will print a
valid list of commands. Note that mainten-
ance display commands are not valid unless
the DISPLAY command has been entered.

INVALID EXPRESSION – The command required a valid numeric expres-
sion. The expression did not exist or could
not be decoded. Remember to use only HEX
numbers in HEX mode and only OCTAL numbers
in OCTAL mode.

TOO MUCH DATA – When specifying data to be stored in memory, too
many data words were specified.

TURN UPDATE OFF – Updates must be turned off with the UPDATE com-
mand before performing this function.
Specifically, microstep cannot yield meaningful
results with updates enabled.

HP 3000 Series 40/44 Computer Systems

CMP Remote Maintenance

Manual

HEWLETT PACKARD

19447 Pruneridge Ave., Cupertino, California 95014

LIST OF EFFECTIVE PAGES

The List of Effective Pages gives the date of the current edition and of any pages changed in updates to that edition. Within the manual, any page changed since the last edition is indicated by printing the date the changes were made on the bottom of the page. Changes are marked with a vertical bar in the margin. If an update is incorporated when an edition is reprinted, these bars are removed but the dates remain.

PRINTING HISTORY

CONTENTS

SECTION I - GENERAL INFORMATION

SECTION II - OPERATING INSTRUCTIONS

APPENDIX A - CMP COMMAND DESCRIPTIONS

APPENDIX B - IOMAP COMMAND DETAILED DESCRIPTION

1.0   INTRODUCTION

This section contains information on hardware requirements, soft-
ware requirements,   and limitations   for   using   the   Control and
maintenance Processor (CMP)   Remote Maintenance   Facility   of   an
HP 3000 Series 40/44 computer system.

1.1   REQUIRED HARDWARE

The CMP Remote Maintenance Facility requires the following   hard-
ware:

   a. An HP 3000 Series 40/44 computer system with at least 256
      kbyte of main memory.

   b. The CMP PCA must be installed.

   c. A Bell 212 compatible modem with   auto-answer   capability
      must be connected to the REMOTE   CONSOLE/ADCC1 connector.

   d. A cold load device (Tape Unit) if the   Diagnostic/Utility
      System (DUS) diagnostics are to be used.

   e. An HP 26XX terminal, an RTC type telephone with exclusion
      key,   and a Bell 212 compatible modem   at the remote ser-
      vice site.

1.2   REQUIRED SOFTWARE

The CMP Remote Maintenance Facility requires no additional   soft-
ware,   since   all necessary microcode is resident   on the CMP and
CPU PCAs.   If DUS is to be used, then a DUS tape is necessary   at
the computer site.

1.3   LIMITATIONS

Refer to the CMP Maintenance Mode manual (p/n 30090-90007) or the
CMP/System Selftest manual (p/n 30090-90005) for any limitations.
When using the   DUS   refer to the appropriate   diagnostic   manual
contained in the Diagnostic Manual Set (p/n 30070-60068).

Operating Instructions

2.0   INTRODUCTION

This section contains information on how to implement the CMP Re-
mote Maintenance Facility so that it can be used to run CMP micro
diagnostics from a remote terminal.

2.1   CMP REMOTE MAINTENANCE FACILITY IMPLEMENTATION

Once it has been decided to use the CMP Remote Maintenance Faci-
lity to diagnose the problem, use the following procedure to
implement the facility.

   a. Establish telephone contact with the System Operator and
      ask them to set the REMOTE COMSOLE, CONTROL FUNCTION, and
      MAINTENANCE MODE switches on the the system front panel
      to the ON position.

   b. Ensure that both modems are set to the same baud rate.
      Ensure that both the remote terminal and system console
      are set to the same baud rate.

   c. If MPE is running, ask the system operator to use the MPE
      :SPEED command to set the CMP and MPE to the   same baud
      rate.   If MPE is not running, the CMP SPEED command will
      do the same thing. Refer to Appendix A for CMP command
      descriptions.

   d. Set the remote console switches as follows:

         REMOTE switch to active (depressed)
         DUPLEX switch to FULL
         PARITY switch to NONE

   e. Verify that you have the correct system modem telephone
      number and terminate the call to the system operator.

   f. Using the remote modem RTC telephone, lift the handset
      off the hook, pull up the exclusion key, and dial the re-
      mote modem telephone number. The system modem will auto-
      answer and generate a high pitched tone.   When the tone
      is received replace the handset on the hook.   The remote
      link is now established and both the remote terminal and
      system consoles have system console capability.

The following CMP micro diagnostics may now be executed:

   SELFTEST - Refer to the CMP/System Selftest manual
             (p/n 30090-90005).

   IOMAP   - Refer to Appendix B of this manual.

   LOG   - Refer to Appendix A of this manual.

   MAINTENANCE DISPLAY - Refer to CMP Maintenance Mode manual
                        (p/n 30090-90007).

Communication between remote terminal and system console oper-
ators can be established following a CMP prompt (->) by preceed-
ing the message with a question mark (?) as shown in the example
below.

   ->?THIS IS A SAMPLE MESSAGE   (CR)

Messages sent in this manner are passed between the remote term-
inal and system console only and not transmitted to the system.

If it becomes obvious that the DUS diagnostics must be used, use
the above method to send a message to the system operator asking
them to mount a DUS tape on the cold load device. Refer to the
Diagnostic Manual Set (p/n 30070-60068) or the Customer Engineer
Handbook (p/n 30070-90010) for information about using DUS.

2.2 REMOTE SESSION TERMINATION

To terminate the remote session, use the following procedure:

   a. Contact the System Operator via console message or telephone
      to inform them that the remote session is completed and pass
      on any pertinent information about the system.

   b. Tell the System Operator to set the REMOTE CONSOLE and MAIN-
      TENANCE MODE switches to OFF.

   c. Lift the modem telephone handset off the hook and pull up
      the exclusion key. Hang up the handset and the session is
      now terminated.

CMP Command Descriptions

This appendix contains a listing of the commands accepted by the
CMP operating system and a description of what the commands do.
Maintenance Display commands are not included. Refer to CMP Main-
tenance Mode manual (p/n 30090-90007) for descriptions of Main-
tenance Display commands.

COMMAND   DESCRIPTION

HELP –   This command causes a list of the CMP commands to be
            displayed at the console. If the Maintenance Display
            is presently being used, this command will list the
            Maintenance Display commands also.

HALT –   This command causes the CPU run/halt flip flop to be
            set to halt. This command performs the same function
            as the HALT button on the system control panel.

RUN –   This command causes the CPU run/halt flip flop to be
            set to run. This command performs the same function
            as the RUN button on the system control panel.

DUMP –   This command causes a dump to be performed from the
            device specified in front panel thumbwheel switches.
            If the system is running, the message "IS IT OK TO
            ABORT THE SYSTEM?" will be displayed. The user must
            enter "YES" to continue with the dump.

LOAD –   This command causes a cold load to be performed from
            the device specified by front panel thumbwheel switch
            settings. If the system is running, the message "IS IT
            OK TO ABORT THE SYSTEM?" will be displayed. The user
            must enter "YES" to continue with the load.

START –   This command causes a warm start to be performed from
            the device specified by front panel thumbwheel switch
            settings. If the system is running, the message "IS IT
            OK TO ABORT THE SYSTEM?" will be displayed. The user
            must enter "YES" to continue with the warm start.

SELFTEST – This command initiates a selftest of the system. If
            the system is running, the message "IS IT OK TO ABORT
            THE SYSTEM?" will be displayed. The user must enter
            "YES" to continue with the selftest. Refer to the
            CMP/System Self Test manual (p/n 30090-90005) for more
            information on SELFTEST.

A-1

LOG -   This command causes a log of actions performed or de-
         tected by the CMP to be displayed. Log entries are:

         Display On   'Display' command entered.

         Dump   'Dump' command entered.

         Halt   CPU went to halt.

         Halt Command   The HALT command was used.

         IOMAP   'IOMAP' command entered.

         Load   'LOAD' command entered.

         Power On   Power-on reset to CMP when battery
                                power was lost.

         Powerfail/Reset   A powerfail occurred or the PON
                                signal went low without loss of
                                memory backup power.

         Run   CPU went to run.

         Selftest OK   CMP/System selftest passed.

         Selftest failed   CMP/System selftest failed.

         Shuttest   Power shutdown test performed.

         Shuttest failed   power shutdown test failed.

         Start   'START' command entered.

         These events are listed along with the elapsed time
         (not including power off time) since the event occur-
         red. The last 63 events are shown. The log is re-
         tained during a power failure since it is stored in
         memory which is on battery backup. An example is:

                        EVENT LOG

         DAYS   HRS   MIN   SINCE EVENT
           2   23   05   Power On
                 3   20   Overtemp Shutdown
                 3   20   Powerfail/Reset
                 2   25   Halt
                 2   20   Start Failed
                 2   05   Halt
                 2   03   Load Failed
                 1   51   Halt
                 1   50   Cold Load
                 1   50   Selftest Failed
                       03   Load

DISPLAY –   This command causes the Maintenance Display to appear
            on the screen.   Refer to the CMP Maintenance Mode man-
            ual (p/n 30090-90007) for more information.

SPEED –   This command allows the user to change baud rates when
            MPE is not running.   The receive and transmit baud
            rates must be specified in the parameters used with
            the command.   The receive and transmit baud rates must
            be compatible with MPE and the ADCC.   This command is
            identical to the MPE :SPEED command.

                    COMMAND SYNTAX:

                    SPEED (send baud rate) (receive baud rate)

:SPEED –   This command is identical to the standard SPEED com-
            mand but it allows the CMP and MPE to be set to the
            same speed simultaneously.   Use this command when MPE
            is running.

                    COMMAND SYNTAX:

                    :SPEED (send baud rate) (receive baud rate)

IOMAP –   This command causes the current system configuration
            to be displayed.   The memory size, control panel
            switches, and all channels and devices in the system
            are identified.   Refer to Appendix B for more informa-
            tion about IOMAP.

Invalid commands will cause the message 'INVALID COMMAND, USE
HELP' to be displayed.

The following two CMP microdiagnostics may only be used at the
computer site and not in remote mode.

SHUTTEST – This command tests the power fail and overtemp shut-
            down circuitry on the CMP and in the power supply.
            The command first causes the message 'Cycle Power to
            Restart System' to be displayed and then pulls PFW-
            low on the backplane.   This should cause all power to
            be shut off in the system except the memory supply.

            When the system shuts down the following conditions
            should be present:

            The Overtemperature LED should be ON.

            The Power LED should be OFF.

            The Memory LED on the back plane should be ON.

To bring the system back up after this test, cycle the main power breaker of the system. When power is cycled, the LOG function should log as its last entry POWERFAIL/AUTORESTART. If the LOG is cleared at this time there may be a battery problem. If the test fails the CMP will display the message 'Test Failed'.

DCTEST – This command performs tests of the RS-232C signals on the CMP. A special test adapter (p/n 30090-60052) must be installed between J3 on the CMP and the standard CMP cable. The CMP cannot be in remote mode while performing this test. Refer to CMP/System Selftest manual (p/n 30090-90005) for more information.

The IOMAP command is used to display the current configuration of the system. The parameter 0 may be used with the IOMAP command (IOMAP 0) to loop IOMAP until the user types a character (any character) at the console. The following is a sample IOMAP display:

```
->IOMAP
I/O CONFIGURATION
1024KB MEMORY
START 11 0
DUMP  11 0
LOAD   9 1
CHL=01     ID#8011   ADCC/EXTENDER
CHL=02     ID#8001   ADCC
CHL=09     ID#0000   GIC
  DEVICE#01    ID#0183    7970E
CHL=10     ID#0000   GIC
  DEVICE#01    ID#2001    2608
CHL=11     ID#0000   GIC
  DEVICE#0     ID#0002    DISC(S)
  DEVICE#01    ID#4002    INP
  DEVICE#03    ID#2003    2617
```

The IOMAP display first shows the memory size of the system in kbytes. The memory size shown is the number of contiguous bytes starting at adress 0.

The START, DUMP, and LOAD channel and device numbers are read from the system control panel and displayed.

Each channel is identified. The channel number is shown in decimal.   An ID of 8001 indicates an ADCC without an extender, and an ID of 8011 indicates an ADCC with an extender. If the channel ID code is neither an ADCC or a GIC, then ??? is displayed.

All devices on all GICs are identified. If the ID code of the device does not match any device known to the CMP, ??? is displayed instead of the device type.

If errors are detected during IOMAP, error messages in inverse video will be displayed. Refer to CMP/System Self Test manual, Appendix A, for error message descriptions.

HP 3000 Computer System

7902\9895 FLEXIBLE DISC UNIT (FDU)
DIAGNOSTIC MANUAL

LIST OF EFFECTIVE PAGES

The List of Effective Pages gives the date of the current edition and of any pages changed in updates to that edition. Within the manual, any page changed since the last edition is indicated by printing the date the changes were made on the bottom of the page. Changes are marked with a vertical bar in the margin. If an update is incorporated when an edition is reprinted, these bars are removed but the dates remain.

All pages original issue . . . . Mar 1982

PRINTING HISTORY

First Edition .... Mar 1982

| **CONTENTS** |
---

v

```
+------------------------------------------------+ +---------+
|                                                | | SECTION |
|                             GENERAL INFORMATION| |---------|
|                                                | |    I    |
+------------------------------------------------+ +---------+
```

1.0 INTRODUCTION

This document specifies the operation of the 7902A/9895 Flexible
Disc Unit (FDU) Diagnostic.   It performs certain tests on the
7902A/9895 Flexible Disc Subsystem, to verify correct operation,
in an HP 3000 HP-IB version system.   It is written in the AID
Diagnostic Language. Thoughout this manual all references to the
7902 are applicable to the 9895 Flexible Disc Unit.

1.1 Required Hardware

Hardware required to run the Diagnostic Utility System. (DUS)

1.2 Required Software

Diagnostic/Utility disc

1.3 Messages and Prompts

Information and error messages are displayed by the diagnostic:
Information messages are used to inform the operator of the pro-
gress of the diagnostic or to request some action be taken by the
operator. Error messages indicate when some response to a test
was not normal. Error messages are also emitted from pre-defined
functions when an unusual condition is detected.

i.e.,"Error in step XXX: Current operation is < read >"
     "Current disc status is (1)= !XXXX (2)= !YYYY"

Indicates that after a read command was sent to the drive the
subsequent   status check detected an error condition. (See
Pre-defined function section for further information.)

   "Channel program timeout in section X step Y (Chan=a Dev=b)"
   "Diagnostic aborted" -or-

   "Channel program malfunction in section X step Y (Chan=a
   Dev=b)" "Diagnostic aborted"

Indicates that an error occurred within a channel program. The
timeout message indicates that commands sent to the device were
not acknowleged in a given amount of time. The malfunction mes-
sage indicates that some other condition caused the error. i.e.,
channel hardware failure. (Make sure you entered the correct DE-
VICE ADDRESS of the 7902A)

If desired, the abort can be avoided by using the appropriate GO parameter. (See GOPARAMETERS)

A message indicating the completion of each test section is selectable by using GO parameters. (See GOPARAMETER section)

   "Completed section X"

X= test section just executed

1.4 Philosophy

The Diagnostic relies mostly on the 7902A Self-Test to find faults in the subsystem. Other test sections are provided to test parts of the system not checked by Self-Test:

a. response to commands sent over the HP-IB.

b. integrity of data transmitted and received from the 7902A.

c. Detection of changes in drive status.

The operator can run this diagnostic in the standard mode, which requires no operator interaction, or the operator may select specific tests that may or may not require operator intervention.

The standard tests (tests 1,2,3,4) check the hardware on the controller. and perform a Read test.

In addition to the built-in tests, the operator may create custom tests by using the SECTION feature of AID in conjunction with the pre-defined functions provided in the diagnostic. (See Section 6 for pre-defined function descriptions)

1.5 TEST LIMITATIONS

(1) No media checking is performed.

(2) No large data transfer checking (maximum transfer is 256 bytes).

(3) 'EXIT' after an error or abnormal diagnostic termination may leave the FDU in a state that will disallow FDU access. In this case, recoldloading of DUS is required.

(4) No checking for address uniqueness with respect to FDU access is done.

```
+-----------------------------------------------------+ +---------+
|                                                     | | SECTION |
|                            OPERATING INSTRUCTIONS   | |---------|
|                                                     | |   II    |
+-----------------------------------------------------+ +---------+
```

2.0 INTRODUCTION

To run the diagnostic follow the procedures for running a diag-
nostic from the Diagnostic Utility System (DUS) Reference manual
and enter "D7902".

2.1 GO Parameters

When entering the 'GO' command, after any pause prompt, two
diagnostic features can be controlled:

Command format: GO [GOPARAM1 [,GOPARAM2]]

If a parameter is omitted, it defaults to zero.

GOPARAM1: If zero or omitted then the 'Section complete' message
          will be suppressed. This reduces number of messages
          emitted when looping.

GOPARAM2: If zero or omitted causes diagnostic to abort if an
          error occurs during the execution of a channel program.
          If not zero then when a channel program error occurs, a
          message will be displayed and the diagnostic will con-
          tinue.

    i.e., GO 0,1   (GOPARAM1=0, GOPARAM2=1)
          GO ,1   (Same as above)

2.2 MESSAGE OUTPUT AND ERROR PAUSE CONTROL

The following commands may be employed before entering the GO
command to either suppress or enable error or non-error messages.

    *EEPR   - enable error messages.
     SEPR   - suppress error messages

    *ENPR   - enable non-error messages.
     SNPR   - suppress non-error messages.

    *EEPS   - enable pauses after error messages
     SEPS   - suppress pauses after error messages
     RST   - reset print and pause commands to 'enable'.
              (Equivalent of EEPR + EEPS)

    *Default Value

2.3 TEST SELECTION

Several different tests make up the 7902 Flexible Disc Diagnostic
program. You may select one, more than one, or all of these
tests by using the TEST command. The format for the test command
is as follows:

TEST   - change from the default set of section execution

'TEST 1,5,8' -- execute sections 1,5 and 8.
'TEST 1/3,8' -- execute sections 1,2,3, and 8.
'TEST +3,6' -- add sections 3 and 6.
'TEST -3,6' -- remove sections 3 and 6.

Note that Sections 14 thorugh 17 are not executed as part of the
standard execution set, but must be requested by the operator via
the TEST command.

2.4 PROGRAM EXECUTION CONTROL

Use of one of the following commands actually causes the 7902
diagnostic program to execute:

GO   - continue Diagnostic execution from a pause.
GO,1   - suppresses section complete message.
GO,2   - aborts diagnostic when channel program failure is
            detected.
RUN   - restart execution of diagnostic at the beginning.

Use of the following command will cause the 7902 to return con-
troll to the Diagnostic/Utility system file manager.

EXIT and RETURN

```
+-----------------------------------------------------+ +----------+
|                                                     | | SECTION  |
|                                  EXECUTION TIMES  | |----------|
|                                                     | |  III  ·  |
+-----------------------------------------------------+ +----------+
```

3.0 INTRODUCTION

The following information describes the test sections that com-
prise the diagnostic.

```
+---------------------------------------------------------------+
| ------------------------------------------------------------- |
|| Def |Test    Test    |   Name             |Approx   |Notes|  |
||     |Sect.   Steps   |                    |Run      |     |  |
||     |                |                    |Time/Pass|     |  |
||-------------------------------------------------------------||
||  *  | 1      101-102  | Identify           |   ms    |     |  |
||  *  | 2          201  | HP-IB Loopback     |  3 sec  |     |  |
||  *  | 3      301-302  | Basic Self-Test    |  6 sec  |     |  |
||  *  | 4      401-402  | Read-Only Self-Test|  6 sec  |     |  |
||  1  | 5      501-503  | Read/Write Self-Test|7sec/cyl|  1  |  |
||     | 6      601-603  | Status Change      |operator |     |  |
|  ------------------------------------------------------------  |
|                                                               |
|                                                               |
|  Total run time per pass:                                     |
|    no writing              15 sec.                            |
|    write one cylinder      22 sec.                            |
|    write all cylinders      8 min                             |
|                                                               |
|  Notes:                                                       |
|    *   part of default set of sections                        |
|    1   writes on flexible disc if operator answers 'YES' to   |
|        query                                                  |
+---------------------------------------------------------------+
```

Figure 3.1 - Table of Sections

```
+-----------------------------------------------------+ +---------+
|                                                     | | SECTION |
|                                   TEST DESCRIPTIONS | |---------|
|                                                     | |   IV    |
+-----------------------------------------------------+ +---------+
```

4.0 INTRODUCTION

The diagnostic consists of three parts:

  *Control section
  *Test sections 1-6
  *Operator created test sections (7-48)

The following gives a detailed description of each area.

4.1 CONTROL SECTION

This section initializes and controls execution of the diag-
nostic. The title and associated messages are displayed. (See
below) Channel and device addresses are requested from the oper-
ator. If specific test sections are not selected then the
standard set is selected. (Tests 1,2,3).

Typical scenario starts with the 7902 diagnostic title message
and continues as shown below:

"DIAGNOSTIC: 7902A Flexible Disc Unit (FDU) Revision XX.XX"

"Type 'GO' to continue ('LC' to list commands)."
>TEST 2,4 (TEST 2,4 entered by operator selecting tests 2 and 4)
>GO   (GO entered by operator)

Once GO is entered the following 2 questions are asked to com-
plete the information needed by the 7902 diagnostic program.

"What is the CHANNEL ADDRESS of the controlling GIC (1-15) ?" 2

"What is the DEVICE ADDRESS of the FDU (0-7) ?"6

Responses to the channel address question are checked for correct
range and if valid, the channel is tested for being a GIC. A
message will be displayed if channel is not a GIC or if the num-
ber entered was for a non existent channel. The request will be
repeated in the event of an error.

Response to the device address request is checked for range only.
If out of range the request is repeated.

The diagnostic proceeds to execute after the channel and device
numbers have been input and outputs the following message when
complete and looping has not been specified.

"End of pass 1"
Control now returns to DUS if not in 'LOOP' mode.

4.2 TEST SECTION 1 - DEVICE IDENTIFY

STEP 101 - Checks if the diagnostic can communicate with the drive over the HP-IB. (Attempts the IDENTIFY command. If no response occurs then the diagnostic assumes that the 7902A cannot communicate over the HP-IB) and a message is displayed:

"7902A Subsystem does not respond"

STEP 102 - Uses the returned value of the previous step, if successful, and verifies the correct ID code was returned.

"Identify returned !XX expected !81"

!XX=returned hex ID code.

4.3 TEST SECTION 2 - HP-IB LOOPBACK TEST

This section performs the HP-IB loopback test on the 7902A. No writing on the disc occurs in this section. The HP-IB path is tested and the ability of the 7902A controller to accept and transmit data.

STEP 201 - Transmits and receives 256 bytes of data in loopback mode. Verifies data received is the same as the data transmitted. (Pattern is binary progression: !00,!01 !02,!03,...!FF)

"Loopback returned X bytes. Expected 256"

"Loopback data byte X was !YY and should be !ZZ"

STEP 202 - Same as step 202 except data pattern is alternate ones and zeroes. (!00,!FF,!00,!FF...etc.)

4.4 TEST SECTION 3 – BASIC SELF-TEST

This section runs the 7902A Self-Test. No writing or reading will occur. (Performs basic controller tests)

STEP 301 - Runs the 7902A Self-Test and checks results.

"Self-Test result was !XXXX expected !0000"

!XXXX is the returned hex result.

STEP 302 - Checks the DSJ value to insure it is '2' after Self-Test completes.

"DSJ of 2 expected after Self-Test. Received xxx"

A CLEAR is issued after the test completes.

4.5 TEST SECTION 4 – READ–ONLY SELF-TEST

This section runs the 7902A Self-Test. Reading from the flexible disc will occur. Therefore a formatted flexible disc is required to to be in the drive. No writing will occur in this section.

STEP 401 - The drive status is checked to insure that a flexible disc is in the drive and the drive is ready.

"FDU not ready, unreadable flexible disc or drive empty"

"Current disc status is: (1)= !XXXX (2)= !YYYY"

(see Section 7 for breakdown of status words one and two)

STEP 402 - Self-Test is invoked and the result checked.

"Self-Test result was !XXXX expected !0000"

STEP 403 - DSJ value is verified to be '2' after Self-Test completes.

"DSJ of 2 expected after Self-Test. Received xxx"

A CLEAR command is issued after this section completes.

4.6 TEST SECTION 5 - READ/WRITE SELF-TEST

This section runs the 7902A Self-Test. Writing on the disc will occur. The operator is asked if it is okay to write on flexible disc currently in the drive.

STEP 501 - Displayed:

"Insert formatted scratch flexible disc, then type 'GO'"

"*** TEST 5 MAY CAUSE LOSS OF DISC DATA ***"

Insure that the flexible disc is OK to write on then type 'GO'.

"Diskette is write protected."

The above message will not be output if diskette is not write protected. If this message does appear, type 'GO' to continue.

"Note: next step requires about 7 seconds per cylinder."
"What cylinder should be used for this test (0-75 or 'ALL') ?"

Operator response required. If operator enters a number out of range, a message is displayed and the request is repeated.

"Cylinder number out of range"

A check is made as to drive status. If not okay then:

"FDU not ready or no flexible disc in drive"
"Current disc status is (1)= !XXXX (2)= !YYYY"

STEP 503 - Runs the 7902A Self-Test. The results are checked:

"Self-Test result was !XXXX expected !0000"

If 'ALL' was entered in response to the cylinder request then:

"Self-Test result was !XXXX expected !0000 Cyl=Y"

A CLEAR is issued after the test completes.

If the 'LOOP'command is being used, the questions will be asked only on the first pass. Subsequent passes use the same cylinder or all cylinders if 'ALL' was entered for the requested cylinder in pass 1.

'ALL' also will cause the numbers 0 through 75 to be displayed as each cylinder is checked.

4.7 TEST SECTION 6 – STATUS CHANGE DETECT

This section tests the ability of the 7902A controller to detect and report changes in drive status. The operator opening the door is used as an external stimulus for this test. The drive is put into a 'polling' state. (Parallel poll response to the HP-IB is disabled. The controller polls the drive until a status change is detected, then asserts parallel poll on the HP-IB to indicate the change)

A flexible disc must be in the drive and the door must be closed. The operator is reminded by a message:

"Be sure that a flexible disc is in the drive and that the door is closed." "Then type 'GO' to continue"

STEP 601 - Requests status to clear any error indications and to insure the drive is ready.

"STATUS1=!xxxx   STATUS2=!yyyy"

STEP 602 - Issues an 'END' command to the drive to put it in the polling state.

STEP 603 - Operator is requested to open the drive door.

"Open the door of the FDU. (You have about 15 secs. to respond)"

If the test fails then:

"Status change test. FDU did not assert parallel poll"

If the test passes:

"Close the FDU door and type 'GO' to continue"
":>"

A CLEAR is issued after the test completes.

4.8 PRE-DEFINED FUNCTIONS

This section describes the pre-defined functions availble to the operator to create custom test sections.

Several of the functions check drive status during execution and if an error occured a message is displayed and the program will pause. The message gives information about the function executing when the error occured.

Note: Any function mnemonics followed by '*' contain the status
      check feature.

i.e., "Error in step YYY: Current operation is < seek >"
      "Current disc status is: (1)= !XXXX (2)= !ZZZZ"

Where XXXX and ZZZZ are the returned status words one and two
respectivly. YYY is whatever step number set by the operator. The
error message and pause can be selected using the STATON and
STATOFF commands. (See further explanation in this section)

4.80 Example of Operator-Created Test Section

The first statement entered by the operator MUST replace the last
statement in the diagnostic and the last statement entered by the
operator should be a "GOTO" the second statement of the diagnos-
tic. This will insure compatibiltiy with the standard diagnostic
test sections.

                            CAUTION

      The first three statements in the diagnostic should not
      be changed in any way. They are vital to the correct
                  execution of the diagnostic.

Operator enters the following program:

Note: line "7000" was picked only for demostration. The last
      line in the diagnostic may be different

```
>7000 SECTION 7,7140            .DEFINES THE TEST SECTION
>7010 LET STEP:=700             .DEFINE USER STEP NUMBER
>7020 DB AA,128,0               .CREATE BUFFER 128 WORDS
>7030 DB BB,128,%70707          .CREATE BUFFER 128 WORDS OF %70707
>7035 STATON                    .ENABLE STATUS CHECKS
>7040 RANDCYL A                 .PICK A RANDOM CYLINDER
>7050 RANDSEC B                 .PICK A RANDOM SECTOR
>7060 SEEK A,0,B                .SEEK CYL/HEAD/SEC
>7070 WRITE BB(0)               .UNBUFFERED WRITE, 128 WORDS FROM
                                 'BB'
>7080 SEEK A,0,B                .SEEK TO SAME CYL/HEAD SEC
>7090 READ AA(0)                .UNBUFFERED READ,128 WORDS INTO
                                 'AA'
>7095 DISPADDR                  .SHOW DISC ADDRESS AFTER READ
>7100 CB AA(0),BB(0),128        .COMPARE READ/WRITE BUFFERS
>7110 IF INDEX = -1 THEN 7130   .SKIP IF NO ERROR
>7120 EPRINT"WORD ";INDEX;" WAS ";%AA(INDEX);" SENT ";%BB(INDEX)
>7130 DISCCLR                   .CLEAR THE DISC AFTER TEST
>7140 GOTO 20                   .BRANCH TO SECOND STATEMENT
```

    *** End of operator program. now ready to run ***

The display when you now enter RUN looks like this:

>RUN

DIAGNOSTIC: 7902A Flexible Disc Unit (FDU) Revision XX.XX
Enter test selection(s) and/or 'GO' to continue.

> TEST 7 *** selects the custom section just added ***
LOOP
>GO

What is the CHANNEL ADDRESS of the controlling GIC (0-15) ? 2

What is the DEVICE ADDRESS of the 7902A (0-7) ? 6

End of pass 1

  *** Operator-created error message including step number ***

Error in step 700:WORD 14 WAS %70706 SENT %70707
>GO

End of pass 2

Program continues until interrupted. It is recommended that step
numbers be used in custom test sections and that they start with
the section number as the hundreds digit.

i.e., Section 7 steps should be from 700 to 799.
      Section X steps should be from X00 to X99.

This will maintain diagnostic conformity as this is how the steps
are numbered in the diagnostic test sections. Also any error
messages that are emitted from custom test sections should use
the "EPRINT" statement. This will cause the step number to be
included in the message. (See example program above)

4.81 Functions Defined

 SEEK [cylinder [,head [,sector]]] *

    Causes the drive to seek to cyl/head/sector. If a parameter
    is omitted, it is assumed to be zero. The range of parameters
    is not tested by the program. It is up to the operator to
    determine the correct values.

 READ buffer(element) [,bytecount [,1/0 [,1]]   *
                                    +   +
                                    | |
      Buffered/Unbuffered--------- |
      Read with verify (reduced margins)

    Read disc data into buffer(element) for bytecount.   If the
    byte count is 0 or omitted then 256 bytes will be read. (Buf-
    fered read uses burst mode transfers with burst length=256

bytes. Multiple bursts are used when reading more than 256 than 256 bytes)

WRITE buffer(element) [,bytecount [,1]]   *
                                          +
                                          |
        Buffered if included---------

Same as read except data is written to the disc. (There is no write with verify)

GETSTAT buffer(element)

Reads current disc status into buffer(element) and buffer(element+1)

GETADDR buffer(element) [,1]

Reads current disc address (Logical address) stored in the controller into buffer(element) and buffer(element+1) If the optional "1" is included then the physical location of the head positioner is read.(Physical address) (see appendix A for word breakdown)

FORMAT [0/1 [,databyte]]   *   (See note below)
               +
               |
               -- HP/IBM format

Causes the 7902A to format the flexible disc currently in the drive. HP or IBM formatting is selected by the parameter shown. The databyte if included is written to each sector after it is formatted. If parameter or databyte are omitted they are assumed to be zero.(HP format,write zeroes)

Note: The current version of the diagnostic does not transmit 'databyte' to the 7902A as this feature applies only to the double sided 7902A. Omit this parameter when testing single sided drive.

DISCEND

Sends the END command to the 7902A.

DISCCLR

Sends the SELECTED DEVICE CLEAR command to the 7902A. Also issues GETSTAT command. (Puts disc in known state)

DISCINIT buffer(element) [,bytecount [,1]]

   Performs the 7902A INITIALIZE command. Data is written from
buffer(element) for bytecount. If omitted, bytecount=256. If
the "1" is included the Defective bit is cleared other- wise
it is set.

VERIFY [sectorcount]

   Sends the 7902A it's VERIFY command. Data is verifed for
sectorcount. If sectorcount is omitted or zero then the
entire disc is verified from the current disc address to the
end of the disc. No data is transferred over the HP-IB.
Errors are indicated by the status words which should be
requested and checked after this command completes.
Requesting the DISCADDR will indicate the failing
cyl/head/sec. (reduced margin read is used during the verify)

LOCK

   Sends the LOCK command to the 7902A.
(Lock drive door)

UNLOCK

   Sends the UNLOCK command to the 7902A.
(Unlock drive door)

DISPADDR [1]

   Requests disc address and displays returned values IF the "1"
is included then the Physical address is obtained and
displayed. (see GETADDR)

   "Current disc address is: Cylinder=X Head=Y Sector=Z"

DISPSTAT [1]

   Requests and displays current disc status words. If the "1"
is included then status is not requested but the values in
the reserved buffer elements TT(10) and TT(11) are displayed.

   "Current disc status is: (1)= !XXXX (2)= !YYYY"

 NOTE: If the GETSTAT function is used to actually obtain the
      status words, the buffer element specified in the GETSTAT
      function should be TT(10). Otherwise the "DISPSTAT 1" may
      display erroneous values.

RANDCYL variable -or- buffer(element)

   Generates a random cylinder number between 0 and 75 and re-
   turns it in variable or buffer(element).

RANDSEC variable -or- buffer(element)

   Generates a random sector number from 0 to 29 and returns it
   in variable or buffer(element)

RANDHEAD variable -or- buffer(element)

   Generates a random head number 0 or 1 and returns it in var-
   iable or buffer(element). (Use for doublesided 7902A only)

DISCDSJ variable -or- buffer(element)

   Executes the DSJ instruction and returns the resulting value
   in variable or buffer(element)

LOOPBACK buffer1(element),buffer2(element)

   Sends 256 bytes of data from buffer1(element) to the 7902A
   using loopback mode. Requests the data be sent back and puts
   it in buffer2(element).

STATON

   Enables the status check feature in the READ, WRITE, READFULL
   SEEK, and FORMAT functions. (If status indicates an error
   condition, a message is displayed and a pause occurs)

STATOFF
   Disables status checks in the READ, WRITE, READFULL, SEEK and
   FORMAT functions.

EOSMSG

   If GOPARAM1 is not zero then the following message is dis-
   played:

   "Completed section X"

   X= current test section executing or just executed. (Reflects
   the AID reserved variable 'SECTION')

SELF-TEST C,H,W,R - buffer(element)

    Causes 7902A to invoke its Self-Test. Returns results in
    variable or buffer(element).

    C= (0-75) Cylinder number for write test

    W= (0/1) Write part of Self-Test suppressed/included

    R= (0/1) Read part of Self-Test suppressed/included

    H= (0/1) Head selection (may or may not be selectable, but
             applies to doublesided 7902A only. Use zero
             for single sided).

     Note: The following rules apply to the read/write
     Self-Test.

       (1) If the read part is suppressed then the write part is
           also suppressed.

       (2) If the read part is included but the write part is
           suppressed, then the cylinder number is not used.

       (3) All parameters must be included even if they are not
           used by the 7902A Self-Test.

       i.e., SELF-TEST 0,0,0,1,AA(0)

       Meaning: Invoke   Self-Test,   Cylinder=0   as   dummy
                 parameter, Head=0 as dummy parameter, write
                 suppressed, read included, results into buf-
                 fer(element)=AA(0).

READFULL buffer(element)   [,bytecount [,0/1 [,1]]]   *
                                          +   +
                                          | |
 (Trigger on target sector ID. -1/-2)---- |
                                                  |
   (Include if trying to read first sector)-----
   (of cylinder using this command.   )

    Reads disc data using the 7902A 'READ FULL SECTOR' command.
    Data is read into buffer(element) for bytecount. If byte-
    count is omitted or zero it is set to 256 bytes. If the
    trailing parameters are omitted then they are set to zero.

4.82 Information for Operator-Created Tests

The following variables and buffers are reserved for exclusive use by the diagnostic. Using any of the reserved variables or buffers by an operator created test, except as noted, may cause unusual results.

     RESERVED VARIABLES: None
     RESERVED BUFFERS   : TT,LL,MM,ZZ,&ZZ

     VARIABLES AND BUFFERS USED BY SECTION:

        Control section : A,C,D,TT
        Test section 1 : D,TT
        Test section 2 : A,B,C,D,TT,LL,MM
        Test section 3 : TT
        Test section 4 : TT
        Test section 5 : TT,&ZZ
        Test section 6 : TT

     Note: Any buffers or variables not listed under 'RESERVED'
           are released after exiting the section they are used in.
           They may be used by the operator in custom test sections.
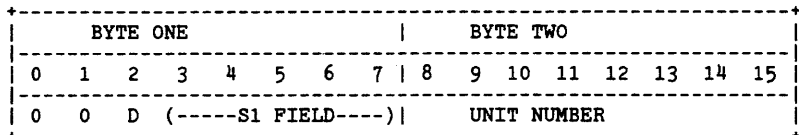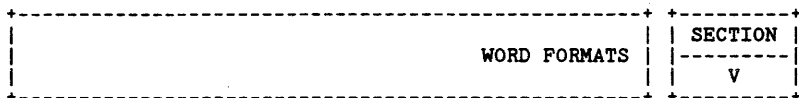
```
+----------------------------------------------------+  +----------+
|                                                    |  | SECTION  |
|                                    WORD FORMATS    |  |----------|
|                                                    |  |    V     |
+----------------------------------------------------+  +----------+


+----------------------------------------------------------------+
|         BYTE ONE                 |        BYTE TWO             |
|----------------------------------------------------------------|
|  0   1   2   3   4   5   6   7 | 8   9  10  11  12  13  14  15 |
|----------------------------------------------------------------|
|  0   0   D  (-----S1 FIELD----)|     UNIT NUMBER               |
+----------------------------------------------------------------+
              +
              |
              Defective bit

  S1 field:          !00.......Normal completion
                     !01.......Illegal opcode
                     !07.......Cylinder compare error
                     !08.......Uncorrectable data error
                     !09.......Sector compare error
                     !0A.......I/O program error
                     !11.......Defective cylinder/sector
                     !12.......Retryable hardware error
                     !13.......Status 2 error (see status word 2)
                     !1F.......Seek complete or drive error occurred

              Figure 5.1 - Status Word One
```

```
+------------------------------------------------------------------+
|       BYTE ONE             |       BYTE TWO                       |
|----------------------------------------------------------------|
| 0   1   2   3   4   5   6   7 | 8   9  10  11  12  13  14  15 |
|----------------------------------------------------------------|
| X   0   0  (--Diskette---)  0 | X   X   0   X   X   X   X   X |
+------------------------------------------------------------------+
    +                              +   +       +   +   +   +   +
    |Diskette: 00-Empty drive      |   |       |   |   |   |   |
    |        : 02-Blank or ??Fmt.  |   |       |   |   |   | Busy
    |        : 04-HP Fmt.          |   |       |   |   |Not Ready
    |        : 10-IBM Fmt.         |   |       |   | Seek Check
    |                              |   |       |   1st Status
    |                              |   |       Drive Fault
    |                              |   Write Protect
    |                              Attention
    on if bits 11 or 13 or 14 or 15 are on.

    * Bits 14-15:  00-Ready
                   01-Never occurs
                   10-No drive connected
                   11-No flexible disc in drive
```

Figure 5.2 - Status Word Two
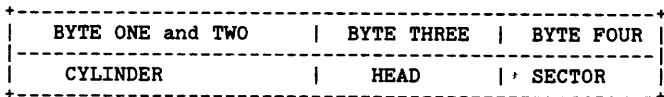
```
+---------------------------------------------------+
|   BYTE ONE and TWO    |  BYTE THREE  | BYTE FOUR  |
|---------------------------------------------------|
|     CYLINDER          |     HEAD     | , SECTOR   |
+---------------------------------------------------+
                                     +
                                     |
                 Zero if physical address-+
```

Figure 5.3 - Disc Address

```
+-------------------------------------------------------------+
| 0   1   2   3   4   5   6   7   8   9   10 |11  12  13  14  15 |
|-------------------------------------------------------------|
|                                           | Displayed on LEDs |
|-------------------------------------------------------------|
|           Returned from Drive to Diagnostic                 |
+-------------------------------------------------------------+
```
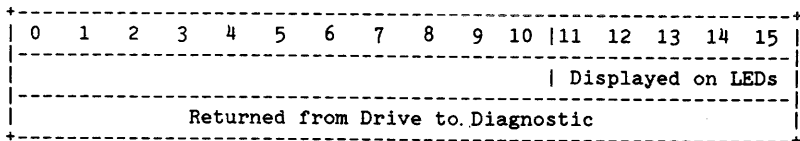
Figure 5.4 - Self-Test Results

```
+------------------------------------------------------+ +---------+
|                                                      | | SECTION |
|                             ERROR INTERPRETATION   | |---------|
|                                                      | |   VI    |
+------------------------------------------------------+ +---------+
```

6.0 INTRODUCTION

The table on the following page relates functions and blocks
tested (e.g. channel program execution; PHI) to specific tests
(Sections in the Diagnostic) and subsystems (GIC, IMB, CPU,
Memory).
There are two ways to use the table:

(1) Enter the table at the failing section - this tells you:

    a. which function or block was under test

    b. the most likely failing subsystems in order of probability

(2) Enter the table at a block or function -- this tells you:

    a. which test section explicity tests that block or function

    b. which test sections depend on that block or function in
       order to pass

    c. which subsystems cause failure of that block or function

If every block or function were independent of the others and
could be tested independently, then the table would have only a
diagonal line of 'X's. Since this is not the case, the table
provides a compact guide to these interrelationships -- to deal
with the cases in which the subsystem appears to fail due to the
failure of other subsystems.

Block or Function Tested	Test Sections 1 v 2 3 4 5 6	Subsystems in order of failure probability
Communication with Controller	X - - - -	PHI, MCC, CTRL, GIC
Controller Operation	X - - -	CTRL, Drive Board, GIC
Reading on one track	X -	CTRL, Drive Board, Mech.
Writing and reading on tracks	X	CTRL, Drive Board, Mech
Drive Status	- - X	Door Switch, Drive Board, CTRL

Figure 6.1 - Function/Block Reference Table

HP 3000 Computer System

MAGNETIC TAPE
DIAGNOSTIC MANUAL

The List of Effective Pages gives the date of the current edition and of any pages changed in updates to that edition. Within the manual, any page changed since the last edition is indicated by printing the date the changes were made on the bottom of the page. Changes are marked with a vertical bar in the margin. If an update is incorporated when an edition is reprinted, these bars are removed but the dates remain.

All pages original issue . . . . Mar 1982

PRINTING HISTORY

First Edition . . . . Mar 1982

```
------------------------------------------------------------------
|                                                     CONTENTS |
------------------------------------------------------------------
```

SECTION I - GENERAL INFORMATION

SECTION II - OPERATING INSTRUCTIONS

SECTION III - TEST SECTION EXECUTION TIMES

SECTION IV - TEST SECTION DESCRIPTIONS

SECTION V - ERROR/ACTION SUMMARY

```
----------------------------------------------------------------
| TABLES                                                        |
----------------------------------------------------------------
```

Title   Page

```
+-----------------------------------------------------+ +---------+
|                                                     | | SECTION |
|                           GENERAL INFORMATION       | |---------|
|                                                     | |    I    |
+-----------------------------------------------------+ +---------+
```

1.0 INTRODUCTION

The Aid Stand-Alone Mag Tape diagnostic verifies the forward and backward data transfer and control functions of the magnetic tape when installed in an HP 3000 HP-IB computer system. The diagnostic is used to detect and isolate tape, drive, interface and controller failures in a 7970E Mag Tape unit or a 7971A which consists of a 7970E Master and a 7970E Slave in one cabinet.

The test method consists of executing one or more instructions to provide stimulus to the circuitry under test. This involves the use of many CPU base set instructions as well as Channel Program Processing instructions.

Errors take the form of messages that are output to the console at the time the error is detected.

| WARNING |

Before running this diagnostic, be sure that the logical configuration of the system (set up for MPE) matches the physical configuration as established by the thumbwheel settings for each channel and device) To obtain a listing of the logical configuration, perform a COLDSTART of the MPE operating system and answer "yes" to the first 3 questions. Then compare the listing numbers with the actual device settings. If they do not match, adjust the thumbwheel settings of the channel(s) or device(s) in error.

1.1 REQUIRED HARDWARE

This diagnostic requires that the console and the "cold load" facility of the system be operational. It is not possible to execute the Magnetic Tape diagnostic program if you are unable to perform a "cold load".

The following hardware is also required:

o An HP 3000 HB-IB version system with at least 128K bytes of memory.

o The Mag Tape unit under test and its controller.

o A Disc or Tape Unit (cold load device to load program).

1.2 REQUIRED SOFTWARE

The only software required to execute this diagnostic is the Diagnostic Utility program and the Magnetic Tape Diagnostic program properly recorded on a flexible disc. or tape unit.

1.3 TESTING PHILOSOPHY

This diagnostic tests all control functions and data transfers. Should an error occur, the diagnostic points to its location and outputs an error message that includes the step number in which the error occured. Testing to the functional level is limited so that errors can be isolated only to the PCB in most instances.

1.4 DIAGNOSTIC LIMITATIONS

The Single Track error can not be simulated for the Mag Tape models. Because of this, bit 3 of status byte 1 will be tested together with bit 6 (Multi-Track Error).

1.5 MESSAGES AND PROMPTS

Three types of messages are output by the diagnostic: Fatal Errors, Non Fatal Errors and Information Messages.

   a) Fatal Errors - would probably cause an operating system to irrecoverably fail or have to be restarted. Example of such errors are: No handshake, waiting time exceeded for a response, Read Status not obtained, etc.

   b) Non Fatal Errors - Soft errors that don't destroy the program and the operator can continue to execute.

   c) Information Messages - are used to inform the operator of the progress of the diagnostic and to instruct the operator to perform an operation.

NOTE

After a message, if an associated pause occurs (indicated by the prompt characater ">" on the console), the operator must input "GO" to continue execution of the diagnostic or any other applicable command.

1.6 MINI-OPERATING INSTRUCTIONS

```
+---------------------------------------------------------------------+
|1.  Turn on the power to all necessary devices.  Magnetic tape |
|    units not to be tested must be turned off.                 |
|                                                               |
|2.  Insert the diagnostic flexible disc or mount the Diag Tape |
|    and enable the unit.                                       |
|                                                               |
|3.  Select the channel and device number of the 7902 Disc or  |
|    Mag Tape and perform the "cold load" procedure.            |
|                                                               |
|4.  Select the channel and device number of the console and   |
|    press the RUN button.                                      |
|5.  The system outputs the following messages:                 |
|                                                               |
|    DIAGNOSTIC-UTILITY SYSTEM REV=xx.xx                        |
|    ENTER YOUR PROGRAM NAME                                    |
|                                                               |
|    Enter either:                                              |
|                                                               |
|         D7970S13 (for a basic check of the drive)            |
|         D7970S45  (for a random read/write verification)     |
|         D7970S68 (for extended interactive diagnostics)      |
|                                                               |
|6.  Enter "GO" in response to the prompt (>).                  |
|                                                               |
|7.  Respond to following instruction messages appropriately.   |
|                                                               |
|    7970E CHANNEL NUMBER?                                      |
|                                                               |
|    ENTER THE IMB NUMBER FOR CHANNEL #(as subsequently entered)|
|    ? (asked only if multiple IMB system)                     |
|                                                               |
|    7970E DEVICE # ?                                           |
|                                                               |
|    MOUNT A TAPE WITH A WRITE RING ON EACH UNIT TO BE TESTED.  |
|    SET OTHERS OFFLINE.                                        |
|                                                               |
|8.  Respond GO.                                                |
+---------------------------------------------------------------------+
```

```
+----------------------------------------------------+ +---------+
|                                                    | | SECTION |
|                          OPERATING INSTRUCTIONS    | |---------|
|                                                    | |   II    |
+----------------------------------------------------+ +---------+
```

2.0 INTRODUCTION

There are two ways to execute the diagnostic program; the Basic way usually includes the loading and execution of D7970S13 and D7970S45. However, D7970S68 may also be executed using the procedure in paragraph 2.1, but keep in mind that it is interactive and requires more time to execute.

Including the extended features of the Magnetic Tape diagnostic is the second way and requires that the extended features be specified before you invoke execution of the diagnostic (i.e. entering "GO").

To execute in either mode, the first 8 steps of paragraph 2.1 must be executed.

|WARNING|

> After, running the diagnostic and repairing any problems, be sure that the logical configuration of the system (set up for MPE at configuration time) matches the physical configuration as established by the thumbwheel settings for each channel and device.

2.1 BASIC OPERATING MODE

In this operating mode, the diagnostic program loops through sections 1-3, 4, 5, and 6-8, respectively, for D7970S13, D7970S45, and D7970S68 the number of times specified, pausing at the end of section 3, 5, and 8 for every loop pass. At the end of each loop you are given the opportunity to 1) continue execution, 2) end the execution of the diagnostic program, or 3) invoke the extended features of the Mag Tape diagnostic (see paragraph 2.2).

1. Turn on power to all devices required to run the diagnostic (see Section 1, Hardware Requirements). Select the Mag Tape units to be tested and turn off all other Mag Tape units.

2. Place DUS disc or tape in the COLD LOAD device and enable the unit.

3. To "cold load" the diagnostics, set the COLD LOAD thumbwheels
    to the channel/device number of the 7902A Disc Unit or **Tape**
    unit.

4. Press the HALT button.

5. Press the SYS RESET button.

6. Press the COLD LOAD button. The cold load procedure should
    begin, then the RUN button lights.

    At this time, the Diagnostic Utility program will have been
    loaded and begun execution by displaying its title message
    and the prompt character (:).

8. Enter "D7970S13, D7970S45, or D7970S68 at the system console.
    The Magnetci Tape diagnostic title messate and prompt charac-
    ter will be displayed (>).

9. Enter "GO" at the system console. The diagnostic program
    responds by outputting the following instruction messages:

    ENTER CHANNEL NUMBER TO WHICH THE 7970E CONTROLLER IS
    CONNECTED (1-15)
    ?
    ENTER DEVICE NUMBER ASSIGNED TO CONTROLLER BY HP-IB (0-7)
    ?

10. Enter the channel number of the GIC to which the Mag Tape
    controller is connected (the controller of the Mag Tape unit
    to be tested). Next enter the device number of the Mag Tape
    controller that controls the unit under test.

11. Once the diagnostic knows which configuration to test, the
    following requests are made:

    ENTER THE NUMBER OF REQUIRED PASSES
    ?
    PUT MAGNETIC TAPE WITH WRITE RING ONTO UNITS UNDER TEST AND
    PUSH ON-LINE AND SELECT UNIT #'S! ENTER GO TO CONTINUE
    ?
    ENTER "GO" TO CONTINUE

    The system responds with:

    UNIT 0 WILL BE TESTED. IF OK TYPE 0, ELSE 1?

    As soon as you input your answer to the above question, the
    diagnostic will   begin execution and continue execution
    through the last selected section in the program or until an
    error condition is detected.   If no errors are detected and
    the last section is completed, the system pauses. The system
    will also pause after an error occurs.

12. Enter "GO" if you want to execute the basic diagnostic
    sections again, or enter AB (abort) if you want to exit from
    the execution of the diagnostic program. For the execution
    of the extended diagnostic features, see paragraph 2.2.

The basic operating mode has the following constraints:

o Sections of programs D7970S13 and D7970S45 are normally
  executed and looped on the number of times specified.
  Sections 6 through 8 are interactive and not normally
  included in the Basic Operating Mode.

o Error, Information, and Prompt messages are displayed.

o Pauses on errors and prompts are not suppressed.

2.2 EXTENDED OPERATING MODES

The extended operating mode includes the following diagnostic features:

o The ability to select particular test sections for execution.

o The ability to suppress or enable error printout and pauses, and suppress or enable non-error printout and pauses.

o The ability to invoke additional diagnostic data accumulation.

o The ability to read variables, arrays, and reserved variables (see AID manual for more details). CPVA table is in array KK, programs in HH, RR or QQ, and status in SS arrays.

To specify the optional diagnostic features mentioned above, perform the following procedure:

1. Execute steps 1 through 8 of the basic operating mode.

2. When the system pauses and the prompt character (›) is displayed, enter one or more of the setup commands shown in table 2-1 in the appropriate format given.

3. Once you have specified the optional diagnostic features you desire, enter "GO" via the system console.

Table 2-1 and 2-2 describe two (2) types of commands: setup commands and control commands. Setup commands can be entered before the diagnostic program is to be executed or during a program pause and are not actually acted upon until the diagnostic program is running.

Control commands can be entered whenever the system pauses, and, they are acted upon immediately (i.e., entering "RUN" causes the diagnostic to begin execution from the beginning).

NOTE

Optional diagnostic features may be specified only at the beginning of the diagnostic execution.

Table 2-1. 7970E Diagnostic Setup Commands

```
|------------------------------------------------------------------|
|                        SETUP COMMANDS                            |
|------------------------------------------------------------------|
| COMMAND FORMAT:  EEPR cr         (where cr means to press the    |
|                                   RETURN key on 3000/33 systems  |
|          (default)                or the ENTER key on HP300's)   |
|                                                                  |
| DESCRIPTION:  The execution of this command lets the program     |
| display error messages as they occur.  The displaying of        |
| error messages in normally enabled when the diagnostic is       |
| initiated for execution.  Therefore, this command would         |
| normally be executed after the SEPR command (suppress error     |
| messages).                                                       |
|------------------------------------------------------------------|
| COMMAND FORMAT:  EEPS cr          (default)                      |
|                                                                  |
| DESCRIPTION:  The execution of this command lets the program     |
| generate pauses after an error occurs.  Pausing after error     |
| messages are output is invoked when the diagnostic is first     |
| executed.  Therefore, the execution of this command would       |
| normally take place after the SEPS command is executed.          |
|------------------------------------------------------------------|
| COMMAND FORMAT:  ENPR cr          (default)                      |
|                                                                  |
| DESCRIPTION:  The execution of this command lets the program     |
| display non-error messages as they occur.  When the program     |
| is first initiated, this condition automatically exists.         |
| Therefore, the execution of this command would normally take    |
| place after the SNPR command is executed.                        |
|------------------------------------------------------------------|
```

```
|-----------------------------------------------------------------|
| |---------------------------------------------------------------| |
| |                  SETUP COMMANDS (cont'd)                      | |
| |---------------------------------------------------------------| |
| COMMAND FORMAT:  GO 1                                            |
|                                                                 |
| DESCRIPTION:  This command allows you to requests additional    |
| diagnostic data accumulation.  When the command is entered      |
| the system outputs the following requests for information:      |
|                                                                 |
| ENTER THE OPTION NUMBER DESIRED: (1,8,12)                       |
|                                                                 |
| Entering a "1" tells the diagnostic program to list the         |
| channel program currently in control when an error occurs as    |
| well as all additional information needed to analyze the        |
| existing problem.  Entering a "-1" clears this request.         |
|                                                                 |
| Entering an "8" tells the diagnostic program to eliminate       |
| all blank lines between messages displayed and to display       |
| the pass message if an error occurs during a pass in which      |
| non-error messages are suppressed.  Entering a "-8" clears      |
| this request.                                                   |
|                                                                 |
| To request a listing of the timing messages in test Sections    |
| 3 and 4, enter a "12".  Enter a "-12" to list only the Error    |
| timing messages.                                                |
| |---------------------------------------------------------------| |
| COMMAND FORMAT:  LOOP cr                                         |
|                                                                 |
| DESCRIPTION:  This command tells the diagnostic to loop on      |
| all test sections to be executed (as indicated by the TEST      |
| command).  To exit a loop execution perform the following       |
| steps:                                                          |
|                                                                 |
| 1.  Enter CNTL Y for HP 3000/33 or ATTENTION for HP300          |
| 2.  When the diagnostic program pauses, enter "ABORT cr"        |
| |---------------------------------------------------------------| |
| COMMAND FORMAT:  SEPR cr                                         |
|                                                                 |
| DESCRIPTION:  The execution of this command lets the program    |
| suppress error messages and error pauses until an EEPR or       |
| RST command is acknowledged.                                    |
| |---------------------------------------------------------------| |
|-----------------------------------------------------------------|
```

```
|------------------------------------------------------------|
|                    SETUP COMMANDS (cont'd)                 |
|------------------------------------------------------------|
| COMMAND FORMAT:  SEPS cr                                    |
|                                                            |
| DESCRIPTION:  The execution of this command lets the program|
| suppress error pauses only until an EEPS or RST command is  |
| acknowledged.                                              |
|------------------------------------------------------------|
| COMMAND FORMAT:  SNPR cr                                    |
|                                                            |
| DESCRIPTION:  The execution of this command lets the program|
| suppress non-error messages that can appear on the console. |
| The ENPR and RST command will override this command.       |
|------------------------------------------------------------|
| COMMAND FORMAT:   TEST [+ or -][x(/x)(,x)]                  |
|                   TEST ALL                                 |
|                                                            |
| where x = a program section number                         |
|                                                            |
| DESCRIPTION:  The TEST command allows you to select one or  |
| more test sections for execution.  A summary and detailed   |
| description of all test sections can be found in Section 4. |
|                                                            |
| TEST 6,8        Executes only test sections 6 and 8.        |
| TEST 1/3        Executes only test sections 1 through 3 once|
| TEST 1,2        Executes test sections 1 and 2.            |
| TEST + 4        Adds section 4 to the sections being tested |
| TEST - 8        Deletes section 8 from the sections being   |
|                 tested.                                    |
|------------------------------------------------------------|
```

Table 2-2. 7970E Diagnostic Control Commands

```
|-----------------------------------------------------------------|
|                        CONTROL COMMANDS                         |
|-----------------------------------------------------------------|
| COMMAND FORMAT:  EXIT cr                                         |
|                                                                 |
| DESCRIPTION:  When you complete entering the EXIT command,      |
| control is returned to the Diagnostic Utility System (FMGR)     |
| where the Mag Tape diagnostic program may be initialized        |
| again or, you may take any other appropriate action.           |
|                                                                 |
| The EXIT command can only be entered when the diagnostic        |
| program pauses.  However, you may force an exit via the         |
| following steps:                                                |
|                                                                 |
| 1. Enter CNTL Y for HP 3000/33 or ATTENTION for HP 300.         |
| 2. When the diagnostic program pauses, enter "EXIT cr".         |
|-----------------------------------------------------------------|
| COMMAND FORMAT:  GO                                              |
|                                                                 |
| DESCRIPTION  The GO command can only be entered during a        |
| diagnostic program pause.  When the GO command entry is         |
| complete, the system continues from the point of pause.         |
|-----------------------------------------------------------------|
| COMMAND FORMAT:  RST cr                                          |
|                                                                 |
| DESCRIPTION:  This command resets all execution commands to     |
| their default state as follows:                                 |
|                                                                 |
| Error Pauses are enabled (EEPS command); Error Messages are     |
| are not suppressed (EEPR command); Non-error Messages are       |
| are not suppressed (ENPR command); operator interaction is      |
| enabled.                                                        |
|-----------------------------------------------------------------|
| COMMAND FORMAT:  RUN                                             |
|                                                                 |
| DESCRIPTION:  The "RUN" command, when entered without any       |
| parameters, causes the diagnostic program to begin execution    |
| at the lowest numbered statement regardless of the present      |
| state of execution.                                             |
|-----------------------------------------------------------------|
```

```
+-----------------------------------------------------+ +---------+
|                                                     | | SECTION |
|                 TEST SECTION EXECUTION TIMES        | |---------|
|                                                     | |   III   |
+-----------------------------------------------------+ +---------+
```

3.0 INTRODUCTION

The exact time for each test or for a group of test is determined
by the following contraints:

- o the length of magnetic tape (the diagnostic tests up to EOT)

- o the number of tested units in a pass (multi-unit tests grow
     quadratically).

- o the selection of test sections to be executed

EXAMPLE:

The execution of Sections 1 through 3 with one tape unit that has
only 200 feet of tape can take between 10 and 14 minutes.   Up to
one (1) hour may be expended if four (4) tape units with 600 feet
of tape are tested.

```
+-------------------------------------------------+ +---------+
|                                                 | | SECTION |
|                 TEST SECTION DESCRIPTIONS       | |---------|
|                                                 | |   IV    |
+-------------------------------------------------+ +---------+
```

4.0 INTRODUCTION

This manual section describes the diagnostic program test
sections which are divided into steps that perform a specific
test. The test section descriptions are categorized as follows:

1. Initialization (Diagnostic Configuration)

2. Start

3. Test Sections 1-5 (non-interactive on disketts D7970S13 and
   D7970S45)

4. Test Sections 6-8 (interactive on diskette D7970S68)

Following is a brief summary of all possible test sections you
can select during diagnostic execution.

    SECTION 1 - Non Interactive test of 16 basic commands and their
                  response by the tape subsystem.

    SECTION 2 - Non Interactive test of 3 STATUS bytes obtained for
                  all possible situations during execution.

    SECTION 3 - Non Interactive test checking time limits.

    SECTION 4 - Non Interactive test to check BOT,EOT and Creeping.

    SECTION 5 - Non Interactive test with a large data-block (up to
                  2 K words) transferred (WRITE - READ). The same for
                  the multi-unit test if more then one tape unit is
                  available.

    SECTION 6 - Interactive test to handling a tape reel, and write
                  ring, and all control elements on the control
                  panel.

    SECTION 7 - Interactive; read and write heads are verified when
                  scoping is used to locate the hardware problem.
                  (Requires a large data block - up to 16K words)
                  SECTION 8 - Interactive; a tape is written and read
                  to find how many feet of tape from BOT (loading
                  point) are bad.

4.1. INITIALIZATION (Diagnostic Configuration)

When the diagnostic is first loaded from the diagnostic flexible disk, and begins execution for the first time, it must know the GIC channel number of the Mag Tape controller to which the unit under test is attached. You supply this information via the following request message which is output just after "GO" is entered.

ENTER CHANNEL NUMBER TO WHICH THE MAG TAPE CONTROLLER IS CONNECTED (1-15)

The above request is repeated if the channel number entered is not recognized as a valid channel.

4.2. START

The startup procedure for this diagnostic is not a selectable section and it is always executed whenever the diagnostic is executed for the first time. Variables used in this program are initialized, all extended diagnostic features are setup, and a check is made on how many units are ready to be tested. The following paragraphs describe the steps in the start section.

STEP
NUMBER   DESCRIPTION

1   This step tests the IMB/channel command transfer by
           issuing a ROCL (roll call) command. When the
           resulting word is returned, the diagnostic verifies
           that the appropriate channel number bit sis set to
           "1". If this step fails to get the expected
           response, the following message and requests are
           output to the console.

           NON EXISTENT CHANNEL - CHANNELS PRESENT = xx TRY
           ANOTHER THUMBWHEEL SETTING ON GIC, OR CHECK IMB, OR
           RUN GIC DIAGNOSTIC.

           ENTER CHANNEL NUMBER TO WHICH THE MAG TAPE
           CONTROLLER IS CONNECTED (1-15) You may want to enter
           the same channel number entered previously or
           another channel number, depending upon the cause of
           the error. Possible causes are:

           o   The channel number entered and the actual
               thumbwheel setting are different. Verify your
               channel number entry; it must match the
               thumbwheel setting.

o The channel specified does not respond to the
   ROCL command. Run the GI⌐ diagnostic to verify
   the GIC PCA after you r ve verified that the
   channel number being entered, is indeed a GIC.

2 This step issues an INIT command to clear the
   channel specified.

3 This step verifies that the channel contacted is a
   GIC (in the instance where an ADCC channel number
   might have been inadvertantly entered).

   Register 14 of the GIC channel is read. Bit 0 must
   be reset to 0 (GIC has DMA capability) and bits 12
   through 15 (channel identification) must be reset to
   zero (0). If this step fails, the following
   messages and request are output to the console:

   CHANNEL NUMBER ENTERED IS NOT A GIC VERIFY CHANNEL
   NUMBER ENTERED, OR RUN GIC DIAGNOSTIC

   ENTER CHANNEL NUMBER TO WHICH THE MAG TAPE CONNECTED
   IS CONNECTED (1-15)

   The diagnostic returns to the Initialize phase and
   waits for you to enter the GIC channel number after
   outputting the above messages and request.

   If the thumbwheel setting and the channel number you
   entered were identical, the GIC PCA is bad. Run the
   GIC diagnostic to make sure and then replace the bad
   GIC PCA.

   If this step passes, the following request is output
   to the console:

   ENTER DEVICE NUMBER OF THE MAG TAPE UNDER TEST (0-7)

   Enter the Mag Tape controller device number that
   controls the unit(s) that are to be tested. If the
   value is out of range the request is repeated. The
   following request is also output to the console: PUT
   MAGNETIC TAPE(S) WITH WRITE RING ONTO CONSIDER
   TAPE(S) AND PUSH ON-LINE AND SELECT UNIT#(S)!

   This step also allows you to set up the number of
   passes to be made and the actual tape units that are
   to be tested via the following requests:

   ENTER THE NUMBER OF REQUIRED PASSES
   ?

If "-1" is entered, the diagnostic will loop on the test sections indicated indefinitely. However, you are always given an opportunity to exit the program execution via the ABORT command, etc. (see Section 2 for more explanation).

4   The ability of the CPU to interrupt the GIC is tested. This step forces an interrupt by writing into register 12. A channel program is then used to verify proper channel interrupt.

Program execution is terminated if this step fails and the following error message is output to the console:

CPU TO GIC INTERRUPTS NOT WORKING - PROGRAM TERMINATED

If this step fails, there is a good indication that the CPU has failed. Run the CPU self test for verification.

5   The device number, entered in step 3, is used to verify that the device specified is a 7970E magnetic tape subsystem. To do this the diagnostic executes a channel program to identify the device entered in step 3. The identify logic obtains a value from the device which is then used to verify that the device is a magnetic tape subsystem. If it is not a magnetic tape subsystem, the following error messages and request is output to the console:

IDENTIFY FAILURE. ID NUMBER RECEIVED FROM CONTROLLER IS xx SHOULD BE **.

Then the program is aborted.

One other message can be output if no value is obtained via the identify logic. In this instance, the following message and request is output to the console: NO ID NUMBER RECEIVED FROM CONTROLLER. Then the program is aborted. If the device did not identify itself correctly, verify the following possible causes and perform the recommended corrective action.

o Subsystem power is off - verify and correct.

o   tape controller HP-IB address is configured wrong
    - verify channel and device numbered entered.

o   HP-IB cable connection between GIC and tape
    controller is not made - could be faulty cable or
    dirty edge connectors.

o   Bad tape controller interface to HP-IB board -
    replace board.

6   reserved

7   The purpose of this step is to test data transfer
       accuracy when operating at full speed. The
       diagnostic executes a WRITE and READ HP-IB loopback
       using 256 bytes for a write and 129 bytes for a read
       to verify data communication among Memory/GIC/Tape
       Controller.

       If you are not successful, the following message is
       output to the system console:

       NO W/R LOOPBACK BETWEEN CPU AND TAPE CONTROLLER
       CHECK INTERFACE PCB, CONTROLLER PCB OR CABLES.

       In this error condition, verify that the GIC or the
       tape controller is good. The diagnostic is
       terminated and control is returned to the AID entry
       mode.

9   The status from all selected units to be tested is
       requested in this step. As each ready unit is
       discovered, the following message is output:

       UNIT x WILL BE TESTED

       where x equals the unit number to be tested.
       Example: If unit 0 and 3 respond with a ready
       status, the following messages are output to the
       console:
       UNIT 0 WILL BE TESTED
       UNIT 3 WILL BE TESTED
       The diagnostic asks for your agreement via the
       following request:

       IF OK TYPE 0 ELSE 1

       If you enter "0" the diagnostic executes step 9. If
       "1" is entered, the diagnostic returns to step 8 to
       repeat the unit set up instructions. Verify the
       unit set up as follows:

       o Verify thumbwheel settings on unit to be tested.

o   Check to see whether the write ring is missing on the tape spools installed. If installed, verify that the WRITE RING switch is good.

o   Verify that the select switches on the units to be tested are working.

o   Check the cable connection between Mag Tape controller and all units to be tested.

10   Step 10 does the final housekeeping before passing control to the first selected test section.

NOTE

This diagnostic always loops to step 10 until the limit on the number of passes is reached.

4.3 TEST SECTIONS 1-5 (Non Interactive Tests)

There are eight (8) test sections in the diagnostic. Sections 1 through 5 are non-interactive and can be executed as one group of tests to every available tape unit specified via the Basic Operating Mode procedure (see Section 2). In sections 1 through 5 all applicable tests are executed starting at the tape unit with the lowest number.

Note that sections 1 through 5 can be executed individualy when specified via the Extended Operating Mode. (Refer to tables 2 and 3 at the end of this manual section.)

Each test section specified is looped on, as a group, the number of times specified in step 8.

TEST SECTION 1. This section tests fifteen (15) basic commands and one illegal (reserved) command code. The command 'REWIND and GO OFF- LINE' will be tested in the interactive Test Section 6.

STEP: DESCRIPTION:

100   Controls the execution of all steps of Test Section 1 (101- 124) to all available Tape Units starting with the lowest number (0) up to the highest number (3).

101   Issues SELECT UNIT X to all existing units (0-3) and checks status.

102   Issues SELECT UNIT No., originally selected in step 100.

103   Issues WRITE FM (File Mark) and checks status.

104   Issues WRITE RECORD of a buffer with 100 words and burst length =60. The buffer has 10 words: 0,!1111,!2222,!3333, !4444,!5555,!6666,!7777,!AAAA,!BBBB, repeated 10 times. Then checks status.

105   Issues WRITE RECORD of an another buffer with 100 words and burst length=63.The buffer has 10 words: !BBBB,!AAAA,!7777, !6666,!5555,!4444,!3333,!2222,!1111,0 repeated 10 times. Then checks status.

106   Issues WRITE GAP and checks status.

107   Issues WRITE FM (File Mark) and checks status.

108   Issues BACK SPACE RECORD and checks status.

109   Issues BACK SPACE RECORD and checks status.

110   Issues READ RECORD (written in step 104) and compares read data with the original and checks status.

111   Issues READ RECORD BACKWARD (The same record in opposite direction), compares read data with original and checks status.

112   Issues BACKSPACE FM (File Mark) and checks status.

113   Issues FORWARD SPACE FM (File Mark) and checks status.

114   Issues FORWARD SPACE RECORD and checks status.

115   Issues READ RECORD (written in step 104) and compares read data with the original and checks status. 116 Issues REWIND and checks status.

117   Issues 450 GAPs to clean tape and checks status.

118   Issues READ RECORD BACKWARD to prove GAP works in the backward motion, BACK SPACE FILE and BACK SPACE RECORD respectively. After each program, the status is checked to get tape runaway. Then a REWIND is issued.

119   Issues REWIND to check no motion occurs and checks status.

120   Issues FORWARD SPACE FM (File Mark) to get Tape Runaway in status byte 2 (bit 4 set).

121   Issues FORWARD SPACE RECORD to get Tape Runaway in status byte 2 (bit 4 set). (See table 5.)

122   Issues READ RECORD to get Tape Runaway in status byte 2 (bit 4 set). (See table 5.)

123   Issues a channel program with the illegal command code = 0 to get Command Rejected in status byte 1 (bit 4 set). (See table 5.)

124   Issues REWIND to complete Test Section 1 for one Tape Unit and checks status. (See table 5.)

TEST SECTION 2. This section tests three (3) status bytes in more detail. Channel programs are executed to simulate situations which set bits in the three (3) status bytes and the program checks their proper setting. The following bits are not tested in this test section Bits 5 (File Protected) and 7 (Off-line) of byte 1, bit 8 (Reserved) of byte 2 and bits 5,7/8 (Reserved) and bit 6 (Power has been restored) of byte 3. The Reserved bits of status will be not tested, the rest will be tested in the interactive section 6. (See tables 4, 5, and 6 at end of this Section.)

STEP: DESCRIPTION:

200   Controls the execution of all steps of Test Section 2 (201- 230) to all available Tape Units starting with the lowest number (0) up to the highest number (3). 201   Issues SELECT UNIT 0 and checks proper bit set in status byte 2 (bits 9,10 - Select unit number)

202   Issues SELECT UNIT 1 and checks proper bit set in status byte 2 (bits 9,10 - Select unit number)

203   Issues SELECT UNIT 2 and checks proper bit set in status byte 2 (bits 9,10 - Select unit number)

204   Issues SELECT UNIT 3 and checks proper bit set in status byte 2 (bits 9,10 - Select unit number)

205   Issues SELECT UNIT No., originaly selected in step 200.

206   Issues illegal command "0" (Reserved) to simulate COMMAND REJECTED in status byte 1 (bit 4 set).

207   Issues WRITE FM (File Mark) and checks status.

208   Issues WRITE RECORD with 20,000 words and checks status.

209   Issues REWIND without WAIT and then checks status byte 2: bits 5 (Rewinding) set, 6 (Tape Unit busy) set, and 7 (Interface busy) reset.

210   Waits until Unit busy reset and checks status.

211   Issues FORWARD SPACE FM (File Mark) and checks
        status.

212   Issues READ RECORD with the record/burst length=60
        and DSJ (Device Specifies Jump) exits after first
        read burst to check bit 0 (EOF - End of File) of
        status byte 1 not set.

213   Issues REWIND and checks status.

214   Issues FORWARD SPACE RECORD and checks status.

215   Issues READ RECORD written in step 208 with burst
        length = 64 and 20,000 words and DSJ (device
        specified jump) providing proper control to complete
        reading and checks bit 0 (EOF - End of File) of
        status byte 1 is not set.

216   Issues REWIND and checks status.

217   Issues FORWARD SPACE FM (File Mark) and checks
        status.

218   Issues WRITE RECORD with large record and without
        WAIT to provide a incomplete write via the next
        step.

219   Issues SELECT UNIT NO (selected in step 200), checks
        status, and waits for it to be ready.   220
        Issues GAPs in loop until EOT (End of tape) is
        reached to check if bit 2 of byte 1 set.

221   Issues REWIND and checks status.

222   Issues FORWARD SPACE FM (File Mark) and checks
        status.

223   Issues READ RECORD with 20,000 words to check DSJ
        not equal to 0. Then issues CLEAR DEVICE, SELECT
        UNIT and checks status.

224   Issues FORWARD SPACE FM (File Mark) to check bit 4
        (Tape Runaway) of status byte 2 is set.

225   Issues FORWARD SPACE RECORD to check bit 4 (Tape
        Runaway) of status byte 2 is set.

226   Issues READ RECORD to check bit 4 (Tape Runaway) of
        status byte 2 is set.

227   Issues REWIND and checks status.

228   Issues FORWARD SPACE FM (File Mark) and checks status.

229   Issues WRITE RECORD with 6 data blocks and DC (Data Chain). Each data block has only 1 data word (2 bytes) to simulate DATA ERROR (Timing). Bit 3 (DATA ERROR-Timing) of status byte 2 should be set.

230   Issues REWIND to complete Test Section 2 and checks status.

TEST SECTION 3. This section tests the physical length of written data block onto a magnetic tape (timing).

STEP: DESCRIPTION:

300   Controls the execution of all steps of Test Section 3 (301- 312) to all available Tape Units starting with the lowest number (0) up to the highest number (3).

301   Issues SELECT UNIT No. (whichever is selected in step 300) and WRITE FM (File Mark) and checks status.

302   Issues WRITE RECORD with 8000 words and checks status.

303   Issues BACK SPACE FILE (FM) and checks for no bit 1 (BOT - Load point) of status byte 1. 304 Issues WRITE FM (File Mark - the same FM written in step 301) and checks status. 305 Issues READ RECORD with 8000 bytes and checks bit 3 (STE - Single Track Error) and bit 6 (MTE - Multi-Track Error) of status byte 1 not set.

306   Issues WRITE FM (File Mark) and checks status.

307   Issues BACK SPACE FILE (FM), measures the expended time to move back and checks it to be below 85 msec.

308   Issues BACK SPACE FILE (FM) and checks status.

309   Issues WRITE FM (file mark) to overwrite FM written in step 306 and part of record written in step 302.

310   Issues READ RECORD to check DSJ not equal 0.

311   Issues two (2) BACK SPACE FILE (File Mark) and one (1) FORWARD SPACE FM (File Mark) and checks status.

312   Issues REWIND to complete Test Section 3 and checks status.

TEST SECTION 4. This section tests tape creeping with start/stop motion and problems related to this.

    STEP: DESCRIPTION:

400   Controls the execution of all steps of Test Section 4 (401- 431) to all available Tape Units starting with the lowest number (0) up to the highest number (3).

401   Issues SELECT UNIT No. selected in step 400 and two (2) WRITE RECORDs with 200 words per each record and checks status.

402   Issues two (2) BACK SPACE RECORDs and checks status.

403   Executes a channel program with two (2) READ RECORDs, measures elapsed time and checks for bit 3 (STE - Single track error) and bit 6 (MTE - Multi Track Error) of status byte 1 not set.

404   Executes a channel program with BACK SPACE RECORD and WRITE RECORD 10-times to make a drift of the second record and checks status after last execution. 405 Issues two (2) BACK SPACE RECORDs and checks status.

406   Executes a channel program with two (2) READ RECORDs, measures elapsed time and checks for bits 3 (STE - Single Track Error) and 6 (MTE - Multi Track Error) of status byte 1 not set.

407   Issues REWIND and checks status.

408   Computes a difference in time from step 403 (normal written record) and from step 406 (10-times overwritten record). Then checks a time difference for no negative creeping and positive creeping below 12%.

409   Executes 10 channel programs with WRITE FM (File Mark) and 2 milisecond delay between each command to simulate START-STOP, measures the total elapsed time (T1) and checks status.

410   Executes a channel program with 10 BACK SPACE RECORDs, and 10 FORWARD SPACE RECORDs, measures total elapsed time (T2) and checks status.

411   Executes a channel program with 10 WRITE FMs (File Mark), measures total taken time (T3) and checks status.

412   Executes a channel program with 10 BACK SPACE FILEs (FM), and 10 FORWARD SPACE RECORDs, measures total elapsed time (T4) and checks status.

413   Checks if the elapsed time from step 409 (T1) is shorter then elapsed time from step 411 (T3) but no less then half.

414   Checks if the elapsed time from step 410 (T2) is shorter then elapsed time from step 412 (T4) by no less than 12%.

415   Issues REWIND to continue tests and checks status.

416   Issues a WRITE FM (File Mark) and checks status.

417   Writes five (5) data sets with one (1) FM (File Mark) after the last record of each data set. Every data set has 10 records of the same length. The length is computed so that the length of the first data set is 25 words long and each subsequent data set is increased by 25 words.

418   Controls the execution of steps 419 to 422 which loop five (5) times. Starts by selecting first data set (1).Next data is used during next loop(last loop uses data set number 5). 419   Issues REWIND and checks status.

420   Issues as many FORWARD FMs (File Mark) as the data set number selected in step 418 to get proper data set. 421   Executes 10 channel programs with READ RECORDs and proper length of record and BACK SPACE RECORDs.The elapsed time is measured during every execution and checked with limits after each read completion. Time limits are 32,36,40,44,52 msec for five (5) record sets respectively. The total elapsed time should be less than 10 seconds.

422   Loops back to step 418 until fifth data set is used.

423   Issues GAPs until EOT (End of Tape) comes up and one more GAP behind EOT to get proper status. Bit 2 of status byte 1 (EOT) is checked (should be set).

424   Issues WRITE FM (File Mark) and checks status.

425   Issues BACK SPACE FILE (FM) and checks status.

426   Executes a channel program with BACK SPACE RECORD in a loop for 200 msec to pass EOT (End of Tape) and checks status.

427   Issues WRITE FM (File Mark) and checks status.

428   Executes a channel program with FORWARD SPACE FILE and BACK SPACE FILE to check area before and after EOT (End of Tape) and checks status.

429   Issues WRITE RECORD with "Worst Data 'A'" (see Appendix B) and checks status.

430   Executes 200 times a channel program with BACK SPACE RECORD and READ RECORD and checks status.

431   Executes a channel program with WRITE RECORD, BACK SPACE RECORD and READ RECORD until EOT (End of Tape) comes up and checks status. The record is the "Worst Data 'B'" (see Appendix B).

432   Issues REWIND to complete Test Section 4 and checks status.

TEST SECTION 5. This section tests data transportation between tape and memory and among tapes if more then one tape unit is tested (Multi Unit Test).

    STEP: DESCRIPTION:

500   Controls the execution of all steps of Test Section 5 (501- 549) to all available Tape Units starting with the lowest number (0) up to the highest number (3).

501   Issues SELECT UNIT No. selected in step 500 and WRITE RECORD with a single word and checks status.

502   Issues BACK SPACE RECORD and checks status.

503   Issues READ RECORD with 1 word and checks data and status.

504   Issues WRITE RECORD with a single byte and checks status.

505   Issues BACK SPACE RECORD and checks status.

506   Issues READ RECORD with 1 byte and checks data and status.

507   Issues WRITE RECORD with 2000 words and checks status.

508   Issues BACK SPACE RECORD and checks status.

509   Issues READ RECORD with 2000 words and checks data and status.

510   Issues REWIND and checks status.

511   Creates a random data buffer with 4000 bytes.

512   Controls the execution of a sequence of channel
        programs. Each program has a WRITE RECORD with
        variable length. The initial record length is 4
        words increased by 4 for each successive record up
        to %7777 or EOT (End of Tape), whichever comes
        first. The first two words of each record are the
        length of the record and number of the record.

513   Issues WRITE FM (File Mark) and checks status.

514   Issues REWIND and checks status.

515   Controls the execution of the following sequence of
        channel programs (steps 516 to 520) until a FM (File
        Mark) is found.

516   Issues FORWARD SPACE RECORD and checks for EOF
        (End of File). If bit 0 of status byte 1 (The last
        tape motion was over FM-File Mark) is set (EOF),
        Step 523 is executed, else (no EOF) the program
        continues to Step 517 (Reading continues).

517   Issues BACK SPACE RECORD and checks status.

518   Issues READ RECORD with proper length provided in
        Step 515 and checks data and status. 519
        reserved

520   Passes the control to step 515.

521   reserved

522   reserved

523   Issues REWIND to all existing tape units and checks
        status.

Note: Steps 524 to 535 are skipped when only one(1) tape unit is
tested.

524   Controls the execution of the following sequence of
        channel programs until FM(File Mark) is found during
        execution. The execution starts with the tape
        unit with the lowest unit number(usually 0) as the
        source or read tape unit. The tape unit with the
        next lowest unit number is the receiving or
        writing tape unit. If three (3) or four (4) tape
        units are tested, the next test uses the tape
        previously used for receiving/writing as the
        source/read tape unit and the tape unit with the

next lowest unit number as receiving/writing ... up to unit number 3.

525   Controls execution of READ/WRITE records up to a size of 2000 bytes or EOF, whichever comes first. Start data record length is 4 bytes incremented by 4 for each successive record.

526   Issues SELECT UNIT number of the read unit and FORWARD SPACE RECORD and checks status. If EOF, the control passes to step 532, else it continues.

527   Issues SELECT UNIT number of write unit and checks status If EOT (end of Tape), the control passes to step 532, else it continues.

528   Issues SELECT UNIT number for read unit and BACK SPACE RECORD and checks status.

529   Issues READ RECORD and check first two (2) words (length and number of record respectively).

530   Issues SELECT UNIT number of write tape unit and WRITE RECORD and checks status. If EOT, the control passes to step 532, else it continues. 531 Passes the control back to step 525 to read/write next record.

532   Issues SELECT UNIT number for write unit and WRITE FM to write records and checks status.

533   Issues REWIND to write unit and checks status.

534   Issues SELECT UNIT number and REWIND to read unit and checks status.

535   Passes control back to step 524 to select new pair of write/read units.

536   reserved.

537   Issues GAP and checks status.

538   Issues BACK SPACE RECORD and checks for BOT (Load point) - bit 1 of status byte 1 set.

539   Issues WRITE FM (File Mark) and checks status byte 1: bit 1 (BOT - Load point) not set and bit 0 (EOF - End of File) set.

540   Issues SELECT UNIT No.(the same as step 534) and checks for status byte 1: bit 1 (BOT - Load point) not set and bit 0 (EOF - End of File) set.

541   Issues BACK SPACE FILE (FM) and checks status byte
        1: bit 1 (BOT - Load point) not set and bit 0 (EOF -
        End of File) set.

542   Issues FORWARD SPACE FILE and checks for status byte
        1: bit 1 (BOT - Load point) not set and bit 0 (EOF -
        End of File) set.

543   Issues BACK SPACE RECORD and checks for status byte
        1: bit 1 (BOT - Load point) not set and bit 0 (EOF)
        set.

544   Issues FORWARD SPACE RECORD and checks status and
        DSJ=1.

545   Issues BACK SPACE RECORD and check status and DSJ=1.

546   Issues READ RECORD with 10 words and checks for
        status byte and read byte count = 0.

547   Issues   400 WRITE FMs and checks status.   The
        execution is finished when either EOT comes up or
        the count = 400, whichever comes first.

548   Issues BACK SPACE FILE (FM) in loop and checks
        status byte 1, bit 1 (BOT- Load point). When this
        bit of byte 1 is set, the execution is finished.

549   Issues REWIND for all tested units to complete Test
        Section 5.

4.4 TEST SECTIONS 6-8 (Interactive tests)

Test sections 6 through 8 can only be invoked via the extended
operating mode. Each test section specified is looped on, as a
group.

TEST SECTION 6. This section is an optional interactive section
and tests all control elements on the front panel of the tape
unit(s) as well as a write ring, protecting data. When the test
section is default, this section is not executed.

STEP: DESCRIPTION:

601   Issues   two (2) messages with PAUSE. The first
        message asks the operator to put a magnetic tape •
        with   a   write ring onto a drive and set switches
        ON-LINE   and UNIT 0. The second message asks the
        operator to reset other tape units   if any other
        exist. The operator types 'GO'   and RETURN key
        to continue after it's done. Then the program checks
        status.

602   Issues four (4) SELECT UNIT NUMBERs for all 4 unit
        numbers (0-3) and checks status.

603   Issues a message "Push RESET" with PAUSE. The
        operator types 'GO' and RETURN key to continue after
        it's done. Then the program checks status.

604   Issues a message "Push ON-LINE" with PAUSE. The
        operator types 'GO' and RETURN key to continue.
        Then, the program checks status.

605   Issues a message "Push UNIT 1" with PAUSE. The
        operator types 'GO' and RETURN key to continue after
        it's done. Then the program checks status.

606   Issues a message "Push RESET" with PAUSE. The
        operator types 'GO' and RETURN key to continue after
        it's done. Then the program checks status.

607   Issues a message "Push ON-LINE" with PAUSE. The
        operator types 'GO' and RETURN key to continue after
        it's done. Then the program checks status. 608
        Issues a message "Push UNIT 2" with PAUSE. The
        operator types 'GO' and RETURN key to continue after
        it's done. Then the program checks status.

609   Issues a message   "Push RESET" with PAUSE. The
        operator types 'GO' and RETURN key to continue after
        it's done. Then the program checks status.

610   Issues a message "Push ON-LINE" with PAUSE. The
        operator types 'GO' and RETURN key to continue after
        it's done. Then the program checks status.

611   Issues a message "Push UNIT 3" with PAUSE. The
        operator types 'GO' and RETURN key to continue after
        it's done. Then the program checks status.

612   Issues a message   "Push RESET" with PAUSE. The
        operator types 'GO' and RETURN key to continue after
        it's done. Then the program checks status.

613   Issues a message "Push ON-LINE" with PAUSE. The
        operator types 'GO' and RETURN key to continue after
        it's done. Then the program checks status.

614   Issues a message "PUSH UNIT 0,1,2,3" with PAUSE.
        The operator types "GO" and then presses the RETURN
        key to continue after it's done. Then the program
        checks status and issues the next message:

        PUSH UNIT 0

with a PAUSE. The operator types next "GO" and RETURN key to continue after it's done. The the program checks status.

615   Issues a message "Push UNIT 1" with PAUSE. The operator types 'GO' and RETURN key to continue. Then, the program checks status.

616   Issues a message "Push UNIT 2" with PAUSE. The operator types 'GO' and RETURN key to continue. Then, the program checks status.

617   Issues a message "Push UNIT 3" with PAUSE. The operator types 'GO' and RETURN key to continue. Then, the program checks status.

618   Executes a channel program with WRITE RECORD and 100 words, BACK SPACE RECORD, READ RECORD with 100 words, and checks status.

619   Issues REWIND and checks status.

620   Issues an message "Remove write ring from reel tape and mount tape back onto drive" with PAUSE. The operator types 'GO' and RETURN key to continue after it's done. Then, the program checks status.

621   Issues WRITE RECORD with 100 words and checks bit 4 (Command Reject) set and bit 5 (No Write Ring) set, both of status byte 1.

622   Issues READ RECORD with 100 words and checks status.

623   Issues 'REWIND and GO OFF-LINE' and checks status.

624   Issues a message "Push ON-LINE, UNIT 0, and put tape with write ring onto drive" with PAUSE. The operator types 'GO' and RETURNS to continue after it's done. Then, the program checks status.

625   Issues a message "Turn OFF the power of a Tape Unit X" with PAUSE. The operator types 'GO' and RETURN key to continue after it's done. Then, the program checks status.

626   Issues a message "Turn ON the power of a Tape Unit" with PAUSE. The operator types 'GO' and RETURN key to continue after it's done. Then, the program checks status.

627   Issues a message "Push ON-LINE with PAUSE. The operator types 'GO' and RETURN key to continue after it's done. Then the program checks status.

628   Issues REWIND and SELECT UNIT number 0 to complete
        interactive test section 6 and checks status. `

TEST SECTION ˉ. This section is an optional interactive
section and executes a long writing, reading or reading of
special test tape HP 9162-0027 to adjust heads with an
oscilloscope. When the test section selection is default, this
section is not executed.

    STEP: DESCRIPTION:

700   Displays selectable test in a table and issues a
        message

        TYPE NUMBER OF SELECTED TEST The operator types "GO"
        and RETURN key to continue after a test selection.
        The following is an example of the selection table
        that is output.

    SELECTABLE TESTS IN SECTION 7 - USED FOR MEASUREMENT
    ==
    Code: Function: Write: Read:

    0 - EXIT
    1 - READ only   x
    2 - WRITE pattern all '1' then READ   x   x
    3 - WRITE pattern '1010 ... 10' then READ   x   x
    4 - WRITE pattern '11111110' then READ   x   x
    5 - WRITE worst data 'A' then READ   x   x
    6 - WRITE worst data 'B' then READ   x   x
    7 - WRITE 8 selectable bytes then READ   x   x
    8 - WRITE 8 selectable bytes   x

    Type 'Code' of selected test or exit

701   Passes the control to proper steps upon selected
        test in step 700.

702   Enters here, when selection in step 700 = 2 through
       8 and issues a message:

       PUT A MAGNETIC TAPE WITH A WRITE RING ONTO DRIVE
       PUSH UNIT 0 AND ON-LINE

       then issues the next message:

       RESET ALL OTHER TAPE UNITS IF ANY EXIST

       with PAUSE.   The operator types   "GO" to continue
       after it's done. Then the program checks status.
       If selection in step 700 = 7 or 8, the next four
       messages are issued, respectively:

       TYPE THE 1ST DATA WORD
       TYPE THE 2ND DATA WORD
       TYPE THE 3RD DATA WORD
       TYPE THE 4TH DATA WORD

       The operator types one single word in selected form
       (octal, hex, or decimal) after each message and
       RETURN to continue.

703   Issues WRITE RECORD with selected data in a loop
       until EOT is reached.

704   Issues REWIND and checks status.

705   Passes the control to step 703 (loop writing) when
       selection in step 700 = 8 (write only), else it
       continues.

706   Checks "GOPARAM3" to exit from Test Section 8 by
       going to step 700 (Exit), else (Keep Writing in a
       Loop), the control is switched to step 708 (next
       read).

707   Enters here, when selection in step 701=1 (Read in a
       Loop) and issues a message "Put a tape without a
       write ring onto drive, push UNIT 0 and ON-LINE".
       Then, issues the next message "Reset all other tape
       units if any exist" with PAUSE. The operator types
       'GO'   and RETURN key to continue after it's done.
       Then, program checks status.

708   Issues READ RECORD in a loop until EOT (End of
       Tape).

709   Issues REWIND and checks status.

710   Passes the control to step 708 (loop reading) when
       selection in step 700 = 1 (read only), else to step
       704 (loop writing).

711   Issues REWIND to complete interactive Test Section 7
        and and checks status.

TEST SECTION 8.   This   section   is   an   optional interactive
section   testing   magnetic tapes for a drop-out '1's and reports
results. When the test section selection is default, this section
is not executed.

STEP: DESCRIPTION:

801   Displays   two   (2)   messages   with PAUSE.   The first
        message asks the operator to put tested tape with a
        write ring onto a drive and set switches ON-LINE and
        UNIT   0.   The   second   message asks the operator to
        reset   all   other   tape   units   if   any   exist.   The
        operator types   'GO'   and RETURN   to continue after
        it's done. Then, the program checks status.

802   Issues   WRITE   RECORD   with   19000 bytes   (all '1's)
        representing one (1) foot of magnetic tape until EOT
        and   checks   status.   Any   DSJ<>0   is   computed and
        updated for the farthest distance of bad spot from
        BOT in feet.

803   Issues REWIND and checks status.

804   Issues   READ   RECORD with the   same long data blocks
        written   in step 802 until EOT and checks for DSJ=0.
        Any   DSJ<>0 is computed and updated for the farthest
        distance of bad spot from BOT in feet.

805   Issues REWIND and checks status.

806   Issues WRITE FM (File Mark) to shift writing in step
        807 and checks status.

807   Issues   WRITE RECORD (same as in step 802) until EOT
        and   checks   status.   Any   DSJ<>0   is   computed and
        updated for the farthest distance of bad spot from
        BOT in feet.

808   Issues REWIND and checks status.

809   Issues FORWARD FM (File Mark) and checks status.

810   Issues   READ RECORD with the same data block written
        in   step   807   until EOT and   checks for DSJ=0.   Any
        DSJ<>0   is   computed   and   updated for   the farthest
        distance of bad spot from BOT in feet.

811   Issues REWIND and checks status.

812   reserved

813   Checks "GOPARAM3" to exit from this tape testing
       section. Else the control passes to step 801 to
       start this over again.

TABLE 2.   COMMAND REGISTER

Value: Command:

```
0            Reserved
1            SELECT UNIT 0
2            SELECT UNIT 1
3            SELECT UNIT 2
4            SELECT UNIT 3
5            WRITE RECORD
6            WRITE FILE MARK (FM or EOF)
7            WRITE GAP
8  (%10)     READ RECORD
9  (%11)     FORWARD SPACE RECORD
10 (%12)     BACKSPACE RECORD
11 (%13)     FORWARD SPACE FILE (FM or EOF)
12 (%14)     BACKSPACE FILE (FM or EOF)
13 (%15)     REWIND
14 (%16)     REWIND AND GO OFF-LINE
15 (%17)     READ RECORD BACKWARD
```

TABLE 3.   'END' COMMANDS

Bit Number:	DIO Line Number:	Command:
0	8	Reserved.
1	7	Reserved.
2	6	Reserved.
3	5	Clear DSJ register for selected device only.
4	4	Reserved.
5	3	Enable parallel poll response (Service Request) for all devices connected to the interface.
6	2	Stop polling for data/inhibit parallel poll response for read bursts for selected device.
7	1	Clear parallel poll response (Service Request) for selected device only.

TABLE 4.   STATUS BYTE No.1

Word 1 Bit No:	Byte Bit No:	DIO Line No:	Means:
0	0	8	EOF - End of File of File Mark (FM)
1	1	7	BOT - Beginning of Tape or Load Point
2	2	6	EOT - End of Tape
3	3	5	STE - Single Track Error
4	4	4	Command Rejected
5	5	3	File Protected (No Write Ring)
6	6	2	MTE - Multiple Track Error
7	7	1	ON - LINE

TABLE 5.   STATUS BYTE No.2

Word 1 Bit No:	Byte Bit No:	DIO Line No:	Means:
8	0	8	Reserved.
9	1	7	Selected Tape Unit MSB (In channel program)
10	2	6	Selected Tape Unit LSB (In channel program)
11	3	5	Data Error (Timing)
12	4	4	Tape Runaway
13	5	3	Rewinding
14	6	2	Tape Unit Busy
15	7	1	Interface Busy

TABLE 6.   STATUS BYTE No.3

Word 2 Bit No:	Byte Bit No:	DIO Line No:	Means:
0	0	8	Reserved.
1	1	7	Reserved.
2	2	6	Power has been restored
3	3	5	Reserved.
4	4	4	Tape Unit 3 has been placed ON-LINE
5	5	3	Tape Unit 2 has been placed ON-LINE
6	6	2	Tape Unit 1 has been placed ON-LINE
7	7	1	Tape Unit 0 has been placed ON-LINE

```
+-------------------------------------------------------+ +----------+
|                                                       | | SECTION  |
|                         ERROR/ACTION SUMMARY          | |----------|
|                                                       | |    V     |
+-------------------------------------------------------+ +----------+
```

5.0 INTRODUCTION

The error/action summary contained in this section relates to steps 1, 3 through 5, and 7 in the test section descriptions.

5.1 STEP 1 ERROR/ACTION

Message = NON EXISTENT CHANNEL - CHANNELS PRESENT = xx   TRY ANOTHER THUMBWHEEL SEETING ON GIC, OR CHECK IMB, OR RUN GIC DIAGNOSTIC

Cause =   Expected response from roll call was not received.

Action =   o Verify thumbwheel switch settings

           o GIC bad - run GIC diagnostic

5.2 STEP 3 ERROR/ACTION

Message = CHANNEL NUMBER ENTERED IS NOT A GIC   VERIFY CHANNEL NUMBER ENTERED, OR RUN GIC DIAGNOSTIC

Cause =   Register 14 read indicates channel is not a GIC.

Action =   o Verify thumbwheel switch settings

           o GIC bad - run GIC diagnostic

5.3 STEP 4 ERROR/ACTION

Message = CPU TO GIC INTERRUPTS NOT WORKING - PROGRAM TERMINATED

Cause =   CPU cannot interrupt GIC

Action =   o CPU failed - run CPU self-test diagnostic

5-1

Magnetic Tape Diagnostic

5.4 STEP 5 ERROR/ACTION

Message = IDENTIFY FAILURE. ID NUMBER RECEIVED FROM CONTROLLER
          IS xx SHOULD BE **

Cause =   ID received is not a magnetic tape subsystem.

Action = o Power-to subsystem is off

          o HP-IB address configured wrong - verify and correct
          o Cable connection incorrect
          o Tape controller bad - replace

5.5 STEP 7 ERROR/ACTION

Message = NO W/R LOOPBACK BETWEEN CPU AND TAPE CONTROLLER
          CHECK INTERFACE PCB, CONTROLLER PCB OR CABLES

Cause =   Data transfer between system and tape controller
          was bad.

Action = o Verify GIC

          o Verify tape controller

```
+-----------------------------------------------------+  +-----------+
|                                                      |  | APPENDIX  |
|                                 GLOSSARY OF TERMS    |  |---------- |
|                                                      |  |     A     |
+-----------------------------------------------------+  +-----------+
```

BOT   Beginning of Tape or Load Point.

Creeping   A record usually drifts forward when WRITE RECORD
               is   executed after either   BACKSPACE FILE   or
               BACKSPACE RECORD instead of after any command with
               forward motion such as READ RECORD, WRITE RECORD,
               FORWARD SPACE RECORD or FORWARD SPACE FILE.   There
               can be positive or negative creeping in the
               process.   Negative creeping can   destroy   the
               previous record on tape and is not allowed.

DIO   Direct Input Output HP-IB Bus Lines (1-8).

DSJ   Device Specified Jump.

EOF   End of File or File Mark (FM).

EOT   End of Tape.

FM   File Mark or End of File (EOF).

ISA   International Standard Association

```
+--------------------------------------------------+ +----------+
|                                                  | | APPENDIX |
|                ISA STANDARD DATA PATTERNS        | |----------|
|                                                  | |    B     |
+--------------------------------------------------+ +----------+
```

The following lists give the ISA standards for Data A and B
arrays.

    1.   ISA Data 'A' Standard:

        Data Array: %260,
                        %040,
                        %157,
                        %260,
                        %040,
                        %157,
                        %260,
                         .
                         .
                         .
                        %157.

    2.   ISA Data 'B' Standard:

        Data Array: %101,
                        %000,
                        %276,
                        %101,
                        %000,
                        %276,
                        %101,
                         .
                         .
                         .
                        %276.

**13037B DISC CONTROLLER
DIAGNOSTIC MANUAL**

LIST OF EFFECTIVE PAGES

The List of Effective Pages gives the date of the current edition and of any pages changed in updates to that edition. Within the manual, any page changed since the last edition is indicated by printing the date the changes were made on the bottom of the page. Changes are marked with a vertical bar in the margin. If an update is incorporated when an edition is reprinted, these bars are removed but the dates remain.

All pages original issue . . . . Mar 1982

PRINTING HISTORY

```
-------------------------------------------------------------------
|                                                        CONTENTS  |
-------------------------------------------------------------------
```

SECTION I - GENERAL INFORMATION

SECTION II - OPERATING INSTRUCTIONS

SECTION IV - TEST SECTION DESCRIPTIONS

| **CONTENTS** (Cont) |

| **TABLES** |

```
+------------------------------------------------------+  +---------+
|                                                      |  | SECTION |
|                              GENERAL INFORMATION     |  |---------|
|                                                      |  |    I    |
+------------------------------------------------------+  +---------+
```

1.0 INTRODUCTION

This manual describes the loading and use of the 79XX/13037 Disc
Subsystem Diagnostic (hereafter called the program or diagnos-
tic).

The program confirms proper input, output, and control functions
for a disc subsystem which may include a 7906, 7920, and/or 7925
Disc Drive. The diagnostic provides rapid checkout of the con-
troller and drive.

A maximum of eight drives can be tested sequentially. Any combi-
nation of 7906, 7920, and 7925 drives may be tested at one time.

W A R N I N G

> Before running this diagnostic, be sure that the logical
> configuration of the system matches the physical
> configuration as established by the thumbwheel settings
> for each channel and device.

1.1 REQUIRED HARDWARE

This diagnostic requires that the console and the "cold load"
facility of the system be operational. It is not possible to
execute the 79XX/13037 Disc Diagnostic program if you are unable
to perform a "cold load".

The following hardware is also required:

o HP 2893A Option 333 Card Reader

o An HP 3000 HP-IB version computer system with minimum system
   memory configuration.

o A Disc Subsystem which includes a 7906, 7920, or 7925
   Disc Drive, 13037B Controller, and an HP-IB Interface Board.

o A cold load device to load the program.

1.2 REQUIRED SOFTWARE

Software required to execute this diagnostic is the Diagnostic
Utility system, AID, and the HP 79XX/13037A Disc Diagnostic
program properly recorded on a flexible disc, cartridge or
magnetic tape.

NOTE

If the DUS flexible disc, cartridge or magnetic tape is
damaged or lost, a new one may be created in one of two
ways:

As follows:

enter RUN DUSCOPY.HP32231.SUPPORT on console.

An extra copy of a DUS disc or tape may be created via
the FCOPY utility in MPE or by writing a sleuth program
using the Sleuth Simulator.

1.3 TESTING PHILOSOPHY

This diagnostic tests all control functions and data transfers.
Should an error occur, the diagnostic points to the disc's
physical location and outputs an error message that includes the
step number in which the error occured. Testing to the
functional level results in errors being isolated only to the
smallest replaceable assembly in most instances.

1.4 DIAGNOSTIC LIMITATIONS

Unexpected interrupts form a special class of errors; AID reports
these errors and displays the channel and device number of the
interrupt. The interrupts are a result of an interrupt from a
channel and/or HP-IB device not being used by the diagnostic.
Analysis of these errors is beyond the scope of this diagnostic.

Whenever the heads load on any unit while running the diagnostic,
an attention will occur. This attention is not expected by the
diagnostic , except in step 115, and is reported as an error.
The error message that results and the number of errors output
depend on what the diagnostic is doing at the time of the
unexpected attention. The unit portion of the actual status word
one identifies which unit interrupted unexpectedly.

NOTE
Step 115 expects the heads to load, therefore, an error
is not reported.

All possible status word one and status word two indications
are forced except for the following:

o Access not ready during data operation

o Drive fault

o I/O Program Error

1.5 MESSAGES AND PROMPTS

Two types of messages are output by the diagnostic; error and
information messages. Section V contains a summary of all
messages, both error and non-error, their meaning, and an
explanation of action to be taken.

1.51 ERROR MESSAGES

Error messages inform the operator when the
interface, controller, or drive fails to respond to a command or
sequence. Errors may be divided into two types; fatal errors and
non-fatal errors.

FATAL ERRORS - errors which would probably cause an operating
system to irrecoverably fail or have to be restarted. Examples
of this type of error are as follows:

o No handshake from the controller, indicating completion of a
  command from the CPU.

o An uncorrectable data error which remains uncorrectable upon
  rereading five times. This may be the result of errors
  introduced during a write function.

o Reading a sector which has been written to contain a data
  error and not getting a data error status from the disc sub-
  system.

NON-FATAL ERRORS -Soft errors (or non-fatal errors) may be divid-
ed into two types; Random errors and hard errors.

o Random errors are not fatal errors and can only be separated

o from hard errors by the operator, as explained below under
  hard errors.

o HARD ERRORS are reported by the disc subsystem and in
  general they do repeat when retried. The diagnostic does
  not differentiate between random errors and hard errors.
  You may determine which errors are hard by running the 79XX
  Disc Verifier program and locating the errors which seem to
  repeat. A probable cause of these errors is a bad track on
  the disc media. Such bad tracks should be flagged
  defective.

1.52 INFORMATION MESSAGES

Information messages are used to inform you of the progress of the diagnostic or to instruct you to perform an operation. If an associated pause occurs with a message (indicated by the prompt character (>) being output to the console, you may input GO (to continue execution of the diagnostic) or any other appropriate command (refer to Section 2, paragraph 2.3).

1.6 MINI-OPERATING INSTRUCTIONS

```
+=================================================================+
|1.  Turn on the power to all other necessary devices.            |
|                                                                 |
|2.  Select the channel and device number of the 7902 Disc Unit|
|    and perform the "cold load" procedure.  NOTE:  Do not use  |
|    "warm start" switches.                                       |
|                                                                 |
|3.  The system outputs the following messages:                   |
|                                                                 |
|    Diagnostic/Utility System   Revision XX.YY                   |
|    Enter your program name (type HELP for program informa-      |
|    tion) >                                                      |
|                                                                 |
|    Enter "D13037".  The program outputs the following mes-      |
|    sage:                                                         |
|                                                                 |
|    79XX/13037 Disc Memory Diagnostic X.X                        |
|    Install scratch disc packs in all units to be tested.        |
|    Enter: GO to continue                                        |
|      >                                                          |
|                                                                 |
|5.  Enter "GO" in response to the prompt (>).                    |
|                                                                 |
|6.  Respond to the following instruction messages appropri-      |
|    ately.                                                        |
|                                                                 |
|    Enter Channel Number to Which the 13037 Controller is        |
|    Connected (1-15)                                             |
|    ?                                                            |
|    Enter Device Number Assigned to the Controller by the        |
|    HP-IB (0-7)                                                  |
|    ?                                                            |
|                                                                 |
| The program is now ready to execute the diagnostic testing as|
| as soon as the last question is answered.                      |
+=================================================================+
```

```
+-----------------------------------------------------+ +----------+
|                                                     | | SECTION  |
|                 OPERATING INSTRUCTIONS              | |----------|
|                                                     | |    II    |
+-----------------------------------------------------+ +----------+
```

2.0 INTRODUCTION

There are two methods for loading and executing this diagnostic program; the Standard operating mode and the Optional Operating mode. In the Optional operating mode you are given the opportunity to cause the diagnostic to run longer, test the drive more thoroughly, etc.(refer to GO 1 Operating Mode, paragraph 2.3).

The first 8 steps of the Standard Operating procedure are identical to the first 8 steps of the Optional operating procedure and, therefore, are not repeated in the procedure for Optional operation.

WARNING

Before running the diagnostic, be sure that the physical configuration (switches on front of card cages) matches the logical configuration required by the operating system.

2.1 STANDARD OPERATING PROCEDURE

In this operating mode, the diagnostic program automatically executes test sections 1, 3, 4, and 5 using the drives to test the controller. At the end of test section 5, the program prints its pass message and then terminates.

NOTE

A standard pass for each section takes 5 minutes. Since each pass is different due to random tests, an exhaustive controller test should consist of multiple standard passes. Also, more exhaustive testing can be invoked by using some of the extended diagnostic features in the GO 1 operating mode. In the standard operating mode, short data transfers do occur using the drive to test the controller.

1. Turn on power to all devices required to run the diagnostic (see Section I, Hardware Requirements).

2. Place Diagnostic Untility flexible disc or tape in the cold load device.

3. To "cold load" the diagnostics, perform the following steps:

   a. Set the COLD LOAD thumbwheel switches, on the front panel, to the channel/device number of the cold load device.

      b. Press the HALT button on the front panel.

   c. Press the COLD LOAD button on the front panel. The "cold load" firmware will begin and the RUN button will light. The Diagnostic Utility program then displays its title message and prompt character (:) as follows:

      Diagnostic/Utility System   Revision xx.yy
   Enter your program name (Type HELP for program inform-
                           ation)
                             :

   d. Enter "D13037" at the system console. The 79XX/13037 Disc Diagnostic is loaded from the flexible disc or tape and it displays its title message and prompt character(>) as follows:

      79XX/13037 Disc Memory Diagnostic
      Install scratch disc packs in all units to be tested.
      Enter: GO to continue
      >

   Install a scratch cartridge/pack in all units to be tested since the diagnostic will write over information already stored.   If a scratch cartridge/pack is

   unavailable save current information to another media and restore it from that media later, by using a sleuth simulator program (in stand-alone operations only). The RESTORE utility program may also be used on-line.

   4. To continue execution of the standard operating mode, enter "GO" at the console.

   5. The system responds by requesting information about the configuration you want to test via the following request messages:

      Enter Channel number to which the 13037 controller is
                        connected (1-15)
                             >
      Enter Device number assigned to the controller by the
                           HP-IB (0-7)
                             >

Enter the channel number of the GIC to which the 79XX
controller is connected. Next enter the device number of the
79XX controller that controls the unit(s) to be tested.
Verify that the numbers you enter match those actually set on
the channel and device.

6. The program scans all units connected to the controller and
   prints the following messages:

   Units which are ready
   Unit-Type
   n   79XX
   .   .
   .   .
   .   .
   Do you want to write on all removable surfaces?

   Enter "Y" to write on all removable surfaces on all units
   shown. Enter "N" to alter the units selected or the surfaces
   selected for each unit.

7. Once the diagnostic knows which configuration to test, the
   following request is made:

   Enter the number of required passes (-1 = indefinitely)
                              >

   The diagnostic will begin execution and continue execution
   through section 5 or until an error condition is detected.
   If no errors are detected and section 5 is completed, the
   pass is complete.

8. For the execution of the Optional diagnostic features, see
   paragraph 2.2.

When errors occur and the error message includes the step number,
the step number may be used as an index into section 4 of this
manual to obtain a detailed description of the step.

   The standard operating mode has the following constraints:

o Test sections 1, 3, 4, and 5 are executed and looped on the
   number of times specified.

   o Error, Information, and Prompt messages are displayed.

      o Pauses on errors and prompts are not suppressed.

o Run time options 0 and 6 are active (see table 2-1 for an
   explanation of these options).

2.2 OPTIONAL OPERATING MODES

The optional operating mode includes the following diagnostic
features:

   o The ability to select particular test sections for
      execution.

 o The ability to suppress or enable error printout and pauses,
    and suppress or enable non-error printout and pauses.

To specify the optional diagnostic features mentioned above,
perform the following procedure:

   1. Execute steps 1 through 4 of the basic operating mode.

   2. When the system pauses and the prompt character (›) is
       displayed enter one or more of the setup commands shown in
       table 2-1 in the appropriate format given.

3. Once you have specified the optional diagnostic features you
    desire, enter "GO" via the system console.

Table 2-1 describes two (2) types of commands: setup commands
and control commands.   Setup commands can be entered before the
diagnostic program if AID is already running or during a program
pause.   They are not actually acted upon until the diagnostic
program is running.

   Control commands can be entered whenever the system pauses.
They are acted upon immediately (i.e., entering "GO" causes the
diagnostic to begin execution).

NOTE

   Optional diagnostic features may also be specified
   whenever system pauses. To force a pause press the CNTL
   key and while holding this key down, press the Y key on
   the console keyboard.

   At this point all commands shown in Table 2-1 are valid.

Table 2-1. 79XX/13037 Diagnostic Commands

```
|-------------------------------------------------------------------|
|                       SETUP COMMANDS                              |
|-------------------------------------------------------------------|
| COMMAND FORMAT:  EEPR cr        (where cr means to press the       |
|                                 RETURN or ENTER key.               |
|                                                                   |
| DESCRIPTION:  The execution of this command lets the program      |
| display error messages as they occur.  The displaying of          |
| error messages in normally enabled when the diagnostic is         |
| initiated for execution.  Therefore, this command would           |
| normally be executed after the SEPR command (suppress error       |
| messages).                                                        |
|-------------------------------------------------------------------|
| COMMAND FORMAT:  EEPS cr                                           |
|                                                                   |
| DESCRIPTION:  The execution of this command lets the program      |
| generate pauses after an error occurs.  Pausing after error       |
| messages are output is invoked when the diagnostic is first       |
| executed.  Therefore, the execution of this command would         |
| normally take place after the SEPS command is executed.           |
|-------------------------------------------------------------------|
| COMMAND FORMAT:  ENPR cr                                           |
|                                                                   |
| DESCRIPTION:  The execution of this command lets the program      |
| display non-error messages as they occur.  When the program       |
| is first initiated, this condition automatically exists.          |
| Therefore, the execution of this command would normally take      |
| place after the SNPR command is executed.                         |
|-------------------------------------------------------------------|
| COMMAND FORMAT:  GO 1  or  RUN 1                                   |
|                                                                   |
| DESCRIPTION:  This command allows you to further define how       |
| the diagnostic is to be executed.  When GO 1 is entered, the      |
| following request is output to the console:                       |
|                                                                   |
| Enter:(U)nit,(?)errs,(H)ead,(O)utput,(L)ineprinter,               |
| (C)yl,(T)imes,(R)un Time Option,(E)xit.                           |
|                                                                   |
| Select the area to be changed by entering the letter in the       |
| parenthesis (i.e., enter "H" to select the head table).           |
| Once you have made this entry, the program outputs the query      |
|                                                                   |
| Wish to Change?                                                   |
|                                                                   |
| In response, enter (Y)es, (N)o, (E)xit, or any of the above       |
| letters in parenthesis in the first message.  Once you have       |
| responded to the above question, the program (if applicable)      |
| outputs the prompt (?) and waits for changes to be entered        |
| or for another area to be selected.                               |
|-------------------------------------------------------------------|
```

Table 2-1. 79XX/13037 Diagnostic Commands (cont'd)

```
|------------------------------------------------------------------|
|                        SETUP COMMANDS                            |
|------------------------------------------------------------------|
| GO 1 and RUN 1 (cont'd)                                          |
|                                                                  |
| (E)xit must be entered to exit from this mode of operation.      |
|                                                                  |
| Each area of the first message is defined in paragraph 2.3.      |
|                                                                  |
|                        NOTE                            |         |
|                                                                  |
|      If GO 1 is entered, before the end of a test                |
|      section is complete, the program completes the              |
|      test section first and then branches to the                 |
|      GO 1 operating mode.                                        |
|------------------------------------------------------------------|
| COMMAND FORMAT:  LOOP cr                                         |
|                                                                  |
| DESCRIPTION:  This command tells the diagnostic to loop on       |
| all test sections to be executed (as indicated by the TEST       |
| command).  This overrides the initial input of number of        |
| times to loop.  To exit a loop execution perform the            |
| following steps:                                                 |
|                                                                  |
| 1.  Enter CNTL Y for HP 3000/33 or ATTENTION for HP300           |
| 2.  When the diagnostic program pauses, enter "EXIT cr"          |
|                                                                  |
| To Cancel the loop function completely, enter "LOOP OFF".        |
|------------------------------------------------------------------|
| COMMAND FORMAT:  SEPR cr                                         |
|                                                                  |
| DESCRIPTION:  The execution of this command lets the program     |
| suppress error messages and error pauses until an EEPR or        |
| RST command is acknowledged.                                     |
|------------------------------------------------------------------|
| COMMAND FORMAT:  SEPS cr                                         |
|                                                                  |
| DESCRIPTION:  The execution of this command lets the program     |
| suppress error pauses only until an EEPS or RST command is       |
| acknowledged.                                                    |
|------------------------------------------------------------------|
| COMMAND FORMAT:  SNPR cr                                         |
|                                                                  |
| DESCRIPTION:  The execution of this command lets the program     |
| suppress non-error messages that can appear on the console.      |
| The ENPR and RST command will override this command.             |
|------------------------------------------------------------------|
```

Table 2-1.   79XX/13037 Diagnostic Commands (cont'd)
```
|------------------------------------------------------------|
|                      SETUP COMMANDS                        |
|------------------------------------------------------------|
| COMMAND FORMAT:   TEST [+ or -][x(/x)(,x)]                  |
|                   TEST ALL                                 |
|                                                            |
| where x = a program section number                         |
|                                                            |
| DESCRIPTION:  The TEST command allows you to select one or |
| more test sections for execution.  A summary and detailed  |
| description of all test sections can be found in Section 4.|
|                                                            |
| The sections that are executed during the standard operating|
| mode are non-interactive (Sections 1,3,4, and 5). Run time |
| options 0 and 6, set, cause the standard version to be exe-|
| cuted.  Therefore, to execute the Optional version of the  |
| diagnostic, you must first clear run time option 6 (select |
| (R) parameter of GO 1 operating mode) and then select test |
| sections 1 through 5 for complete testing of the controller.|
|                                                            |
| Example usage of the TEST command are as follows:          |
|                                                            |
| TEST 1,3/5     Executes test section 1 and 3 through 5.    |
| TEST 1/5       Executes test section 1 through 5.          |
| TEST + 2       Adds section 2 to the sections being tested |
| TEST - 3       Deletes section 3 from the sections being   |
|                tested.                                     |
|                                                            |
| NOTE:  If you want to do multiple unit testing you must    |
|        execute test section 4 for each unit under test    |
|------------------------------------------------------------|


|------------------------------------------------------------|
|                     CONTROL COMMANDS                       |
|------------------------------------------------------------|
| COMMAND FORMAT:   EXIT cr                                  |
|                                                            |
| DESCRIPTION:  When you complete entering the EXIT command, |
| control is returned to the Diagnostic Utility System       |
| where this diagnostic program may be initialized again,    |
| or, you may take any other appropriate action.            |
|                                                            |
| The EXIT command can only be entered when the diagnostic   |
| program pauses.  However, you may force an exit via the    |
| following steps:                                          |
|                                                            |
| 1. Enter CNTL Y.                                          |
| 2. When the diagnostic program pauses, enter "EXIT cr".    |
|------------------------------------------------------------|
```

Table 2-1. 79XX/13037 Diagnostic Commands (cont'd)
```
|------------------------------------------------------------------|
|                      CONTROL COMMANDS                            |
|------------------------------------------------------------------|
| COMMAND FORMAT:  GO cr                                            |
|                                                                  |
| DISCRIPTION:  The GO command can only be entered during a        |
| diagnostic program pause.  When the GO command entry is          |
| complete the system will continue.                               |
|------------------------------------------------------------------|
| COMMAND FORMAT:  RST cr                                           |
|                                                                  |
| DESCRIPTION:  This command resets all execution commands to      |
| their default state as follows:                                  |
|                                                                  |
| Error Pauses are enabled (EEPS command)                          |
| Error Messages are not suppressed (EEPR command)                 |
| Non-error Messages are not suppressed (ENPR command)             |
| Operator interaction is enabled                                  |
|------------------------------------------------------------------|
| COMMAND FORMAT:  RUN cr                                           |
|                                                                  |
| DESCRIPTION:  The "RUN" command, when entered without any        |
| parameters, causes the diagnostic program to begin execution|
| at the lowest numbered statement regardless of the present       |
| state of execution. Refer to the AID manual for more infor-      |
| mation.                                                          |
|------------------------------------------------------------------|
```

2.3 GO 1 Operating Mode

When GO 1 or RUN 1 is entered during a system pause, the
following request is output to the system console:

Enter: (U)nit,(?)errs,(H)ead,(O)utput,(L)ineprinter,(C)yl,
(T)imes,(R)un Time Option,(E)xit.

Each letter entered, shown in parenthesis, causes a particular
action to occur as defined in subsequent paragraphs. When asked
if you want to change an item you may answer "Y" (yes). "N"
(no), or use one of the letters shown above, to examine other
items.

2.30 (U)nit - Unit Selection Table

Units to be tested by the diagnostic are listed in this table.
When "U" is entered, the program asks if you "Wish to change?".
If "Y" is entered you may, when prompted, enter all of the units
you wish to test. Example: If unit 0 has already been specified
and you decide to test unit 1 also, enter 0,1. If only unit 1 is
specified then unit 0 will be deleted from the table.

If you specify a unit whose "not ready" bit in the status word is not set, the following message is output:;

Unit X not ready

Verify that the drive is on line, is up to speed, the heads are loaded, and/or the DRIVE FAULT indicator is not lit.

2.31 (?)errs - Unit Removal Selection

When the number of errors per pass indicated in the error table is reached during diagnostic execution, the unit being tested is removed from the unit table.

When the diagnostic program is first executed (initiated), the number of errors allowed per pass is 20.

When "?" is entered, the program asks if you "Wish to change?". Enter "Y" for yes and then when the prompt (?) appears enter desired number of errors that can occur before the unit is removed from the unit table.

2.32 (H)ead - Head Selection Table

The heads used in the diagnostic are listed in this table. A separate head table is provided for each unit in the Unit table (the unit must be ready at the time it is included in the unit table).

When the diagnostic is first executed (initialized), the heads used for the 7906 are 0 and 1; for the 7920 the heads used are 0,1,2,3, and 4; for the 7925 the heads used are 0 through 8.

2.33 (O)utput - Select Error Status Format

There are 2 error status formats output by the diagnostic. The extended format is the one normally output when the program is first executed. When "O" is entered, the program asks if you "Wish to change?". Entering "Y" for yes causes the program to change from the expanded format to the non-expanded format or vise versa. If "N" for no is entered, a prompt (?) is output and you can enter any appropriate letter. The following are examples of the expanded and non-expanded formats provided by this diagnostic.

Expanded format (default):

	S	P	D	TSTAT	XXXX	UNIT	/	E	DRTYPE	X	A	P	F	DP	PS	SC	NR	B
Status Is	0	0	0	01000	0000	0000	/	0	000001	0	0	0	0	0	0	0	0	0
Should Be	0	0	0	00000	0000	0000	/	0	000001	0	0	0	X	0	0	0	0	0

TSTAT is uncorrectable data error should be normal completion

Non-expanded format:

	S	P	D	TSTAT	XXXX	UNIT	/	E	DRTYPE	X	A	P	F	DP	PS	SC	NR	B
Status Is	0	0	0	01000	0000	0000	/	0	000001	0	0	0	0	0	0	0	0	0
Should Be	0	0	0	00000	0000	0000	/	0	000001	0	0	0	X	0	0	0	0	0

2.34 (L)ine Printer - Direct Messages to Line Printer

Entering this parameter allows you to choose whether error messages are to be output to the line printer or to the console. When the diagnostic program is first executed (initiated), all error messages are automatically directed to the console.

When "L" is entered, the program asks if you "Wish to change?". Entering "Y" for yes causes the program to switch from outputting to the console to outputting to the line printer and vice versa. Entering "N" for no causes the program to output the prompt (?) so that you can enter any letter; leaving the message output device unchanged.

2.35 (C)yl - Select Cylinder Table

When the diagnostic program is first initiated (executed), a cylinder table is created, for each disc device type, with twelve cylinder numbers in each table. The following values are initially placed in the tables accordingly.

   For 7906: 0,1,2,5,8,17,32,65,128,192,256, and 410

   For 7920 and 7925: 0,1,2,5,6,17,32,65,128,257,512, and 822

When "C" is entered, the program asks "Wish to change?". To change, enter "Y" for yes; the program responds with a prompt. You may now enter the cylinder numbers you desire to test. Note that cylinders are replaced in the order that they are entered. Consecutive commas imply no change. Maximum number of cylinders you may enter is 12.

Entering "N" for no implys that you do not want to change the cylinder table. If "N" is entered, the program outputs a prompt and waits for an appropriate entry.

2.36 (T)imes - Number of Times Diagnostic is Run

This variable is initialized to the number you enter in response
to the request "Enter the number of required passes (-1 =
indefinitely)". When "T" is entered, the program asks "Wish to
change?". Enter "Y" for yes to change and when the program
prompt is output, enter the number of passes you want to occur.

Note that any number entered that is preceded by a minus (-) sign
will cause the diagnostic to run until it is interrupted (see
Control Commands in table 2-1).

If you do not want to change this value, enter "N" for no and the
program will output a prompt so that you can enter any letter
appropriate to this mode of operation.

2.37 (R)un Time Option - Select Extended Diagnostic Options

When the program is first executed, options 0 and 6 are set.
When "R" is entered, the program asks "Wish to change?". Enter
"Y" for yes and the program will output its prompt. Once the
prompt appears, enter all option numbers that you want
complemented. Note that an option number set means something
different than one that is not set (refer to the explanation
below for the differences). If "N" for no is entered in response
to the "Wish to change?" query, all option numbers set, remain
so.

OPTION
NUMBER   DESCRIPTION

   0   When set, the following is allowed:

            o Tracks flagged spare or defective will not cause an
               error when the diagnostic is executed.

            o In Step 112, after five (5) consecutive attempts to
               verify are made, a track is flagged defective.

            When cleared, an error is reported for every defective
            and spare track.

   1-5   Reserved (not used)

6   When set, all cylinders to be tested are selected from
        the Cylinder table. This option is normally set in
        order to run short versions of the tests and test only
        the controller because only the cylinders found in the
        cylinder table are used.

        When cleared, all cylinders on the drive are tested plus
        the controller.

7-15   Reserved (not used)

2.38 (E)xit – Exit from Go I Operating Mode

Once you are satisfied with the setup of the extended features on
the diagnostic, enter "E" for exit and the program will continue.

2.4 USING THE DIAGNOSTIC AS A TROUBLESHOOTING AID

During the execution of the diagnostic, various information and
error messages are output by the diagnostic. These messages can
be used to determine the cause of errors, if they are interpreted
correctly. The following steps describe the procedures for using
the diagnostic as a troubleshooting aid:

a. Every line displayed by this diagnostic is significant. If a
    step number is given, turn to the explanation of that step
    which is given in section IV of this manual. In some cases it
    may be necessary to read a few steps preceding or following
    the step in order to get a full understanding of what is
    being done. Along with this explanation, several of the
    steps include probable causes as to what might be
    malfunctioning. These are given starting with the most
    likely assembly.

    Section V of this manual provides a summary of message that
    are used in more than one test. If you cannot locate a
    particular message, always refer to Section V.

                              NOTE

    It is important to look at each of the status bits in
    the status error message which gives the two status
    words obtained and the two values showing what they
    should be. Special attention should be focused on those
    bits which differ.

```
+------------------------------------------------------+ +---------+
|                                                      | | SECTION |
|                 TEST SECTION EXECUTION TIMES  |      | |---------|
|                                                      | |   III   |
+------------------------------------------------------+ +---------+
```

3.0 INTRODUCTION

The exact time for each test or for a group of tests is
determined by the following constraint:

o The optional operating mode, is variable depending upon the
   options chosen and the amount of looping requested.

o The standard operating mode only executes test sections 1, 3,
   4, and 5 for a total time of approximately 30 seconds.

Table 5-1, below, lists the execution times for diagnostic
initiation, test sections, and both the standard and optional
modes of operation.

Table 5-1. Test Section Execution Times

SECTION	STANDARD MODE	OPTIONAL MODE
Initiate	2.0 sec	2.0 sec
1	0.3 sec	0.3 sec
2	variable (interactive)	variable (interactive)
3	18.5 sec	18.5 sec
4/7925	36 sec plus 25 seconds per se-lected head	~ 1 hour (7920/7925 ~ 20 min/selctd head
(7906,7920)	35 seconds plus 21 sec/selected head	~ 1 hour 7906 ~ 10 min/ sel. head
5	0 seconds * * 35 sec on nth pass when n units present	0 seconds * *~ 1 hr on nth pass when n units present

```
+---------------------------------------------------+ +---------+
|                                                   | | SECTION |
|                  TEST SECTION DESCRIPTIONS        | |---------|
|                                                   | |   IV    |
+---------------------------------------------------+ +---------+
```

4.0 INTRODUCTION

The diagnostic program is literally divided into sections. Each
test section is divided into steps which describe tests performed
by the diagnostic.   A list of possible causes for errors that
occur during that particular step are given immediately following
each step.

This section is divided as follows:

  1. Initialization (diagnostic configuration)

  2. Start (CPU/GIC/HP-IB protocol checking)

  3. Test Section Discriptions for 1-5

The following paragraphs give a detailed description of each
area.

4.1 INITIALIZATION (DIAGNOSTIC CONFIGURATION)

When "D13037" is entered to begin diagnostic execution for the
first time, the diagnostic program outputs its title message and
prompt.

  79XX/13037 Disc Memory Diagnostic   X.X
  Install scratch disc packs in all units to be tested.
  Enter: GO to continue

The program next requests configuration information by outputting
the following:

  Enter Channel number to which the 13037 controller is
  connected (1-15)

If the value input is out of range, the requests are repeated.

Once the information has been input, the program passes control
to the Start phase of the diagnostic operation.

4.2 START

This phase is always executed when the program is first initiated
and performs the following:

  o Initializes all variables used in the diagnostic

o Handles GO 1 options (See Section 2 of this manual)

o Checks that the first unit listed in the unit table is present.

o Performs the following steps

STEP   DESCRIPTION

1   This step tests the IMB/channel command transfer by
           issuing a ROCL (roll call) command. When the
           resulting word is returned, the diagnostic verifies
           that the appropriate channel number bit is set to "1".
           If this step fails to get the expected response,the
           following usage and requests are output to the
           console.

           Non existent channel - Channels Present = XX
           Verify Thumbwheel setting on GIC,or run GIC diagnos-
           tic.

           Enter Channel number to which the disc controller is
           connected (1-15)

           You may want to enter the same channel number entered
           previously or another channel number, depending upon
           the cause of the error. Possible causes are:

           o The channel number entered and the actual
             thumbwheel setting are different. Verify your
             channel number entry; it must match the thumbwheel
             setting.

           o The channel specified does not respond to the ROCL
             command. Verify that the channel number entered
             was for a GIC; if so run the GIC diagnostic to
             verify the GIC board.

2   This step issues an INIT command to clear the channel
           specified. If the diagnostic is looping, it starts
           the next pass from this step.

3   This step verifies that the channel contacted is a GIC
           (in the instance where an ADCC channel number might
           have been inadvertantly entered). Register 14 of the
           GIC channel is read. Bit 0 must be equal to 0 (GIC
           has DMA capability) and bits 12 through 15 (channel
           identification) must be equal to zero (0). If this
           step fails, the following messages and request are
           output to the console:

Channel number entered is not a GIC Verify Channel
Number Entered, or run GIC diagnostic

Enter Channel number to which the disc controller is
connected (1-15)

The diagnostic returns to the Initialize phase and
waits for you to enter the GIC channel number after
outputting the above messages and request.

If the thumbwheel setting and the channel number you
entered were identical, the GIC Board is bad. Run the
GIC diagnostic to make sure and then replace the bad
GIC Board.
If this step passes, the following request is output
to the console:

Enter Device number assigned to the controller by the
HP-IB (0-7)

Enter the disc controller device number that controls
the unit(s) that are to be tested. If the value is out
of range the request is repeated.

4   The ability of the GIC to interrupt the CPU is tested.
        This step forces an interrupt by writing into register
        12. A channel program is then used to verify proper
        channel interrupt.

        Program execution is terminated if this step fails and
        one of the following error message is output to the
        console:
        Interrupts not working
                    or
        Channel program failure

        If this step fails, there is a good indication that
        the CPU has failed or that the GIC Board's TEST/OPER
        switch is in the TEST position. It should be in the
        OPER position. Run the Cold Load Self-Test to verify
        that the CPU is bad.

5   The device number, entered in step 3, is used to
        verify that the device specified is a 13037 Disc
        controller subsystem. To do this the diagnostic
        executes a channel program to identify the device
        entered in step 3. The identify logic obtains a value
        from the device which is then used to verify that
        device is a disc subsystem. If it is not, the
        following error message is output to the console:

        Identify failure. Word returned is XX; Should be !2
        Channel = x, Device = y - Verify System Configuration.
        The program then terminates. One other message can be

output if no value is obtained via the identify logic. In this instance, the following message and request is output to the console:

Disc subsystem does not respond - Verify system configuration for power,cable connection,thumbwheel settings.

The program then terminates.

If the device did not identify itself correctly, verify the following possible causes and perform the recommended corrective action.

o Subsystem power is off - verify and correct.

o Disc controller HP-IB address is configured wrong - verify channel and device number entered by executing the default mode of IOMAP.

o HP-IB cable connection between GIC and disc controller is not made - could be faulty cable or dirty edge connectors.

o Bad disc controller interface to HP-IB board - replace board.

7   Verifies HP-IB loopback operation

8   Clears attention on all drives connected to this controller and waits for all units to be ready, which is indicated by the two least significant bits of status word two from each unit.   Requests status from all possible units to determine the drive type and status of each drive.   The following message is output if the drive type changed:

WARNING: Drive type of unit X has changed

Unit X now identifies itself as a different drive type than unit X was previously.   To avoid errors, you should revise drive dependent parameters (unit and head tables) via the GO 1 operating mode before continuing.

Issues a seek to units that are ready and an address record to all other units in order to clear any seek status that the drives might have.

Unit selection is performed automatically on the first pass through the diagnostic.

Units which are ready
Unit - type
N 79xx
.   .
.   .
.   .
Do you want to write on all removable surfaces?

If "Y" is not input then the user is shown the unit table ane allowed to change it and then shown the head table and allowed to change it.

If the tables have been changed previously, the last message is changed to "do you want to write on selected surfaces?"

If execution was started by a RUN 1 or continued from a pause with GO 1, the diagnostic enters the GO 1 operating mode, displays the following, and allows you to change the area specified:

Enter: (U)nit,(?)errs,(H)ead,(O)utput,(L)ine Printer, (C)yl,(T)imes,(R)un Time Option,(E)xit

This dialogue is terminated when an (E)xit is entered to this message or the subsequent message "Wish to Change". This dialogue may occur following any test section.

If "-1" is entered for number of passes, the diagnostic will loop on the test sections indicated indefinitely. However, you are always given an opportunity to exit the program execution via the EXIT command, etc. (see Section II for more explanation).

11   This step checks that the current unit is present. Program terminates if an error occurs in the first pass. This following message is output whenever the selected unit X is not ready.

Unit X missing - Present A,B, (etc.)
                or
Unit X missing - Present none

Any other units found to be ready are shown. Possible problem areas are:

1.   Desired unit is not connected

2.   Device controller board is bad

3. I/O Sector board is bad

4. Micro processor board is bad

4.3 TEST SECTION DESCRIPTIONS

There are five test sections in the diagnostic. Test sections 1,3, 4, and 5 are automatically executed when the standard test is indicated. Test section 2 is the only test that is not executed in the standard mode of operation.

Via the TEST command in AID, each test section can be executed separately or in any combination you desire. Following is a description of each test section broken down into steps within that section.

TEST SECTION 1. This section of the diagnostic tests normal status and recalibrate.

STEP   DESCRIPTION

101   Gets and checks status for zero to ensure that a normal
         complete is returned and that no unexpected status
         conditions occurred.

102   Issues a recalibrate to the drive to ensure that the
         heads are positioned correctly. Check the following for
         possible cause of errors:

         a. Servo board

         b. Track follower board

TEST SECTION 2. This optional section executes the interactive tests that relate to pack addressing and tests the front panel switches on the drive. To run this section (in addition to the noninteractive sections) enter TEST +2 (or TEST ALL).

STEP   DESCRIPTION

103   Asks operator to move the drive FORMAT switch to the dot
         (it may already be at the dot). Checks to see that
         FORMAT bit (status word two, bit 10) is set. Check the
         following for possible cause of errors:

         a. Drive format switch

         b. Harness

         c. I/O sector board

         d. Data cable may not be properly attached if WAC fault
             results.

e. Preamplifier Board

f. Drive control board

g. Mother board

104   Writes one track of random data on cylinder zero.   Sets
       the protect bit in the preamble of each sector.   Reads
       sector 7 and compares it against the values written onto
       it.   Verifies all the sectors on the track.   Check the
       following for possible cause of errors:

WARNING

Since this is the first data transfer to be performed
(both read and write), and if an error occurs in this
step, further data transfers may also fail.   It is
recommended that you do not continue until you have
corrected the cause(s) for error in this step.

a. If 7906 and heads 0 and 1, and weak upper
    index pulse:

    (1) Transducer clearance too great

    (2) Transducer may be weak or faulty

    (3) I/O Sector board

b. If 7920/7925 or head 2 or 3 of 7906, could be bad
    track follower board.   To verify a bad track
    follower board, exit form this diagnostic program
    and run Sleuthsm (see the Sleuth Simulator manual).
    Enter and execute the following Sleuthsm program.

    5000 DEV 0,C,D,20,U   Define device
                                     characteristics

    5005 DB AA,138   Defines buffer
                                     for one full
                                     sector (includes
                                     preamble)

    5010 SEEK 0,X,2   Seek to logical
                                     unit 0 (defined
                                     by DEV)

    5015 RFS 0,AA(0)   Read full sector
                                     to obtain
                                     preamble
                                     information

5020 PRINT !AA(0);2;!AA(1);2;!AA(2) Display words
                                     0-2 of preamble

5025 GOTO 5010   Loop on program

5030 RUN

The expected result is: !80FE   CYL   HD/SEC

where !80FE = sync word, CYL = X, and the HD/SEC =
the head and sector addressed by the seek.   This
result should be the same on all lines of output.
To halt the program, interrupt with CNTL Y.   re-
enter the disc diagnostic again and specify the same
test that was specified when this error occured.

NOTE

If the program is interrupted before step 111 is reached
the protect bit is set for all sectors of a track at
cylinder zero.

105   Writes one track of random data on cylinder one with the
       defective cylinder bit in the preamble set. Attempts to
       read the track in step 107.

106   Asks operator to move the drive FORMAT switch away from
       the dot.   Checks to see that FORMAT switch bit (status
       word two, bit 10) is clear.   Bit 10 set in the status
       error message indicates the switch may defective. Check
       the drive FORMAT switch for possible cause of errors.

107   Reads last sector of cylinder one, which was written in
       step 105, and checks for the defective track indication
       in status word one. (Status word one is !310X, where X
       = unit number of the current drive).

108   Attempts to write on cylinder zero and checks status
       word one for an indication of an attempt to write on a
       protected track.   This track was flagged protected in
       step 104.   (Status word one is !560X, where X = unit
       number of current drive.)

109   Seeks to cylinder zero and issues an initialize data
       command to the controller.   Before executing the
       command, the controller checks to see that the drive
       FORMAT switch is at the dot and the drive PROTECT or the
       READ ONLY switch is away from the dot.   The FORMAT
       switch is away from the dot (operator should have

cleared it in step 106) and a status word two error
should be returned from the controller to the diagnostic
as the termination status in status word one. Any other
status will result in an error message being displayed.
Asks operator to move the drive FORMAT switch to the
dot. Writes random data on cylinder zero and checks
that the write attempt on the protected track with the
FORMAT switch at the dot is possible.

110   Initializes each track that is accessed by any of the
        heads in the head table for the unit specified. Each
        track initialized in this step is verified in step 112
        before the next track is initialized.

111   Verifies each track initialized in step 111.

112   Asks operator to place drive RUN/STOP switch in STOP
        position.

113   Checks the status word one for attention (!1F0X, X=unit
        number) and status word two from the drive for not ready
        and busy bits (bits 14 and 15, respectively) set.
        Issues a seek to check that the controller channel
        program ends immediately, if the disc drive is not
        ready. Check the I/O Sector board for possible cause of
        errors.

115   Asks operator to place RUN/STOP switch in the RUN
        position then waits for the device to interrupt.
        Requests status word two from the drive and checks to
        see if bit 12 is set, indicating the heads have been
        loaded. Check the Drive Control board for possible
        cause of errors.

116   Asks operator to move the drive READ ONLY switch to the
        dot, if the current unit is not a 7906 drive, otherwise
        move the UPPER/LOWER DISC PROTECT switch to the dot to
        correspond with the heads, which are in the head table
        for the unit specified. Steps 116 thru 119 are
        repeated, if all the heads are selected and if the unit
        specified is a 7906 drive. Check the following for
        possible cause of errors:

        a. Harness

        b. I/O Sector board

117   Performs a seek operation to cylinder zero and requests
        status from the drive. Checks for protect bit (status
        word two, bit 9) set. The protect bit should be set
        from step 116. The UPPER/LOWER DISC PROTECT switch on a
        7906, or the READ ONLY switch on a 7920 or 7925 is
        probable defective in case of errors.

118   Attempts to write data on the disc and checks that data
       was not transferred. Checks for status word two error.
       Asks operator to move the appropriate drive DISC PROTECT
       (7906) or READ ONLY (7920 or 7925) switch away from the
       dot. See Step 117 for possible cause of errors. Asks
       operator to remove power from the controller, then
       restore it. A write should not be allowed until a DSJ
       equal to 2 is obtained.

119   Sets file mask to cylinder mode and issures a CLEAR
       channel program instruction. If the CLEAR functions
       properly and the HP-IB interface thumbwheel is set to
       zero (0), the controller will be reset and the file mask
       in the controller will be set to zero indicating the
       drive is in surface mode rather than cylinder mode.
       This is checked by writing two sectors, starting at the
       last sector on a track, and checking the status for an
       end of cylinder indication (status word one is !C00).
       This test is repeated using device clear. Same results
       are expected. Errors during this step may be the result
       of the CPU thumbwheel not set to 0. Check the following
       for possible cause of errors:

       a. See that the CPU thumbwheel on the HP-IB Interface
           Board is set to 0. Only when the CPU thumbwheel is
           0 will the HP-IB interface issue a clear command to
           the controller.

       b. Device Controller board is bad.

TEST SECTION 3. This section of the diagnostic tests controller
functions by performing short writes and reads. The tests occur
in the following sequence:

STEP   DESCRIPTION

121   Seeks to cylinder zero. After the seek is complete,
       seeks to the last cylinder on the unit specified.
       Checks that the busy bit (status word two, bit 15) is
       set. (The busy bit should be set since the head
       positioner on the drive should still be seeking to the
       last cylinder.) Check the following for possible cause
       of errors, if the head positioner moves, but the busy
       bit is clear:

       Device Controller board, I/O Sector board, or, Drive
       Servo Board

       Check the following for possible cause of errors, if the
       arm does not move:

       I/O Sector board or Drive Servo board

122   Issues a recalibrate from the last cylinder. Check the
        following for possible cause of errors:

        a. Seek times too fast as set on the drive Servo board

        b. Track Follower board

        c. Drive Servo board

        d. A faint high pitched sound may indicate a faulty
            coil.

        e. Carriage (linear motor) problems

        f. Servo code on servo surface.

123   Verifies that the drive did detect a request to an
        illegal address by issuing a status command after each
        seek and checking that the seek check bit (status word
        two, bit 13) is set, since each seek constitutes an
        illegal disc address.   The following seek requests are
        issued, depending upon which device type is being
        tested.

TYPE	CYLINDERS	HEAD	SECTOR
7906	411, 414, 442, and 474	none	48
7920	823, 830, 886, and 950	5	48
7925	823, 830, 886, and 950	9	64

Check the drive Servo board for possible cause of errors if
the carriage moves and crashes .

        If the carriage does not move, check the following:

        a. I/O Sector board

        b. Error Correct board

        c. Device Controller board

        d. Microprocessor board

        e. HP-IB Interface Board

        f. GIC board

124   Seeks to the first entry in the cylinder table and
          writes one sector of random data at sector zero. This
          data is read in step 126.   This is the first write
          performed in the diagnostic if test section 2 is not
          selected.   Check the following for possible cause of
          errors:

          a. Cartridge/pack format error. Reformat using Sleuth.
              Note that you must first exit from this diagnostic
              to execute the following Sleuth program (see Sleuth
              Simulator manual). Also note that you may reformat
              using test section 2.

              5000 DEV 0,C,D,20,U
              5005 FMT 0
              RUN

          b. HD/SEC compare errors may indicate a defective data
              cable.

          c. WAC fault indicates the data cable is not connected
              or a defective head

          d. Drive control

          e. Preamplifier Board

125   Writes two sectors beginning at sector 7 of the same
          track as in step 124.

126   Reads one sector at sector zero and compares against
          data that was written in step 124.

127   Reads two sectors beginning at sector 7 and compares
          against data that was written in step 125.   The read
          without verify command is used.   (This step is skipped
          if this cylinder is flagged defective or spare and (R)un
          Time Option "0" is set.)   NOTE: The program skips to
          step 131 if heads 0 and 1 are not in the head table for
          the unit specified.

128   Issues a set file mask to the controller to select
          cylinder mode in the drive.   Write four sectors
          beginning at sector 45 (61 for 7925), head 0, and the
          first cylinder in the cylinder table. If a status check
          reveals an end of cylinder error, it is probable the set
          file mask command failed, or the file mask flip-flop in
          the controller, which designates cylinder mode, is not
          functioning properly. Check the Device Controller board
          for possible cause of errors.

          NOTE: The program skips step 129 if all heads for the
          unit specified are not in the head table.

129   Checks for an end of cylinder condition from the largest numbered sector (status word one, bits 4 and 5 set) starting at cylinder 0, head 0, sector 0. Depending upon which device type is being tested, the following sectors are verified for this condition.

DEVICE TYPE	SECTORS CHECKED
DEVICE TYPE	SECTORS CHECKED
7906	1, 2, 4, 8, 16, 32, 64, 128, 192, 193
7920	1, 2, 4, 8, 16, 32, 64, 128, 240, 241
7925	1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 576, and 577

Check the following for possible causes of error:

a. Device Controller board

b. Error Correct board

130   Reads the four sectors written in step 128 and compares the data read against that which was written.

131   Forces overrun on a write by not sending data when it is expected. Overrun is indicated by status word one, bits 4, 5, and 6 being set. Check the following for possible causes of error.

a. HP-IB Interface Board

b. GIC board

c. Device Controller board

NOTE: The program skips step 132 if the last head for the unit specified is not in the head table.

132   Write four sectors of random data beginning at sector 45 (61 for 7925) on the last head for the last cylinder. Checks status for an end of cylinder condition (status word one, bits 4 and 5 set).

133   Issues illegal commands to the controller. Checks status for an illegal opcode indication (status word one, bit 7 set). Check the following for possible cause of errors:

a. Device Controller board

      b. **Microprocessor board**

      c. **Error Correct board**

      d. **HP-IB Interface Board**

      e. **GIC board**

      NOTE: The program skips step 134 if the last head for the unit specified is not in the head table.

134   **Reads four sectors beginning at sector 45 (61 on 7925) on the last head for the cylinder. Checks status for an end of cylinder condition (status word one, bits 4 and 5 set). Data in the first three sectors is compared against that written in step 132.**

133   **Checks the first cylinder in the cylinder table and verifies that sector addresses on each sector of the track are correct for each head in the head table for the unit specified. Check the following for possible cause of errors:**

      a. **I/O Sector board**

      b. **Microprocessor board**

      NOTE: The program skips step 137 if heads 0 and 1 are not in the head table.

137   **Tests the time-out timer in the controller. The first part of the test sets the file mask in the controller to cylinder mode and waits one second. The controller will not poll after receiving the set file mask command. The diagnostic then reads two sectors beginning at the last sector of the first cylinder in the cylinder table and head 0 and then checks that a normal status is returned indicating the timer did not time-out and that the file mask was not reset.**

      The second part of the test sets the file mask in the controller to cylinder mode and waits 2.8 seconds. The controller will not poll after receiving the set file mask command and should time-out after the 2.8 seconds. The file mask in the controller is reset when the time-out occurs, which puts the controller back into surface mode. Following the wait, the diagnostic writes two sectors beginning at the last sector of the first cylinder in the cylinder table and checks that an end of cylinder condition is returned indicating the timer timed-out and the file mask was reset. Check the device controller for possible cause of error.

NOTES

The program skips to step 145 if the SPD (spare,protect, and defective) field of either of the first two entries in the cylinder table do not equal zero (0), or if the drive FORMAT switch is away from the dot.

The program skips to step 139 if the first two entries in the cylinder table are equal and the SPD field equals zero (0) and the FORMAT switch is at the dot.

138   Forces cylinder compare error. Performs a Read Full Sector at sector zero of the second entry in the cylinder table. Writes the full sector read onto sector zero of the first entry in the cylinder table. Reads sector one of the first entry in the cylinder table. Since the controller verifies the sector immediately preceding the sector to be read, a cylinder compare error (status word one, bits 5, 6, and 7 set) should occur (an incorrect cylinder address was placed into sector zero). The cylinder address in sector zero is corrected before proceeding to the next step. Check the following for possible cause of errors:

a. Device Controller board

b. Microprocessor board

139   Forces a head-sector compare error with a bad sector address. Performs a Read Full Sector at sector one of the first entry in the cylinder table. Writes the full sector read onto sector zero of the same cylinder. Reads sector one of the first entry in the cylinder table. Since the controller verifies the sector immediately preceding the sector to be read, a head-sector compare error (status word one, bits 4 and 7 set) should occur (an incorrect sector address was placed into sector zero). The sector address in sector zero is corrected before proceeding to the next step. Check the following for possible cause of errors:

a. Device Controller board

b. Microprocessor board

c. I/O Sector board

d. Track Follower board

NOTE: The program skips to step 141 if heads 0 and 1 are not in the head table.

140   Forces a head-sector compare error with a bad head
        address. Performs a Read Full Sector at sector zero,
        head one of the first entry in the cylinder table.
        Writes the full sector read onto sector zero, head zero
        of the first entry in the cylinder table. Reads sector
        one, head zero of the first entry in the cylinder table.
        Since the controller verifies the sector immediately
        preceding the sector to be read, a head-sector compare
        error (status word one, bits 4 and 7 set) should occur
        (an incorrect head number was placed into sector zero).
        The head number in sector zero is corrected before
        proceeding to the next step. Check the following for
        possible cause of errors:

        a. Drive preamp

        b. Drive Control board

        c. I/O Sector board

141   Forces a correctable data error. (The error correction
        circuitry will correct up to a 32-bit burst error (i.e.,
        the distance between any two bits in error is less than
        or equal to 32 bits) in each sector.) Reads two full
        sectors from the first entry in the cylinder table.
        Selects a random location N (0 < N < 255) in the buffer
        of the first of two consecutive words to be modified.
        Replaces these two words with two random
        words which are different from those originally there.
        Writes into the two sectors read from with a Write Full
        Sector of 278 words. Reads the same two sectors.
        Checks status for a possibly correctable data error
        (status word one, bits 4, 5, 6, and 7, set). Checks
        that the syndrome returned would correct the error.
        (The syndrome returns seven words from the error
        correction circuitry to correct the data error.) The
        sectors are corrected before proceeding to the next
        step. Check the following for possible cause of errors:

        a. Error Correct board

        b. Microprocessor board

142   Forces an uncorrectable data error (an error not
        correctable by the error correction circuitry). This
        step is basically the same as step 141 except four
        consecutive words in one of the two sectors are replaced
        by random words. Checks status for an uncorrectable
        data error (status word one, bit 4, set). Check the
        following for possible cause of errors:

        a. Error Correct board

        b. Microprocessor board

NOTE: The program skips to step 145 if the two tracks selected in step 143 are equal, or if the SPD field (preamble) of either track is not zero.

143   Tests sparing. Selects two random tracks A and B. Sets file mask to surface mode, no spare and no auto seek. Initializes track B defective with address A. Initializes track A spare with address B. Verifies track B (should be defective). Verifies track A (should get illegal access to spare track). Sets file mask to surface mode, sparing, and no auto seek. Writes a random sector through defective track B to spare track A. Reads back and checks. Sets file mask to surface mode, no spare, and no auto seek. Initializes track B (which removes the defective track). Reads from track A using false B address and checks data. Initializes track A (which removes the spare track). Check the following for possible cause of data errors:

    a. Device Controller board

    b. Microprocessor board

    c. Error Correct board

145   Forces unit unavailable status. Issues a seek to unit 11. Checks status for a unit unavailable status (status word one, bits 3, 5, 6, and 7, set). Check the micro processor board for possible causes of error.

NOTE: The program skips to step 147 if one of the following is true:

    o Cylinder zero on the selected head is defective,

    o Heads 0,1,2,3 for any other type drive are not in the head table.

146   Tests cold load read on unit zero. Selects a random cylinder from the drive, or if run time option 6 is set, selects a random cylinder from the cylinder table. Selects a random head (0-3) from the head table and selects a random sector. Reads from cylinder zero and the head and sector selected above. Seeks to the cylinder selected above. Issues a Cold Load Read to the selected head and sector. Compares the two buffers to check that data read is the same in both instances.

The Cold Load Read resets the file mask to spare enable,
incremental seek not allowed, and surface mode. Reads
two sectors beginning at the last sector of the track.
Checks for an end of cylinder condition (status word
one, bits 4 and 5, set). Check the following for
possible cause of errors:

a. Device Controller board

b. Microprocessor board

c. Error Correct board

NOTE: The program skips to step 148 if cylinder
table zero (0) does not equal cylinder table one (1)
minus 1.

147   Sets the file mask to surface mode, no spare, and
increment. Writes two sectors at sector 47, cylinder
table (0). Reads back and verifies. Reads second
sector only and verifies (only if table (0) is not
defective, or spare and (R)un Time Option "0" is set).
Sets file mask to surface mode, no spare, and decrement.
Writes two sectors beginning at sector 47, (63 for 7925)
cylinder table (1). Reads back and verifies. Reads
second sector only and verifies, only if cylinder table
(1) is not defective, or spare and (R)un Time Optiion
"0" is set). Check the following for possible cause of
errors:

a. Device Controller board

b. Microprocessor board

148   Checks the disc address on a random sector by issuing a
Request Sector Address after seeking to a random sector
and checking that the head sector address returned is
the same as that to which the seek was issued. Uses
reads with cylinder offset = 28, cylinder offset = -28.
For ACCESS NOT READY DURING DATA OPERATION or CYLINDER
COMPARE ERROR, check the following:

a. Servo board

b. Track follower

c. I/O Sector board

149   Check the following for possible cause if data errors
occur:

a. Current cartridge/pack previously formatted on
misaligned drive

b. Device Controller board

c. Preamplifier Board

d. Drive Control board

150   Writes eight sectors starting at sector zero, first
        cylinder in cylinder table using burst mode. Reads back
        same sectors using burst mode and checks data. (This
        step tests burst mode).

151   Sends all unused secondary HP-IB addresses to interface.
        Data which follows on a write should be ignored until
        EOT is sent. On read, one byte with EOT should be
        recieved.

152   Tests buffers on HP-IB Interface board by doing a
        loopback write of x bytes and doing a loopback read of
        512 bytes. X varies from 1 to 512.

        If errors occur replace the HP-IB Interface board.

153   Tests DSJ byte after seek (1), data transfer (0) and
        error (1).

TEST SECTION 4. This section fills the pack full of
checksummed data and then reads the data back to verify that the
correct sector has been read. This section verifies that no two
sectors are accessed from the same address and that no sector is
accessed from two different addresses. Each sector is
checksummed separately. The entire sector sums to zero. The
first two words sum to the cylinder number. The next two words
sum to the head/sector number. Each sum operation is actually an
exclusive or.

NOTE: The program uses all cylinders on the cartridge/pack if
(R)un Time Option "6" is clear and used the twelve cylinders in
the cylinder table if (R)un Time Option "6" is set.

STEP   DESCRIPTION

176   Seeks to the next cylinder according to (R)un Time
        Option "6". Generates a buffer of checksummed data,
        sixteen sectors in length. Writes the checksummed
        random data onto the entire track by writing sixteen
        sectors at a time.

        NOTE: Executes steps 179 and 180 only when the
        current cylinder also occurs in the cylinder table and
        the current head is head 2.

179   Reads sector zero with offset = +63 (decimal).   Checks
        status word two to ensure a drive fault did not occur.
        Check the track follower for possible cause of errors.

180   Read sector 33 with offset = -63 (decimal).   Checks
        status word two to ensure a drive fault did not occur.
        Check the track follower for possible cause of errors.

        NOTE: The program returns to step 176 until all
        heads in the head table have been selected.   It then
        selects a cylinder and returns to step 176 until all
        cylinders have been selected.

        NOTE: Step 182 is repeated 8192 times (64 times if
        run time option 6 is set).

182   Seeks to a random cylinder, head and sector according to
        (R)un Time Option "6".   Reads and verifies checksummed
        data of one sector for the cylinder, head, and sector
        written in Step 176.

        Check the following if more than one error concerning
        cylinder, head, and sector information occurs:

        a.   Servo

        b.   Track follower

        c.   Servo head lead

        d.   Inspect Servo code

        Check the following if data errors occur:

        a.   Device Controller board

        b.   HP-IB Interface Board

        c.   GIC board

In test section 4, AGC faults may be caused by the Track Follower
board or the Servo board. Seek check errors are likely to happen
here if the Servo board has heat sensitive counting problems. To
verify use a Sleuth program which performs a random seek (see
example in step 104).

TEST SECTION 5. This section runs multiple unit test if at
least two units are present and no errors occurred for any unit
in test section 4.

WARNING

This test section is not run until each unit under test has Been
tested in test section 4. A warning message is output if this
section is executed without first executing test section 4.

STEP   DESCRIPTION

185   Issues seeks to random locations on each drive present,
        one after another.   As the seek is completed, the data
        read is checked as in step 182 to ensure the checksums
        agree with those written in step 176. (See step 182 for
        selection of the random locations.)   Possible cause of
        errors is the RAM on the device controller.

```
+------------------------------------------------------+  +---------+
|                                                      |  | SECTION |
|                          ERROR/ACTION SUMMARY  |  |---------|
|                                                      |  |    V    |
+------------------------------------------------------+  +---------+
```

5.0 INTRODUCTION

Throughout this manual explanation for errors, their causes, and
possible action to be taken have been interspersed with the
appropriate test section description. The purpose of this
section is to summarize this information for easier reference.
The following table provides the error messages in alphabetical
order.

Table 5.1 Error Message Summary for Disc Diagnostics

```
|------------------------------------------------------------------|
|Buffer Checksum !X Cyl !X (Y) Hd/S !X (H=Y S=Y)                   |
|                                                                  |
|The checksum should be zero and the address in parentheses        |
|(decimal) should match the one displayed in the next message.     |
|Either the wrong sector was read or a data error occured.         |
|Possible causes of error are:                                     |
|                                                                  |
|  A.  Device Controller board                                     |
|  B.  Microprocessor board                                        |
|  C.  HP-IB Interface board                                       |
|                                                                  |
|                         NOTE                                     |
|                                                                  |
|  Each Sector is Checksummed separately.  The entire sector       |
|  sums to zero.  This hex sum is reported as the buffer check-    |
|  sum.                                                            |
|                                                                  |
|  The first two words sum to the cylinder number.  This hex       |
|  sum is reported as the Cyl.  In addition, the decimal equi-     |
|  valent is shown in parenthesis.  This equivalent is mean-       |
|  ingless if the sum is an invalid cylinder number.               |
|                                                                  |
|------------------------------------------------------------------|
```

13037B Disc Controller Diagnostic

Words two and three sum to the head/sector number. The head
is in the upper byte of the word and the sector is in the
lower byte. This hex sum is reported as the Hd/S. The deci-
mal equivalent for each is shown in parenthesis. These val-
ues may also be meaningless for an invalid head or an invalid
sector. Example: The following error report indicates a
data error occured in one of the first two words of the sec-
tor.

Error is Step 182: Current Operation is Read. See Section 5
Status is 0 0 0 01111 0000 0000 / C 000010 0 0 0 0 0 0 0 0 0
Should be 0 0 0 00000 0000 0000 / 0 000010 0 0 0 X 0 0 0 0 0
Buffer Checksum !10 Cyl !10 (16) Hds/S !0 (H=0 S=0)
Start 0/0/0-Last 0/0/0 Word Count 128 Old Cyl 64 Unit 0

Channel Program Timeout

Channel program did not interrupt within two seconds. Possible
causses or error are:

  a. Device Controller board
  b. Microprocessor board
  c. Error Correct board
  d. I/O Sector board
  e. Drive Control board
  f. Track Follower board
  g. GIC board (in system)
  h. HP-IB Interface board

Current Operation is Cold Load Read

Check the Error Correct board

Current Operation is Initialize Data

Possible causes of error are:
  a. Drive FORMAT switch not at dot
  b. Bad spot on media
  c. Device Controller board
  d. Microprocessor board
  e. Preamplifier board
  f. Drive Control board
  g. Data cable

Current Operation is Read.
Possible causes of error are:

  a. Device Controller board
  b. Error Correct board

```
|------------------------------------------------------------------|
|Current Operation is Read with offset=!Y                          |
|                                                                  |
|Current offset is !Y.  See Appendix A for offset format.          |
|Possible causes of error are:                                     |
|                                                                  |
|  a.   Device Controller board                                    |
|  b.   Track Follower board                                       |
|  c.   Heads                                                       |
|  d.   I/O Sector board                                           |
|------------------------------------------------------------------|
|Current Operation is Read Without Verify                          |
|                                                                  |
|Check the Microprocessor board                                    |
|------------------------------------------------------------------|
|Current Operation is Recalibrate                                  |
|                                                                  |
|Possible causes of error are:                                     |
|                                                                  |
|  a.   Microprocessor board                                       |
|  b.   Device Controller board                                    |
|------------------------------------------------------------------|
|Current Operation is Request Disc Address                         |
|                                                                  |
|Possible causes of error are:                                     |
|                                                                  |
|  a.   Microprocessor board                                       |
|  b.   Error Correct board                                        |
|------------------------------------------------------------------|
|Current Operation is Request Sector Address                       |
|                                                                  |
|Check the Microprocessor board                                    |
|------------------------------------------------------------------|
|Current Operation is Request Syndrome                             |
|                                                                  |
|Check the Error Correct or Microprocessor board                   |
|------------------------------------------------------------------|
|Current operation is Seek                                         |
|                                                                  |
|Possible causes of error are:                                     |
|                                                                  |
|  a.   Device Controller board                                    |
|  b.   Microprocessor board                                       |
|  c.   Check ribbon cable connection if seek executes with the    |
|       disc service unit, but not with the computer.              |
|------------------------------------------------------------------|
|Current Operation is Set File Mask                                |
|                                                                  |
|Possible causes of error are:                                     |
|                                                                  |
|  a.   Device Controller board                                    |
|  b.   Microprocessor board                                       |
|------------------------------------------------------------------|
```

13037B Disc Controller Diagnostic

```
|------------------------------------------------------------------|
|Current Operation is Verify                                       |
|                                                                  |
|Possible causes of error are:                                     |
|                                                                  |
|  a.  Preamplifier board                                          |
|  b.  Drive control board                                         |
|  c.  Heads                                                       |
|  d.  Data Cable attachment                                       |
|------------------------------------------------------------------|
|Current Operation is Write                                        |
|                                                                  |
|Possible causes of error are:                                     |
|                                                                  |
|  a.  Device Controller board                                     |
|  b.  Microprocessor board                                        |
|------------------------------------------------------------------|
|Cylinder X Head Y flagged defective                               |
|                                                                  |
|Track has been flagged defective at test section 2, step 112.    |
|------------------------------------------------------------------|
|Data Error Corected                                               |
|                                                                  |
|This message is displayed if the words read have been corrected   |
|by the syndrome.                                                  |
|------------------------------------------------------------------|
|Data Word X is Y should be Z                                      |
|                                                                  |
|The data returned on a read operation did not match the expec-    |
|ted data.  This message is displayed only for the first three     |
|erroneous words in the buffer.  Check the Device Controller       |
|board for possible cause of error.                                |
|------------------------------------------------------------------|
|Disc Address Error                                                |
|                                                                  |
|A Seek was issued to the controller.  This was folowed by a re-   |
|quest disc address command which returned a cylinder, head/       |
|sector combination different from that to which the seek was      |
|issued.  Value returned is shown as "last" in the next message    |
|output.  Possible causes of rerror are:                           |
|                                                                  |
|  a.  Mocroprocessor board                                        |
|  b.  Device Controller board                                     |
|------------------------------------------------------------------|
|Fatal Error                                                       |
|This error is sufficient to crash an operating system.           |
|------------------------------------------------------------------|
```

```
-------------------------------------------------------------------
|Start C/H/S=Last C/H/S Word cont X, Old Cyl, Unit X              |
|                                                              .  |
|C=cylinder, H=head, and S=sector.  Word count eauals number of  |
|sectors to be verified if current operation was a verify,       |
|otherwise it equals the number of words to be transferred be-   |
|tween the controller and the computer.  The old cylinder is the|
|cylinder below the heads before the last operation began.  This|
|message is added to most error reports.                         |
|---------------------------------------------------------------|
|Status is Z Z Z ZZZZZ ZZZZ ZZZZ / Z ZZZZZZ Z Z Z Z Z Z Z Z Z   |
|Should be Z Z Z ZZZZZ ZZZZ ZZZZ / Z ZZZZZZ Z Z Z Z Z Z Z Z Z   |
|                                                                |
|A status command dis not return the expected status.            |
|Each Z is replaced by 0, 1, or X (X=do not care). See Appendix |
|A for a description of status words.                            |
|---------------------------------------------------------------|
|X Words Transferred   Y expected                                |
|When the channel program interrupted, the data transfer was not|
|complete.  Possible causes of error are:                        |
|                                                                |
|  a.  If termination status is "Normal Complete" or "Overrun", |
|      check the GIC board or the HP-IB Interface.               |
|  b.  Drive Control board                                       |
|  c.  Preamplifier board                                        |
|  d.  I/O Sector board                                          |
|----------------------------------------------------------------|
```

```
+--------------------------------------------------+ +----------+
|                                                  | | APPENDIX |
|                      WORD FORMAT DEFINITIONS     | |----------|
|                                                  | |    A     |
+--------------------------------------------------+ +----------+
```

```
                       STATUS WORD ONE
    0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
  +---+---+---+-------------------+---------------+---------------+
  | S | P | D | T   S   T   A   T | X   X   X   X | U   N   I   T |
  +---+---+---+-------------------+---------------+---------------+
    |   |   |                   |              Not Used   Unit No. of
    |   |   |                   |                         current drive.
    |   |   |                   +-- *Encoded Termination status.
    |   |   |
    |   |   +--Track is defective if set.
    |   |
    |   +------Track is protected if set.
    |
    +----------Track is spare if set.
```

*Encoded Termination status is defined in the following table.

STATUS WORD ONE (hex)	TSTAT (binary)	DEFINITION (controller internal name)
0000	00000	No errors. (NORMAL COMPLETE)
0100	00001	Illegal opcode. (ILLEGAL OPCODE)
0200	00010	Unit available. (UNIT AVAILABLE)
0700	00111	Cylinder compare error. (CYL CMP ERR)
0800	01000	Uncorrectable data error. (UNCOR DATA ERR)
0900	01001	Head-sector compare error. (HD/SEC CMP ERR)
0A00	01010	I/O program error.
0C00	01100	End of cylinder. (END OF CYLINDER)
0E00	01110	Data overrun. (OVERRUN)
0F00	01111	Possible correctable data error.
1000	10000	Illegal access to spare track. (SPR TRK ACCESS)
1100	10001	Defective track. (DEFECTIVE TRK)

STATUS WORD ONE (hex)	TSTAT (binary)	DEFINITION (controller internal name)
1200	10010	Access not ready during data operation. (ACCSS NR DATOP)
1300	10011	Status word two error. (STATUS-2 ERROR)
1600	10110	Attempt to write on protexcted or defective track. (WRT PROTEC TRK)
1700	10111	Unit unavailable. (UNIT UNAVAIL)
1F00	11111	Drive attention. (DRIVE ATTENTION)

```
                          STATUS WORD TWO
  0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
+---+---------------------------+---+---+---+---+---+---+---+---*
| E | D   R   T   Y   P   E | X | A | P | F | DF| FS| SC| NR| B |
+---+---------------------------+---+---+---+---+---+---+---+---+
  |                             |   |   |   |   |   |   |   |   |
  |    Drive type*-+            |   |   |   |   |   |   |   |   |
  +---Error Flg**               |   |   |   |   |   |   |   |   |
                 Attention-------+   |   |   |   |   |   |   |   |
                 Protected-----------+   |   |   |   |   |   |   |
                 FORMAT switch (1=dot)---+   |   |   |   |   |   |
                 Drive fault------------------+   |   |   |   |
                 First status--------------------+   |   |   |
                 Seek check--------------------------+   |   |
                 Drive not ready-------------------------+   |
                 Drive busy----------------------------------+
```
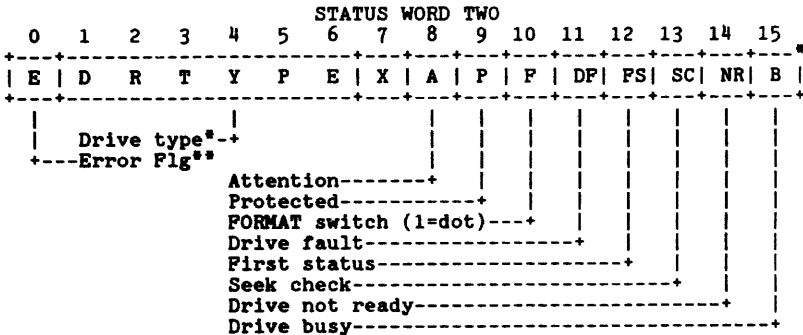
* Drive type is as follows:

```
  000000 = 7906
  000001 = 7920
  000011 = 7925
```
** Error flag - set if bit 11,13,14, or 15 is set.

```
                               OFFSET WORD
+-----------------------------+---+---+---+---+-----------------------+
| 0   1   2   3   4   5 | 6 | 7 | 8 | 9 |10  11  12  13  14  15|
+-----------------------------+---+---+---+---+-----------------------+
                                    |   |
                                    |   |
                                    |   |
                                    |   |
         Sign bit for _____|   |
         cylinder offset                |
                                        |
         Not used _____|


                                MASK
+-----------------------------------------------+---+---+---+--+
| 0   1   2   3   4   5   6   7   8   9  10  11 |12 |13 |14 |15|
+-----------------------------------------------+---+---+---+--+
                                                  |   |   |   |
       Incremental seek if clear and bit 15 = 1.__|   |   |   |
           Decremental seek if set and bit 15 = 1.    |   |   |
                                                      |   |   |
                          Allow track sparing if set._____|_  |   |
                                                          |   |
    Cylinder mode if set;  surface mode if clear._____|   |
                                                              |
                      Allow automatic seek if set._____|
```

```
+------------------------------------------------+ +----------+
|                                                | | APPENDIX |
|                              GLOSSARY OF TERMS | |----------|
|                                                | |    B     |
+------------------------------------------------+ +----------+
```

ABBREVIATION   DESCRIPTION
  OR ITEM
AGC   Automatic gain control.

CPU Thumbwheel   Thumbwheel on the HP-IB Interface Board which
                  identifies a CPU to the 13037 controller.
CRC   Cyclic redundancy check.

Cylinder   A given track on all heads constitutes a cylin-
                  der.

Cylinder Mode   The logical cylinder consists of all sectors on
                  all tracks at a given physical cylinder address.
                  When the end of a track is reached, the next
                  head is used (rather than the next cylinder as
                  in surface mode) unless the track is on the last
                  head.

DMA   Direct memory access.

ECC   Error correction code.

End Of Cylinder   A status condition caused by exceeding the end
                  of a cylinder during data transfer.

EOT   End of transmission bit.

Format   The act of initializing a disc cartridge/pack.
                  The name of a switch that allows initialization.

GIC   General I/O Channel

Head   One surface of a disc is written and read by one
                  head.

HP-IB   Hewlett-Packard Interface Bus.

Intitialize   A disc command which causes a disc track to be
                  written, without verifying the preceding sector.
                  This command is required to put address informa-
                  tion on a new disc cartridge/pack.

Lower Protect   On the 7906 disc, a switch used to protect the
Switch   non-removable portion of the the disc.

Read With Offset	A disc command which performs like a normal read except an offset in head position or clock timing is used.
Recalibrate	A disc command which causes the disc to slowly position the heads over cylinder zero.
Sector	The smallest addressable portion of a disc. One sector contains 256 bytes of user zero.
SPD	Spare, protected amd defective (status word one bits).
Surface Mode	The logical cylinder consists of all sectors on a given track. Data transfers are restricted to one surface.
Track	One surface at one physical cylinder address.
Track Follower	Board which serves head to track center.
Track Sparing	The disc controller will use an alternate track if the addressed track is marked defective and sparing is enabled. The spare track is expected to be marked spare with address matching the addressed track.
Unit	The number assogned to the disc drive (see front of the disc drive).
Upper Protect Switch	On the 7906 disc, a switch used to protect the disc cartridge.
Verify	A disc command which causes the disc controller to read from the currently addressed sector on the disc without passing data to the computer. At the end of each sector, a check is made for data errors.
WAC	Write without alternating current (drive fault LED).

HP 3000 Computer System

**7906/7920/7925 DISC VERIFIER
MANUAL**

LIST OF EFFECTIVE PAGES

The List of Effective Pages gives the date of the current edition and of any pages changed in updates to that edition. Within the manual, any page changed since the last edition is indicated by printing the date the changes were made on the bottom of the page. Changes are marked with a vertical bar in the margin. If an update is incorporated when an edition is reprinted, these bars are removed but the dates remain.

All pages original issue . . . . Mar 1982

PRINTING HISTORY

| **CONTENTS** |

SECTION I - GENERAL INFORMATION

Paragraph   **Page**

SECTION II - OPERATING INSTRUCTIONS

Paragraph   **Page**

```
+-------------------------------------------------------+ +---------+
|                                                       | | SECTION |
|                            GENERAL INFORMATION |  |---------|
|                                                       | |    I    |
+-------------------------------------------------------+ +---------+
```

1.0 INTRODUCTION

The 79XX Disc Verifier program is used to diagnose the 7906 7920,
and/or 7925 disc drives for proper operation. It executes
interactively with you to:

o   test all of the disc drive unit numbers

o   format and verify disc packs/cartridges, and verify that all
    the heads in the disc drive functional

o   check the track switching circuitry and verify the WRITE/READ
    circuitry with worst case and random data patterns while
    exercising the servo circuitry.

This program is written in HP AID. HP AID functions have been
created to make this program appear as similar as possible with
the verifier for HP 3000 Series III.

1.1 HARDWARE REQUIREMENTS

To successfully run this program requires that the console and
the "cold load" subsystem be operational. It is not possible to
execute the 79XX Disc Verifier program if you are unable to
perform a "cold load".

The following hardware is also required:

o   An HP 3000/33 or HP 300 system with at least 128K bytes of
    memory.
o   A disc subsystem which includes a 7906, 7920, and/or 7925 disc
    drive, 13037B controller, and an HP-IB Interface PCA.

o   A 7902A Flexible Disc Unit (cold load device to load program).

o   A scratch pack/cartridge (optional)

1.2 REQUIRED SOFTWARE

The only software required to execute this program is the
Diagnostic Utility System (DUS), AID, and the HP 7906, 7920, 7925
Disc Verifier program.

NOTE

If the flexible disc is bad you can create a new one via
the DUSCOPY utility program in MPE or, by using the
SADUTIL program in a stand-alone DUS environment.

1.3 MESSAGE AND PROMPTS

Two types of messages are output by this program; error and
information messages.

1.30 Error Messages

Error messages are used to inform the operator when the unit
under test fails to respond to a command or sequence of events
Errors may be fatal. In this instance the program aborts and
control is passed back to AID.

When the 79XX Disc Verifier program reports an error during
testing, the message generally includes the current device
status, the desired status, and other conditions as they were
when the error occured. To interpret the status' given, refer to
Appendix A where status words 1 and 2 are described.

1.31 Information Messages

Information messages are used to inform you of the progress of
the program or to instruct you to perform an operation. If an
associated pause occurs with a message (indicated by the prompt
character (>) being output to the console, you may input GO (to
continue execution of the program) or any other appropriate
command (refer to Section 2).

1.4 DIAGNOSTIC LIMITATIONS

The overhead associated with changing a randomized buffer in this
program is quite high. Due to the delay it adds in program
execution, the write data buffer (AA) is only modified three
times during the write/read test. The first pass will use a
preset worst case data pattern and the next three will use a
randomized buffer.

1.5 MINI-OPERATING INSTRUCTIONS

```
+=================================================================+
|1.   Responsibly shut down the system (SHUTDOWN).                |
|                                                                 |
|2.   Place console in REMOTE and reset the terminal.            |
|                                                                 |
|3.   Cold load the Diagnostic Utility System.                   |
|                                                                 |
|4.   Enter "VERIFIER".                                          |
|                                                                 |
|5.   Respond appropriately to all questions that follow.        |
|                                                                 |
| The program will execute the diagnostic testing as soon as the|
| last question is answered.                                     |
+=================================================================+
```

```
+----------------------------------------------------+ +---------+
|                                                    | | SECTION |
|                     OPERATING INSTRUCTIONS         | |---------|
|                                                    | |    II   |
+----------------------------------------------------+ +---------+
```

2.0 INTRODUCTION

This section gives detailed instructions for loading and
executing the 79XX Disc Verifier program. However, before
performing the described procedure in paragraph 2.2, be sure the
system is set up as described in paragraph 2.1.

2.1 DISC SUBSYSTEM SETUP

Setup the disc subsystem that is to be tested as follows:

o Place the disc drive READ ONLY switch in its OFF position.

o Place the disc drive FORMAT switch in its ON position if you
   intend to format the disc pack/cartridge.

o If you intend to use the customer's disc pack/cartridge, make
   a back up copy of the entire system using a 0 dump date in the
   SYSDUMP command.

o If you do not intend to use the customer's disc pack/cart-
   ridge insert a scratch pack/cartridge now or when you are
   instructed to do so by the program.

2.2 LOADING AND EXECUTING THE DISC VERIFIER

1. Shut down the system as follows if it is running:

    a. ABORT any sessions still logged-on.

    b. Use the RECALL command to check for outstanding alloca-
        tion messages.

    c. REPLY to all outstanding messages.

    d. SHUTDOWN the system.

Detailed instructions for performing steps a through d are given
in Section V of the Console Operator's Guide.

2. Reset the console by pressing the RESET
   TERMINAL key twice.

3. Insert the Diagnostic Utility System (DUS) flexible disc. or
   Set the COLD LOAD channel and switches to the channel and
   device number of the flexible disc.

4. Press the HALT, then RESET, then COLD LOAD keys on the front
   panel. The DUS program responds by displaying its title mess-
   age and prompt (:).

5. Enter "VERIFIER" in response to the prompt (:). The 79XX
   Disc Verifier program responds by displaying its title mess-
   age, the following request, and prompt (?):

   7906/20/25 Disc Verifier   Revision xx.xx Place Scratch
   Pack/Cartridge in Units to be Tested

   Enter Channel Number   (GIC channel number of 13037
   ?   controller)

   Enter Device Number   (Disc Unit Device # of 13037)
   ?

   Enter Unit Number   (Disc Drive to be tested)
   ?

   Enter the IMB Number for Channel # -if multiple IMB system
   ?

   Enter error count   (# of errors to display before
   ?   program ends)

6. Now testing begins with the first of a series of prompts.
   Respond according to your wishes to the following request.

   Unit Select Switch Test? (0=N, 1=Y)

   The Unit Select Switch Test is described in Section IV. If
   you plan to skip this test, enter "0". If you enter "1", the
   following message is printed:

   Unit Select Switch Test Expects 8 seperate Inputs
   Set Disc switch to unit #, Enter Unit # on console
   and press RETURN

   Set the switch on the disc drive to the next number and then
   enter that number at the console and press RETURN. If the
   test succeeds, the switch number prompt is repeated. If a
   test fails, you receive an appropriate message and then the
   prompt is repeated. In either case, once a total of eight
   unit numbers have been tested, program execution proceeds
   automatically to the format and verification sections.

NOTE

If a particular switch setting does not respond, a channel program SIOP time out occurs. The program still, however, requires eight numbers to test; so repeat numbers until eight have been tested. Note the position of the UNIT SELECT switch on the disc drive you plan to test. The number you type must be the same as the unit selected.

7. Before disc drive testing begins, you are given an opportunity to request disc pack/cartridge formatting.

    Format Pack? (0=N, 1=Y)

    To skip the formatting procedure, enter "0". Enter "1" to format. Note that formatting requires approximately three to ten minutes.

8. Next you are given an opportunity to request pack/cartridge verification to verify the quality of the media (disc pack).

    Verify Pack? (0=N, 1=Y)

    Enter "0" to skip verification. If "1" is entered, the following question is asked:

    Verify, Long Pass? (0=N, 1=Y)

    Enter "0" to perform the short pass. Enter "1" to perform the long pass. The long pass will write data onto the media and verify it three times.

10. The program now wants to know how many passes are to be executed by making the following request to the console:

    Enter the number of passes you wish to run. Each pass will take approx. 25-30 minutes depending on the disc being tested. A pass, without verifying the disc pack, will take approx 14-18 minutes.

    Enter the number of passes desired.

    The program will ask you whether you want to use the line printer for all messages from this point on.

        Send Output to Line Printer? (0=N, 1=Y)

11. Data errors detected during the verify portion of the test will be checked to determined if they are possible or unccorrectable data errors.

            Do you want to test disc pack for flagged tracks?
            (0=N,1=Y)

You have now supplied all of the information the Verifier needs.
Execution begins with disc formatting (if formatting was
requested) and continues with a short or long pass check of the
pack surfaces (if verification was requested). The next step is
to check the entire disc pack and provide a list of flagged
tracks. All tracks flagged defective or spare will not have any
errors reported during the remaining tests. The Verifier will
continue with a short or long pass check of the pack Write/Read
tests described in Section 4. The program tells you what is
happening with the following messages:

```
Begin Format                     (if formatting was requested)
End Format                                                 .

Begin Verify                     (if verifying was requested)
Verify Pass #X                   (short or long pass)
End Verify

Begin Main
End Head Test
End Track Switch Test
End W/R Test
```

When testing is complete, control of the system is returned to
DUS. Failures always cause a message to be output, and, if the
failure is fatal, the program will abort.

```
+-------------------------------------------------------+ +----------+
|                                                       | | SECTION  |
|                                    EXECUTION TIMES    | |----------|
|                                                       | |   III    |
+-------------------------------------------------------+ +----------+
```

The table below lists the approximate execution times for
diagnostic initiation of the test sections when run on a Series
30/33.

<div align="center">

Test Section Execution Times

</div>

	7906	7920	7925
FORMAT	3 min, 30 sec	4 min, 25 secs	10 min 1 sec
TEST FOR FLAGGED TRACKS	Approx. 5 to 8 min.		
VERIFY	4 min	2 min, 12 sec	4 min, 12 secs
VERIFY LONG	4 min	8 min,	14 min
MAIN TESTS	13 min,	13 min,	18 min,

\#   approximately 12 errors included in time estimate

```
+----------------------------------------------------+ +---------+
|                                                    | | SECTION |
|                            TEST DESCRIPTIONS       | |---------|
|                                                    | |   IV    |
+----------------------------------------------------+ +---------+
```

4.0 INTRODUCTION

The 79XX Disc Verifier has four principal sections of testing.

o The UNIT SELECT switch tests.

o The disc formatting utility

o The disc surface verifier

o The main write/read tests

4.1 UNIT SELECT SWITCH TEST

This test issues eight RECALIBRATE commands to verify that the
disc drive can respond to unit numbers 0 through 7. The test
prompts you to set the UNIT SELECT switch on the disc drive to
one of the unit numbers,and the enter that unit number. When you
press RETURN on the console, unit selection is verified.

NOTE that this test section does not keep track of which
unit numbers you test; only that you enter a unit number, any
number eight times.

4.2 DISC FORMAT UTILITY

This test section initializes and verifies the preamble, data
field, and postamble in the 48 sectors of each disc track;
including spare tracks.

A possible error that can occur during verification is as
follows:

   ID Error During Format

Should this error occur, take the following action:

1. Verify that the disc does not have a drive fault. If it
    does, refer to the appropriate disc service manual for
    further action to be taken.

2. Check the status one and two words displayed for probable
    cause of error. See Appendix A and B.

4.3 DISC SURFACE VERIFIER

This test verifies the data already written on the disc pack and
if selected, writes a worst case data pattern on the disc and
verifies.

You have two options to choose for this test (see section II,
Loading and Execution of Verifier).

Verify Pack: Verifies data on the entire disc pack, including
                    spare tracks. This test will allow a user to
                    verify his pack while maintaining the integrity
                    of his data.

Long Pass: Writes a worst case data pattern over the enti:
                    disc pack and then verifies it. Each succeeding
                    pass will circular shift each element in the
                    buffer before writing it on the disc pack.

A possible error message that can be output is as follows:

  Verify Error

The status displayed will indicate what the problem is. If the
status indicates that the error is possible correctable (status
word 1 (0F00) - see Appendix A), verify that this error can be
repeated (more than once) and flag the entire track defective and
reassign it. This can be accomplished with the Sleuth Simulator.

4.4 MAIN WRITE/READ TESTS

The write/read testing executes in three stages:

1.   The first test verifies that all heads can write and read
     random portions of the disc.

2.   The second test verifies that all heads can write and read
     with track switching. For example, 48 sectors of data are
     written to track 0, head 0, sector 45. After three sectors
     are written, the disc heads must be able to switch to track
     0, head 1, sector 0 in order to write the remaining 45
     sectors of data.

3. The third test randomly writes data, randomly seeks to another location, and returns to read and compare the data written during the write operation. Four different data patterns are used during this test.

Possible error message that can be output in this test section are as follows:

Read Data Error

Refer to status words one and two in Appendix A for cause.

Compare Buffer Error

The data received in the CPU's memory does not compare with the data transmitted to the disc.

```
+-------------------------------------------------------+ +----------+
|                                                       | | APPENDIX |
|                      STATUS AND CONTROL WORDS         | |----------|
|                                                       | |    A     |
+-------------------------------------------------------+ +----------+
```

STATUS WORD ONE

```
  0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
-------------------------------------------------------------------
| S | P | D | T   S   T   A   T   X | X   X   X | U   N   I   T |
-------------------------------------------------------------------
  |   |   |               |                         Unit Number Of
  |   |   |+----------+               Not Used      Current Drive
  |   |   ||
  |   |   |+-*Encoded termination status.
  |   |   |
  |   |   +-If set, track is defiective.
  |   |
  |   +-----If set, track is protective.
  |
  +---------If set, track is spare.
```

*Encoded termination status is defined in the following table

STATUS WORD ONE (hex)	TSTAT (binary)	DEFINITION (Controller Internal Name)
0000	00000	No errors. (NORMAL COMPLETE)
0100	00001	Illegal opcode. (ILLEGAL OPCODE)
0200	00010	Unit available. (UNIT AVAILABLE)
0700	00111	Cylinder compare error. (CYL CMP ERR)
0800	01000	Uncorrectable data error. (UNCOR DATA ERR)
0900	01001	Head-sector compare error. (HD/SEC CMP ERR)
0A00	01010	I/O program error.
0C00	01100	End of cylinder. (END OP CYLINDER)
0E00	01110	Data overrrun. (OVERRUN)
0F00	01111	Possible correctable data error.
1000	10000	Illegal access to spare track. (SPR TRK ACCESS)
1100	10001	Defective track. (DEFECTIVE TRK)

STATUS WORD ONE (hex)	TSTAT (binary)	DEFINITION (controller internal name)
1200	10010	Access not ready during data operation. (ACCSS NR DATOP)
1300	10011	Status word two error. (STATUS-2 ERROR)
1600	10110	Attempt to write on protected or defective track. (WRT PROTEC TRK)
1700	10111	Unit unavailable. (UNIT UNAVAIL)
1F00	11111	Drive attention. (DRIVE ATTNTION)

```
                          STATUS WORD TWO
   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
  ----------------------------------------------------------------
 | E | D   R   T   Y   P   E | X | A | P | F| DF| FS| SC| NR| B |
  ----------------------------------------------------------------
      Drive Type*                   |   |   |   |   |    |   |   |
      Error FLg**                   |   |   |   |   |    |   |   |
                  Attention --------+   |   |   |   |    |   |   |
                                        |   |   |   |    |   |   |
                  Protected ------------+   |   |   |    |   |   |
                                            |   |   |    |   |   |
                  FORMAT switch (1=dot)-----+   |   |    |   |   |
                                                |   |    |   |   |
                  Drive fault -----------------+   |    |   |   |
                                                    |    |   |   |
                  First Status --------------------+    |   |   |
                                                         |   |   |
                  Seek Check ---------------------------+   |   |
                                                             |   |
                  Drive not ready --------------------------+   |
                                                                 |
                  Drive busy ------------------------------------+
```
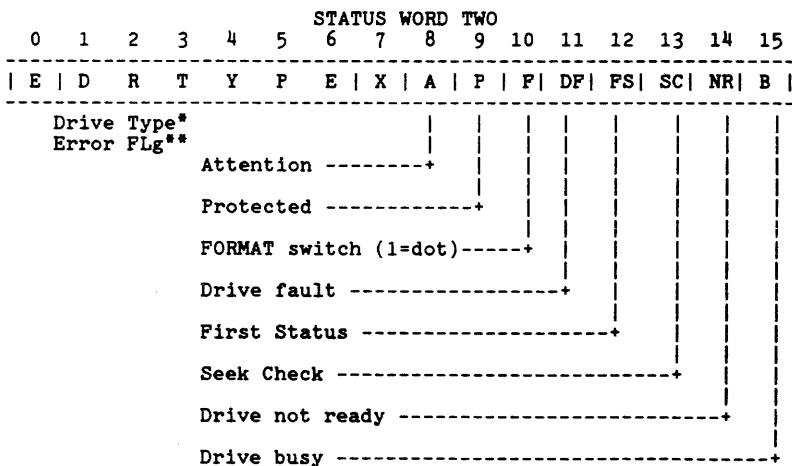
* Drive type is as follows:

```
  000000 = 7906
  000001 = 7920
  000010 = 7905
  000011 = 7925
```

** Error flag - set if bit 11,13,14, or 15 is set.

```
+------------------------------------------------+ +----------+
|                                              | | APPENDIX |
|                   CPVA INTERRUPT STATUS  | |----------|
|                                              | |    B     |
+------------------------------------------------+ +----------+
```

B.0 INTRODUCTION

Normal external interrupt requests are only generated for a
device by the Interrupt channel instruction, when the channel
program is running. However, interrupt requests will also be
generated for channel program aborts due to DMA transfer errors
or hardware or programmer errors. There are four causes of
interrupts, distinguished by the upper three bits of the first
four words of the CPVA.
Note that the first word of the CPVA is used for all channel
program aborts or halts not due to the Interrupt instruction.
Any interrupt instruction specifying that its code word be placed
in CPVA word 0 could have that word overwritten if the channel
program is aborted subsequently. It is safest to specify other
words than the first for Interrupt instructions, leaving it only
for abort messages.

B.1 INTERRUPT TYPE DEFINITION

The Interrupt types and formats of the corresponding Interrupt
Information Word are given below, with the value of bits 0-2
indicated beside the type name.

NOTE

This information will be displayed in a hex format
(i.e., E004). You must convert it to octal before
looking up appropriate bit in the following paragraphs.

100 - INTERRUPT INSTRUCTION:

(BITS 0,1, AND 2 OF CPVA 0) This instruction selects which CPVA
word is to be loaded with the 12 bits of data in the second word
of the instruciton. These bits are stored in the right most 12
bits of the CPVA word and bits 0-3 are set to 1000.

101 - HIOP DURING ACTIVE SERVICE:

(Bits 0,1, and 2 of CPVA 0) This format in CPVA word 0 is used if
an HIOP instruction was previously issued while the channel
program was running and not in the WAIT state. Refering to the
description of the HIOP that was issued, the program simply halts
without an interrupt, but if the program was being serviced, then
it will not halt unitl the next WAIT. In this case, an interrupt
will be generated when it finally halts because the HIOP
instruction had returned a condition code of "greater than",

telling software to expect an interrupt when the channel program halts. Bits 0-2 are set to 101 and bits 3-15 are cleared.

110 - DMA ABORT:

If a DMA transfer is aborted by memory errors or is aborted immediately and an interrupt request is generated, with the high order bits of channel register B stored right-justified in CPVA word 0, with bits 0-2 set to 110. The memory address retained by the channel (the suspected address of the error) will be preserved in words 4 and 5 of the CPVA for the device. The upper 8 bits (extended bits) will be right-justified in word 4, and the least significant 16 bits will be in word 5.

If DMA status (bits 5 and 6 of register B) is not correct for a particular channel instruction, although no system hardware error occurred, then the program will be aborted as described above, except that the DMA address will not be stored as above. Instead, it will be found in its usual place within the instruction. This method is used when no error is detected by the DMA hardware, but the DMA status does not conform with what is expected according to the channel program servicing algorithm.

The channel register bits in CPVA word 0 can be examined to determine which error occurred (see individual channel controller documents for description of register B).

111 - CHANNEL PROGRAM EXECUTION ERROR:

If an error is detected by the CPU during its servicing of a channel program (other than DMA aborts), then when the channel program is aborted, an interrupt request is generated and word 0 of the CPVA will contain an error code indicating why the program was aborted, with bits 0-2 set to 111. Bits 3-15 have the following definition:

Bit 3: Zero. No current usage.

Bit 4: Address rollover detected on Read or Write of single byte burst length which did not use DMA hardware to execute. This occurs if the transferred data byte was read frm or written to the last byte of any 64K word memory sector (address = xx FFFF).

Bit 5: HP-IB parity error on commands received into inbound buffer on channel controller (register 0). This can occur only if the channel controller is not Controller-in-charge of its HP-IB and detects an HP-IB command with even parity. It is a result of a check of register 2, bit 1 before execution of any channel instruction which issues HP-IB commands, or when the Interrupt Instruction tries to halt, so this should theoretically never occur.

Bit 6: Status change in the HP-IB interface on the channel controller during the channel program. This in indicated by register 2 bit 8 = 1, checked before execution of any channel instruction which must issue HP-IB commands, or if the Interrupt Instruction tries to halt. It occurs when the channel controller becomes or ceases to be active Controller-in-charge of the HP-IB, or switches between Remote and Local HP-IB states. Refer to the GIC portion of the Reference Training manual. This is generally a fatal condition for the channel, since it probably means that the GIC has lost the ability to control the HP-IB, and hence, I/O operations may not be possible until control is regained. This condition is cleared (but not rectified) when the channel program aborts.

Bit 7: Device Clear during channel program. This is indicated by register 2 bit 15 = 1, checked before execution of any channel instruction which must issue HP-IB commands, or if the Interrupt instruction tries to halt. It occurs when the HP-IB interface on the channel controller receives an HP-IB "Device Clear" (DCL) or "Selected Device Clear" (SDC) command when it is not Controller-in-charge of its HP-IB. It causes an abort since the condition is an indication that the channel controller has lost control of the HP-IB and has been told to reset itself as an HP-IB device. The condition is cleared when the channel program aborts.

Bit 8: FIFO handshake abort during channel program. This is indicated by register 2 bit 9 = 1, and is checked before execution of any channel instruction which must issue HP-IB commands, or if the Interrupt instruciton tries to halt. It occurs when an attempt is made to write to a full outbound HP-IB buffer or read from an empty inbound HP-IB buffer on the channel controller (register 0). Protection mechanisms exist in hardware and in the channel program service algorithms to avoid this occurrence. Execution of an instruction issuing HP-IB commands is not allowed to proceed unless it is known that there is enough room in the outbound FIFO to accomodate all that must go out to the HP-IB (except for the Write Register instruction), and logic on the GIC prevents access to the inbound buffer unless there is a byte available. So it should only be exceptional conditions which cause this abort.

Bit 9: Serial Poll error. The SRQ line on the HP-IB was causing assertion of SCRQ, but the serial poll table length in word 6 of the CPVA was zero, or a polled device did not respond (in which case bit 13 will also be set). The table length abort will be issued for device 0 on the channel, and the timeout abort will be issued for the device number represented by the low

three bits of the primary address field in the serial poll table entry for the device being polled. The channel program service routine will remove the channel from the serial poll state when the abort occurs. Since this is a fatal channel error, the channel controller will be disabled from asserting CSRQ for nay device requests. It is up to software to re-enable the channel controller (at least the GIC) by writing register F with bit 8 = 0, presumably after correcting the error in the serial poll table. Note that for the GIC, if it is left in the state described after the abort, and CSRQ is not issured within the hardware timeout interval, then a channel program timeout abort interrupt will occur for the same device number, after which time the GIC will again be able to recognize device requests and assert CSRQ for them.

Bit 10: Illegal CSRQ. The channel controller asserted CSRQ, but the channel program status in DRT entry word 3 for the device was not appropriate to the reason for the request given in the channel's OBSI data word when the channel was queried. For example, if the OBSI word indicated that a parallel poll or SIOP or HIOP request was the cause of CSRQ, but DRT entry word 3 indicated that the channel program was suspended waiting for termination of a DMA transfer, then that is a conflict of request indicators, and would cause this abort.

Bit 11: Memory Parity Error during channel program service. This bit indicates that a memory parity error occurred sometime during channel program service of this channel program, probably during execution of the channel instruction pointed to by DRT entry word 0. Memory parity errors occurring during DMA transfers are handled separately as DMA abort interrupts.

Bit 12: Non-responding IMB module during channel program service. This bit indicates that a referenced IMB module (memory or I/O) did not respond within a reasonable period. The facility described in the IMB section of the Reference Traning Manual is used by the CPU for this error. Bit 12 and this abort format is used only by channels executing their own channel programs, which do not cause CPU traps when IMB timeouts occur.

Bit 13: Channel Hardware Timeout. This bit indicates that the
channel program had been suspended to wait for DMA to
complete or for data to arrive from the device, or for
a selected condition on the channel controller to
occur, and the channel controller did not assert CSRQ
for   that   condition   within   its   hardware   timeout
interval. Word 0 of the DRT entry will be the address
of the instruction in which the timeout occurred. Some
channels might not have such timeout logic. The GIC
has a 1 second timeout.

Bit 14: Data Chain Error.   This bit indicates that an errot
occurred during data chain calculations in a Read,
Write, Read Control, or Write Control, or Execute DMA
instruction.   It includes detecting a byte count which
is zero with no data chaining specified.

Bit 15: Invalid Instruction. The instruction word fetched was
an unimplemented channel instruction, or was not a
Read, Write, Read Control, Write Control, or Execute
DMA instruction when coming out of a channel program
suspend waiting for DMA to complete (a highly unlikely
situation, unless software is influencing the channel
program or DRT entry during channel program execution).
Note the most likely causes of this abort are software
setting some incorrect word in the channel program or
incorrectly computing the starting address of the
channel program proir to the SIOP instruction.

HP 3000 COMPUTER SYSTEM

HP 7976A MAGNETIC TAPE UNIT
DIAGNOSTIC LOADER

LIST OF EFFECTIVE PAGES

The List of Effective Pages gives the date of the current edition and of any pages changed in updates to that edition. Within the manual, any page changed since the last edition is indicated by printing the date the changes were made on the bottom of the page. Changes are marked with a vertical bar in the margin. If an update is incorporated when an edition is reprinted, these bars are removed but the dates remain.

All pages original issue . . . . Mar 1982

PRINTING HISTORY

```
-------------------------------------------------------------------
| CONTENTS                                                        |
-------------------------------------------------------------------
```

```
+----------------------------------------------------+  +---------+
|                                                    |  | SECTION |
|                          GENERAL INFORMATION |  |---------|
|                                                    |  |    I    |
+----------------------------------------------------+  +---------+
```

1.0 INTRODUCTION

The HP 7976A Magnetic Tape Unit Diagnostic Loader provides the
capability of testing an HP 7976A Tape Unit ONLINE from the sys-
tem console or a remote maintenance terminal. The user must have
system manager (SM) or system supervisor (OP) capability to run
this program. The diagnostic loader will run under MPE on any HP
3000 computer system that has an HP 7976A connected to it.

The diagnostic loader will load, execute, and report the results
of 88 diagnostic routines, HP-IB loopback test, and selftest.
Installation of the diagnostic loader will occur as part of the
normal IT installation procedure.

The diagnostic loader will automatically locate the HP 7976A from
the I/O configuration tables if only one is connected to the sys-
tem. If there are two or more HP 7976A tape units connected to
the system, the user will receive a message that lists all the
HP 7976A's by logical device number and then asks which one the
user wants to test.

1.1 HARDWARE REQUIREMENTS

An HP 3000 computer system with an HP 7976A Magnetic Tape Unit
connected to it.

1.2 SOFTWARE REQUIREMENTS

The following files must be resident in the HP32340 group of the
SUPPORT account:

PD470A	MONLIST	BASMON	BUFSUB	CNFG
CNVR	DMSSUB	DSTATR	EOTSUB	EROR2
EROR3	EXECUTE	ERRTAB	INTABL	INTRP
INTRSB	INIT	MONITR	OPCDE0	OPCDE1
OPCDE2	OPCDE3	OPCDE4	OPCDE5	OPCDE7
PRCMBF	PRNT	PHDR	REGFIX	RDREG
RDSUB	REGTBL	RGT	RUNCMD	SETSNS
SETSTA	WRTSUB	DIAPAT	FLIP	XEQ
XEQCE				

1-1

PD470A is the loader program file.
MONLIST is the monitor file directory.
BASMON thru FLIP are monitor files.
XEQ and XEQCE are the execution files.

1.3 LIMITATIONS

None

```
+--------------------------------------------------------+ +---------+
|                                                        | | SECTION |
|                               OPERATING INSTRUCTIONS   | |---------|
|                                                        | |   II    |
+--------------------------------------------------------+ +---------+
```

2.0 INTRODUCTION

When the HP 7976A Diagnostic Loader is run it will locate the
HP 7976A Magnetic Tape Unit, request a tape mount, and perform
the opertions specified in the Execution (XEQ) file of the Diag-
nostic Loader.

2.1 OPERATION

The HP 7976A Diagnostic Loader (hereafter referred to as the
Loader) may be run in either Auto or Manual mode. To run the
Loader type the following:

    :HELLO FIELD.SUPPORT,HP32340
    :RUN PD470A
        or
    :RUN PD470A,MANUAL

If the Loader is run in Auto mode, minimal user interaction is
necessary. In Manual mode the Loader prompts the user for the
desired operation:

    Routine (RTssrree), Selftest, Loopback, Auto, Exit?

        Where:
                ss is the section designator in OCTAL
                rr is the routine designator in OCTAL
                ee is the routine extension field in OCTAL

Refer to HP 7976A Subsystem Diagnostic Manual (p/n 07976-90906)
for test routine descriptions, error codes and messages.

2.1.1 Routines

Routines are executed in the 7976 under the supervision of a
Monitor. The Monitor is comprised of a number of individual
files, with their names listed in the file MONLIST. When the
Loader needs to download the Monitor, it reads the names out of
MONLIST. If MONLIST or a Monitor file cannot be found, the situ-
ation is reported:

> Monitor directory "MONLIST" is unavailable.
>> or
> Monitor file (filename) is unavailable.

The above are irrecoverable errors.

2.1.2 Selftest

The 7976 internal Selftest runs identically whether it is initi-
ated by the Loader, the Selftest pushbutton, or at power-on.
However, under the Loader the result is a code indicating passed,
or describing the error that occurred. On error, the Loader re-
ports the code and generates a message corresponding to it. For
more information on error codes or messages, refer to the HP 7976
Operators Manual (p/n 07976-90901).

2.1.3 Loopback

The Loopback command performs the traditional 256 byte HP-IB
loopback (Data bytes %377, 0, 1, 2, 3, ...,%376). If less than
256 bytes are returned, the user is informed:

> Loopback failed. Only xxx bytes returned out of 256.

otherwise, received bytes are checked. If any are in error, they
are printed, along with what value they should have, and the ex-
clusive OR of the expected and received bytes:

```
Loopback failed.
Byte   0 is %375, should be %377; exclusive or=%002
Byte   3 is %000, should be %002; exclusive or=%002
Byte   4 is %001, should be %003; exclusive or=%002
```

If more than 32 bytes are in error, the remaining ones are not
printed, and the user will see the message:

> Remaining bytes not checked.

2.2 EXECUTION (XEQ) FILES

The Execution file will specify which operation (routine, self-test,loopback) to execute, what to do next if it passes, and what to do if it fails. The XEQ file is an EDITOR file and may be modified by the user or the user can create one or more XEQ files using EDITOR. The following is an example of a line in an XEQ file:

    23 RT111002   24   124   Tach symmetry test

Where:
        23 is the EDITOR file line number.
        RT111002 is the routine name.
        24 is the EDITOR file line number to branch to if RT111002 passes.
        124 is The EDITOR file line number to branch to if RT111002 fails.
        Tach symmetry test is a short comment telling what type of test RT111002 is.

Whole lines in an XEQ file may be comments if the first character is a period (.) or the letter C. Using lines of comments will allow XEQ files to be easily documented and identified by the user. An XEQ file must be a standard 80 byte ASCII EDITOR file. The following is an example of a short XEQ file:

```
 1    C                7976 XEQ file version X.00      5/5/81
 2    C
 3    C Loop back and selftest are done before any routines.
 4    C Any failure will invoke manual mode.
 5    C
 6    Command      Pass      Fail      Comment
 7    C------      ----      ----      -------
 8    Loopback     9         18
 9    Selftest     10        18
10    RT020100     11        18        Amp sensors pick and drop
11    RT020200     12        18        Amp sensors decode
12    RT020300     13        18        PE write ones
13    RT020400     14        18        PE write zeros
14    RT020500     15        18        PE write zeros-p track
15    RT020600     16        18        GCR write ones lwr
16    RT021001     17        18        PE read 20% phase error
17    RT021101     19        18        PE write 20% early error
18    Manual
19    Exit
```

Only the first character of an XEQ command is significant and it
may be either upper or lower case. Before an XEQ file is used,
it will be checked for syntax errors and that all referenced line
numbers exist. This is done to prevent the case where the Loader
must abort partway through a run because a problem is only then
discovered.   The offending line will be printed, the field in
question will be identified with a pointer ( ), and followed by
an error message.

Two execution files are furnished as part of the diagnostic, XEQ
and XEQCE.   File XEQ is the standard execution file, which per-
forms basic hardware tests of the tape drive, and should be used
for normal user tests. File XEQ is selected automatically by the
loader, so no special action is required.

File XEQCE is a more thorough but time consuming test of the HP
7976, intended for the use of HP service personnel. XEQCE in-
cludes all tests done by XEQ plus others.   Some tests are
"MARGIN" tests, which check fine adjustments of the tape drive.
Special equipment is required to measure the state of the cir-
cuits and perform the adjustments. Failures indicated by XEQCE
do not necessarily indicate a failure in the HP 7976, but provide
information to HP service personnel as to what service the tape
drive requires at the next preventive maintenance (PM) call.

To specify that the execution file XEQCE is to be used by the
loader, enter the following:

        :FILE XEQ=XEQCE
        :RUN PD470A

To run a user created XEQ file, a file equation (with the users
filename for it) must be used prior to the ":RUN PD470A" command
such as:

        :FILE XEQ=NEWXEQ
        :RUN PD470A

To get a hardcopy listing of the Loader output, use the following
file equation prior to the RUN command:

        :FILE LIST;DEV=LP

2.3 AUTO AND MANUAL MODES

The loader may be run in either Auto or Manual mode and the mode
may be changed. Auto is the default mode when the ":RUN PD470A"
command is used and the XEQ file will control Loader operations.
To run in Manual mode the "RUN PD470A,MANUAL" command should be
used.   To enter Manual mode when running in Auto, simply type a
Control-Y.   To enter Auto mode when running in Manual, type the
word Auto or A when prompted by the following Loader message:

Routine (RTssrree), Selftest, Loopback, Auto, Exit?

Manual mode will also be entered if the XEQ command "Manual" is encountered in the XEQ file.

2.4 ERROR MESSAGES

Any of the following errors will prohibit further operation. The Loader will print "XEQ file has error(s), can't continue", and terminate. The error messages are:
Read of XEQ record xxx failed, can't continue. Error is:
(followed by a file system error message)

XEQ record xxx <>80 bytes---illegal format. Must be standard Editor file, kept numbered.

Illegal Editor line number in record xxx. Must be standard Editor file, kept numbered.

Illegal Routine name. Must be "RT" followed by six digit octal number.

Illegal or missing branch line number.

Command must be RTssrree (Routine), Loopback, Selftest, Manual, Comment, or Exit.
(unknown command)

Can't find this line.
(A Routine, Loopback, or Selftest's "branch" number referenced a line that does not exist).

Too many records in XEQ file - 305 maximum.

More than 16 errors in XEQ. I give up.

It is possible, and allowed, that not all Routines referenced in the XEQ file are actually on disc. The Loader will print:

Warning: RTssrree, ..., RTssrree are not on disc. Execution will stop when one is needed.

The Loader will continue until that point.

HP 3000 Computer System

HP 2680A PAGE PRINTER VERIFER
DIAGNOSTIC MANUAL

ii

The List of Effective Pages gives the date of the current edition and of any pages changed in updates to that edition. Within the manual, any page changed since the last edition is indicated by printing the date the changes were made on the bottom of the page. Changes are marked with a vertical bar in the margin. If an update is incorporated when an edition is reprinted, these bars are removed but the dates remain.

All pages original issue . . . . Mar 1982

```
----------------------------------------------------------------------
|  .
|                                                          CONTENTS  |
----------------------------------------------------------------------
```

SECTION I - GENERAL INFORMATION

Paragraph
                                                              Page

SECTION II - OPERATING INSTRUCTIONS

Paragraph
                                                              Page

SECTION III - EXECUTION TIMES
Paragraph
                                                              Page

SECTION IV - TEST DESCRIPTIONS

Title   Page

1.0 INTRODUCTION

The 2680A Page Printer Verifier is an online diagnostic program designed to perform the following functions:

o Verifies 2680A configuration.

o Checks HP-IB interface between the 2680A and the HP 3000.

o Commands the HP 3000 to run printer diagnostic tests.

o Verifies 2680A ability to detect invalid operations generated by the host HP 3000.

o Verifies printer electro-mechanical operation by printing test patterns and forms to be checked by the user.

1.1 REQUIRED HARDWARE

The page printer verifier operates on any HP 3000 HP-IB system.

1.2 REQUIRED SOFTWARE

The Page Printer Verifier is written in Systems Programming Language (SPL) and is identified as PD467A. Program PD467A will reside in the HP32340 group of the Field.Support account.   The verifier consists of three separate files describe as follows:

SOFTWARE	DESCRIPTION
Program File PD467A	Diagnostic Program
Environment File D467ENV	Contains character sets, vertical form controls, and forms used by the verifier.
Message Set (22)	Resides in the system message catalog. Contains majority of messages used by the verifier.

1-1

1.3 LIMITATIONS

The verifier invokes part of the 2680A microdiagnostics by issuing the self test command. The selftest function tests the printer memory and data control system (DCS). The host system, when invoking self test, cannot exercise the printing capabilities of the page printer.

If the user wants to run the entire set of 2680A microdiagnostics, and thereby test all printer functions, the user must use the 2680A keyboard to issue the appropriate command.

The verifier will test the remaining printer functions, that are not covered by self test, by issuing data, forms, and VFC's.

To maintain system integrity, the verifier should be used by the system manager or system operator.

1.4 MINI-OPERATING INSTRUCTIONS

```
+=====================================================================+
|                                                                     |
|    1.  Verify proper online operation.                              |
|                                                                     |
|    2.  Enter the following system commands:                         |
|                                                                     |
|         :HELLO FIELD.SUPPPORT,HP32340                               |
|         :RUN PD467A                          .                      |
|                                                                     |
|    3.  Perform procedures requested by the verifier.                |
|                                                                     |
|                             NOTE                                    |
|                                                                     |
|         Use the printer self-test function (on top panel            |
|         keyboard) to run the complete set of printer                |
|         diagnostics.                                                |
|                                                                     |
|    4.  To run printer self test, enter the following commands       |
|        from the printer keyboard:                                   |
|                                                                     |
|        a. Press HALT                                                |
|        b. Enter 1 ENT.                                              |
|        c. Press RUN.                                                |
|                                                                     |
|                                                                     |
+=====================================================================+
```

1.5 LIST OF ABBREVIATIONS

The following abbreviations are used in this manual.

```
-----------------------------------------------------------
| ABBREVIATION |            DESCRIPTION                    |
|--------------|------------------------------------------|
|              |                                          |
|    CPVA      |  Channel Program Variable Area           |
|              |                                          |
|    DCS       |  Data Control System                     |
|              |                                          |
|    ENT       |  Enter                                    |
|              |                                          |
|    HPIB      |  Hewlett Packard Interface Bus           |
|              |                                          |
|    GIC       |  General I/O Channel                     |
|              |                                          |
|    IMB       |  Inter Module Bus                        |
|              |                                          |
|    IMBA      |  Inter Module Bus Adaptor                |
|              |                                          |
|    IOP       |  Input Output Processor Bus              |
|              |                                          |
|    LDEV      |  Logical Device                          |
|              |                                          |
|    LDT       |  Logical Device Table                    |
|              |                                          |
|    LPDT      |  Logical Physical Device Table           |
|              |                                          |
|    LPT       |  Logical Page Table                      |
|              |                                          |
|    MCS       |  Machine Control System                  |
|              |                                          |
|    OP        |  Operator                                |
|              |                                          |
|    PCB       |  Port Controller Bus                     |
|              |                                          |
|    PHI       |  Single Chip HPIB Interface              |
|              |  (Processor to HPIB Interface)           |
|              |                                          |
|    SM        |  System Manager                          |
|              |                                          |
|    SPL       |  Systems Programming Language            |
|              |                                          |
|    VFC       |  Vertical Forms Control                  |
|              |                                          |
-----------------------------------------------------------
```

```
+------------------------------------------------------+ +---------+
|                                                      | | SECTION |
|                        OPERATING INSTRUCTIONS | |---------|
|                                                      | |   II    |
+------------------------------------------------------+ +---------+
```

A2.0 INTRODUCTION

The Page Printer Verifier is an online verifier designed to test the operation of the 2680A Page Printer. The verifier is used by the system manager or operator. The user calls the program and responds to prompts issued by the verifier. The verifier first verifies the configuration and communication links. The user is then asked to select either all or individual tests.

Refer to section IV for a detailed description of each test section of the diagnostic process.

2.1 STANDARD MODE OF OPERATION

1. Verify proper system operation.

2. Enter the following commands to call the verifier:

```
:HELLO FIELD.SUPPORT,HP32340
:RUN PD467A
```

NOTE

When the verifier starts to run, the break key on the user terminal will be disabled and the control Y break enabled. This will allow the verifier to restore the original condition of the logical device under test if the verifier is to be aborted. If the verifier is called from a stream job, the disable break and enable control Y will not be implemented.

NOTE

If more than one 2680A Printer is installed in the system, the user will be asked to select the appropriate unit. If the printer is not installed, the verifier will notify the user and terminate. The verifier will also inform the user if the printer is offline.

3. A display similar to the following will be observed:

```
PD467A PAGE PRINTER VERIFIER  REVISION 0.00
```

2680 Page Printer Verifier

TERMINAL BREAK DISABLED, CONTROL Y ENABLED BY THE VERIFIER
(D467MSG 536)
THE VERIFIER WILL STOP SPOOLING TO THE 2680A PAGE PRINTER
if spooling was enabled (D467MSG 8)

PRESENT 2680A ENVIRONMENT

2680A MEMORY SIZE = 256K WORDS   DCS FIRMWARE DATECODE=2114
NUMBER OF VFC`S LOADED = 1   NUMBER OF FORMS LOADED = 0
PAGE LENGTH = 8.50 INCHES (21.59 CM)PAGE WIDTH = 11.0
                                    INCHES(29.94 CM)
NUMBER OF CHARACTER SETS LOADED=1

4.   Section 1 tests will be run and when completed, the
     following message is displayed.

TEST SECTION 1 (CONFIG/BASIC COMM TEST) COMPLETE (D467MSG 65)
DO YOU WANT TO RUN INDIVIDUAL SECTIONS OF THE VERIFIER (Y/N)?
(D467MSG 500)

5.   The user is asked if individual section tests are desired:

                         NOTE

     After section 1 has completed, the user will be asked
     to run section   through 5 or   select individual
     sections.   If the user selects   the   individual
     sections,   the program will execute the selected
     sections and allow the user to reselect sections, or
     exit the program.

6.   If individual sections are reqested the following menu is
     displayed:

SECTIONS AVAILABLE TO RUN ARE:

     SECTIONS 2 (LOOPBACK)
     SECTIONS 3 (SELFTEST)
     SECTION 4 (STATUS TESTS)
     SECTION 5 (DATA TESTS)
     SECTION 6 (INTERACTIVE TESTS)
     ENTER   "E" TO EXIT
ENTER SECTION NUMBER(S) (I.E. 2,3,E)
3,5,E

7.   The user is asked if a given section is to be repeated
     (loop): If N is selected the following data is displayed:
DO YOU WANT TO LOOP ON SECTIONS(S) (Y/N)? (D467MSG 504)
N

TEST SECTION 5 (DATA TESTS) COMPLETE   (D467MSG 65)
END OF PASS 1   (D467MSG 508)

TERMINAL BREAK ENABLED, CONTROL Y DISABLED BY THE VERIFIER
(D467MSG 538)
END OF PROGRAM

8. If a test section is to be repeated the following data
is displayed:

DO YOU WANT TO LOOP ON SECTIONS(S) (Y/N)? (D467MSG 504)
Y

ENTER LOOP COUNT (MAX=20) (D467MSG 506)
10

The test will loop for the select loop count and then issue
the following message:

END OF PASS x (D467MSG 508) X = PASS COUNT

DO YOU WANT TO RUN INDIVIDUAL SECTIONS OF THE VERIFIER (Y/N)

     User can now reselect sections or EXIT.

9.   If all sections tests are to be run respond with N to the
     following message and verify the following system response.

DO YOU WANT TO RUN INDIVIDUAL SECTIONS OF THE VERIFIER(Y/N)
N
TEST SECTION 2 (LOOPBACK COMPLETE)   (D467MSG 65)
TEST SECTION 3 (SELFTEST)COMPLETE)   (D467MSG 65)
TEST SECTION 4 (STATUS TESTS)COMPLETE (D467MSG 65)
TEST SECTION 5 (DATA TESTS)COMPLETE   (D467MDH 65)
END OF PASS 1

TERMINAL BREAK ENABLED, CONTROL Y DISABLED BY THE VERIFIER
END OF PROGRAM

2.2 SECTION 6 INTERACTIVE TESTS

When section 6 is selected from the INDIVIDUAL SECTION display,
the verifier enters the interactive mode.   The user responds to
prompts issued by the verifier. The procedure involves operating
the page printer keyboard as directed by the verifier.   The
verifier then processes the manual data and checks the printer
response to these commands.   The following functions are tested
in the interactive mode:

     Halt/Run
     Power Fail
     Page Length of 4.0 inches
     Page Length of 8.5 inches
     Status

2.3 DIAGNOSTIC PROCEDURE

When the 2680A verifier encounters an equipment fault when
running a test section, a message is displayed to the user
similar to the following:

ERROR IN STEP 42 - OPERATION = TEST STATUS BITS (467445 51)

ATTEMPT TO SATURATE THE CHARACTER BLOCK PROCESSOR FAILED
(I/O STATUS WORD 4 BIT 8).   (D467ERR 528)

DO YOU WANT TO DISPLAY PRESENT PRINTER STATUS (V=VERBAL,O=OCTAL,
N=NO)

Step 42 refers to section 4 test of the printers ability to
detect character block saturation.

The user can display the printer status in two formats. If
verbal is selected a message similiar to the following will be
displayed.

PAGE PRINTER I/O STATUS

ATTEMPT TO SELECT AN ILLEGAL LOGICAL PAGE TABLE ENTRY (D467EER47)
ERROR DETECTED IN RECORD 1 ON SHEET NUMBER 0.
IN RECORD 1 ON SHEET NUMBER 0.

                    PAGE PRINTER ENVIRONMENTAL STATUS
2680A MEMORY SIZE = 256 WORDS   DCS FIRMWARE DATECODE=2114
NUMBER OF VFC'S LOADED =1   NUMBER OF FORMS LOADED=0
PAGE LENGTH=8.5 INCHES   PAGE WIDTH =11.0 INCHES
(21.59 CM)   (27.94 CM)
HPIB DEVICE ADDRESS = 7
SIZE OF THE INCOMING DATA BUFFER = 16 (512 WORD BLOCKS).
NUMBER OF AVAILABLE 20 WORD BUCKETS = 12105.
NUMBER OF BUCKETS USED SINCE LAST JOB OPEN =47
NUMBER OF DATA BLOCKS IN THE INCOMING DATA BUFFER=0
NUMBER OF ACTIVE LOGICAL PAGES=1
NUMBER OF CHARACTER SETS LOADED=1
NUMBER OF CHARACTER SET WORDS=1712
NUMBER OF FORM WORDS=0
NUMBER OF VFC WORDS LOADED=66

If octal is selected, a complete dump of both the I/O status and environment status is displayed. Note, the % sign indicates value is presented in octal notation.

SCREEN MODE TRAILER LINE

WORD	I/O STATUS	ENV STATUS
0	%004004	%000020
1	%000000	%027511
2	%000000	%000057
3	%000000	%010100
4	%001000	%070101
5	%000000	%000654
6	%000000	%000000
7	%000000	%000102
8	%000000	%021156
9	%000000	%000000
10	%000000	%000675
11	%000000	%004102
12	%000000	%000000
13	%000001	%000000
14	%000000	%000000
15	%000000	%000000

Notice that bit 6 in word 4 is on. Refer to the STATUS and CONTROL WORDS section (appendix A) for a description of word 4 bit 6. Word 13 bit 1 indicates error was detected in record one (1) of the transmitted data block.

The octal representation of the environmental octal dump is a duplicate of environmental display and is displayed for information only.

2.4 DATA TESTS

The data test run during test section 5 causes the printer to print specific data and forms. This data is used to verify proper printer operation. Appendix C contains a copy of the data generated during test section 5.

```
+---------------------------------------------------+ +---------+
|                                                   | | SECTION |
|                                 EXECUTION TIMES   | |---------|
|                                                   | |   III   |
+---------------------------------------------------+ +---------|
```

3.0 INTRODUCTION

The following information supplies the time required to run each
section of the page printer verifier diagnostic.

NOTE

The execution time for section 1 tests is not listed as
time is dependent on spool time, configuration, load,
and other factors.

The execution times listed are based on the slowest time
(measured on an HP3000 series 30).

SECTION	NAME	RUN TIME
1	Configuration and Basic Communication Test	NA
2	Loopback Test	4 seconds
3	Selftest	3 seconds 13 if looping
4	Status Test	1 min, 15 sec
5	Data Test	1 min, 33 sec
6	Interactive Test	manual

```
+----------------------------------------------------+ +---------+
|                                                    | | SECTION |
|                              TEST DESCRIPTION | |---------|
|                                                    | |   IV    |
+----------------------------------------------------+ +---------+
```

4.0 INTRODUCTION

The 2680 Page Printer Verifier is a diagnostic program to verify
the configuration and operation of the 2680A Page Printer.   The
program is functionally divided into sections and step numbers to
allow the program to reference section step numbers when error
messages are issued to the user.   In this manner the user can
refer to this section of the diagnostic manual for detailed
information concerning the test performed.

Error messages generated by the verifier will indicate the step
where the error occurred.   The error message format is as
follows:

ERROR IN STEP x (where x equals the step number)

After the error message is generated, the program will issue a
problem description, and when possible, a list of probable
causes. Messages originating from the system message catalog are
identified as Page Printer messages (D467MSG) or Page Printer
errors (D467ERR).

System Message Catalog Examples:

SPOOLING TO LDEV 14 (2680A PRINTER) WAS STOPED BY THE VERIFIER.
(D467MSG 4)

2680A PAGE PRINTER IS NOT CONFIGURED INTO THIS SYSTEM.(D467ERR1)

The verifier uses message set 2 in the system message catalog to
further explain failures relating to intrinsic commands.   *

If an I/O error is detected while the verifier is using an
intrinsic or external procedure (read, print, GENMSG..etc), the
verifier will first attempt to restore th original spooled state,
reset the ALLOW mask for the user, and abort the program by
calling th QUIT intrinsic.

Input/output status errors ocurring in the 2680A, while the
verifier is in process, will be handled by the verifier.   I/O
errors are normally handled by the driver.   The driver is
notified on each request that an I/O request from the verifier is
being processed.

Irrecoverable errors, detected by the verifier, will cause the
program to report the error; restore the users original ALLOW
mask; return the spooler to its original state (providing the
spooler was not altered since the verifier began executing);
enable the console BREAK function, and disable the control Y
break.

All other errors will allow the user to obtain the printer status
(I/O and environmental) and display the status for analysis.
After each error, the user will have an option to continue or to
exit the program.

4.1 TEST SECTIONS

The Page Printer Verifier is divided as follows:

```
+--------+-------------+------------------------+
|SECTION | STEP RANGE  |         NAME           |
|--------|-------------|------------------------|
|        |             |                        |
|   1    |   10-14     | Unit configuration and |
|        |             | present condition      |
|        |             |                        |
|   2    |    20       | HP-IB Loopback         |
|        |             |                        |
|   3    |    30       | Selftest               |
|        |             |                        |
|   4    |   41-47     | Status Tests           |
|        |             |                        |
|   5    |   50-55     | Data Tests             |
|        |             |                        |
|   6    |   60-62     | Interactive User Test  |
|        |             |                        |
|   7    |    70       | Restore Users          |
|        |             | Environment            |
|        |             |                        |
+--------+-------------+------------------------+
```

4.2 TEST SECTION DESCRIPTIONS

The test section description is arranged in STEP NO. order to allow the user to quickly locate diagnostic data. The test description contains a functional description of each test step.

10   Verifies user has system manager or operator capability. Prints title message and checks system tables for a 2680A printer.

Verification is accomplished by checking the LDT for a type 32 (printer) and the LPDT for a subtype 8 (2680A). When types 32 and 8 are detected, the LPDT is checked to determine if the printer is a virtual device.

If the printer is a virtual device, as setup by another process, the LDEV and the spool status are not saved. If the printer is not a virtual device, the logical device number (LDEV) and the spooled state information is store until the entire LDT has been tested. If more than one 2680A is detected, the verifier will identify the LDEV and request the user to enter the LDEV to be tested.

The step is completed by disabling the user terminal BREAK key and enabling the control Y function.

11   Obtains CPVA address for channel program failures. Verifies 2680A is properly connected by checking printer identification.

Failures encountered at this point can be attributed to one of the following 2680A conditions.

PROBABLE CAUSE

Unit not connected or powered-on.

Unit not configured properly

Unable to identify. Defective General Interface Channel (GIC) chip or DCS PHI chip.

Two devices with the same device number sharing the same GIC.

12   Assumes 2680A control by issuing STOPSPOOL, STARTSPOOL, SUSPENDSPOOL, DOWN, UP, GIVE or TAKE console commands.

If the printer is not active, the verifier will place the device under test into the diagnostic mode.

If an active job is printing or waiting to be printed, the verifier will suspend spooling to the LDEV under test.

The user is notified that the verifier will wait for the active job to complete.

The operator will be notified anytime the spooled state of the LDEV is modified by the verifier. The user will also be notified that the verifier will change the spooled state of the device under test.

13   Obtains printer status. Displays the following information on the console:

                      PRESENT 2680A ENVIRONMENT

2680 MEMORY SIZE=   K WORDS   DCS FIRMWARE DATECODE=
NUMBER OF VFC'S LOADED=   NUMBER OF FORMS LOADED=
PAGE LENGTH=   INCHES (cm)   PAGE WIDTH=
INCHES(cm)

14   Issues job abort and job open commands to the printer. Forces printer to clear previous environment and load default environment.

        Reads environmental status and verifies default environment is loaded (1 character set, 1 VFC, 1 active LPT, no forms etc..)

        Issues job close command after status is checked to complete job open request.

        Section 2 HPIB LoopBack

                              NOTE

        User can select and loop on this and any other test section for troubleshooting purposes. After a section is performed, the program returns to this point for test selection or program termination.

20   Verifies integrity of data link between host system and printer. The host system is instructed to transmit 256 data bytes (0-255) to the printer, with a write loopback request. Data is stored in the printer buffer, read back (read loopback request), and compared with the original transmitted data.

        Test will loop 100 times to detect any possible intermittent problems.

Section 3 - Selftest

30   Issues printer selftest command. Command activates a
portion of the printer's internal micro-diagnostics
affecting the printer DCS and memory. Checks selftest
results.

The following selftests are performed:

> Character block processor test
> Main memory test
> External register test
> Data processor test

NOTE

A ten second delay will be added to the selftest
function if section 4 (status) or section 5 (data)
are selected with the looping option. Delay is
required to insure the laser beam has been turned
off before the printer's internal diagnostics are
executed.

Section 4 - Status Test

40   Issues a job abort clear to place printer into the default
state and issues a job open command. Tests printer
ability to:

detect an attempt to print with a deleted character set
(status word 5 bit 2)

detect an attempt to select an undefined form (status
word 4 bit 6)

select an undefined VFC (status word 4 bit 5)

The character set test is performed first by downloading a
character set containing one character and then selecting
and deleting the character set (printer does not know data
has been deleted until data access is attempted).

An attempt to print using this character set should
produce an error.

The form and VFC are tested by downloading a logical page
table with erroneous information (undefined VFC and
turning forms on without the form being loaded). The
printer receives a block of data describing the section
and step number. When the printer attempts to link this
data, the printer will detect an error in each case (form
and VFC) and set the appropriate status bit. Once the
status bit is detected a job abort is issued to prevent
the data from being printed

41   Test the printers ability to detect that the pen has been
        moved off of the logical page (status word 4 bit 7) and
        selection of an illegal logical page table (LPT) (status
        word 4 bit 6) will cause an error.

        The move pen error is generated by issuing the move pen
        function to the printer with an "X" coordinate that is off
        the page.   A select logical page function to a non
        existent logical page table followed by a print request is
        made to generate a selection of an illegal LPT.   Printing
        will not occur.

42   Test the printers ability to detect saturation of the
        character block processor (status word 4 bit 8) and an
        attempt to exceed the maximum number of copies per physi-
        cal page (status word 4 bit 11).

        Saturation of the character block processor occurs by
        sending one line of data (130 A's) followed by a move pen
        command of minus ten dots in the Y direction.   Another
        line of data (130 B's) is then sent followed by a move pen
        of minus ten dots in the Y direction. A line of data (130
        C's) followed by another line of data (130 D's) is then
        sent to the printer to create the error.

        Exceeding the maximum copies per physical page error is
        created by setting the maximum number of copies to two
        (2), setting the repeat page limit to three (3) and
        transmitting a line of data to print.

43   Test the printers ability to detect an attempt to select
        an undefined character set (status word 4 bit 3) and out
        of memory for an attempted Character Set load (status word
        4 bit 0).

        test the printers ability to detect the following:

            An attemptan attempt to select an undefined character
            set (status word 4 bit 3)

            out of memory condition for an attempted character set
            load (status word 4 bit 0)

        The undefined character set selection error is created by
        downloading a character set and selecting it, followed by
        selection of a non-existent character set with an attempt
        to print data using this non existent character set.

        The out of memory error is created by sending the printer
        a character set that has the number of words per character
        equal to 8190 and the number characters in the set equal
        to 128.   The printer will compute that the space required
        to load this set exceeds the printers capacity (256K
        words).

44   Test printers ability to detect no memory available for an
      attempted VFC and Form load.

      The printers memory is filled up with large VFC's (64K
      words each). When the last VFC is loaded the printer will
      obtain the number of words in the VFC table that is about
      to be loaded (64K), calculate that it cannot fit into the
      available memory space and generate an error.

      The form load error is generated by sending the printer a
      form load that has 128 triplet words and 8190 dot per bit
      words in the form descriptor block. The printer will
      calculate that this amount of data will not fit into the
      users memory area and generate this error.

45   Test the printers ability to detect an invalid spoolfile
      block error (status word 1 bit 12).

      A write with an invalid printer function code is
      transmitted to generate this error.

46   Test printers ability to detect a "no memory available for
      data" error and a "select VFC with LPT word 10 of -1"
      error.

      The first error, no memory available for data, is
      generated by filling the available users memory area in
      the printer with VFC's and a form so that there are only
      approximately 500 words of memory left. A block of data,
      1024 words, is then sent to create the error (status word
      4 bit 13).

      The second error is generated by loading an LPT with word
      10 (height of base character set) equal to -1. This LPT
      is selected and then data describing the test being
      performed is transmitted to the printer. The attempt to
      print this data, using the printers default character set,
      with word 10 being -1 will cause the error (status word 4
      bit 14).

47   Test printers ability to create a skip to a non-existing
      VFC error and its ability to clear forms, VFC's and
      character sets.
      A non-existing VFC error is created by sending a VFC with
      all zeros to the printer, selecting that VFC and sending
      data that describes the test being performed. When the
      printer attempts to use this VFC it will detect the error
      (status word 4 bit 15).

      Test printers ability to clear forms, VFC's and character
      sets.

      The clearing of forms, VFC's, and character sets is teste
      by loading the environment file, "D467ENV" and checking

the correct number of forms, VFC'S, and characters were loaded. The printer is then commanded to clear all forms, VFC's, and character sets. The printer's environmental status is then checked to confirm that they have been cleared.

Section 5 - Data tests

This section will exercise the remaining portions of the printer not previously tested by the selftest command. This includes the Machine Control System (MCS) which controls and monitors paper movement and all mechanical functions associated with paper movement.

When the Verifiers environment file (Appendix B) is loaded, all logical pages (0 through 6) sent to the printer will be initially active. The Verifier will deselect and select logical page tables as required. It will also select the appropriate primary and secondary character sets, turn off the multi copy forms overlay feature and back on again when required, turn on and off the auto page eject function and use all three modes of carriage control (<%200, %200-%277 and %300-%317).

This section will generate 17 pages of data. The format of this data is contained in Appendix C. The first page of this data will be reproduced three times on the printer but will be shown once in the Appendix.

50   Refer to figure 838.2.   .

Issue a Job Open and verify proper operation of the previously loaded default environment by printing three pages of data using the entire ASCII Character Set. One page of data will be transmitted and the Printer will be instructed to repeat this page 3 times. This will test the Printers repeat page feature. A Job Close will be issued to force the last page of data to be printed. The environmental status will be read to determine if the three pages were printed. The Job Close will clear all variables associated with the previous job and toggle the job separation marks.

51   Refer to figure 838.3

Issue a Job Open and load all Logical Page Tables, Character Sets, VFC's and Forms that will be used for the remaining tests. Test Printers ability to print 20.1K characters on one page using a 7 X 11 (dot) character set The data will be printed in the Portrait Mode (90 degree shift with respect to the direction of paper motion). There will be two character sets selected at this point. The primary character set is 14 X 22 dots in the landscape

direction (0 degree shift) and the secondary is 7 X 11 dots in the portrait mode. The character set selection will be done using the eighth bit mode (most significant bit on selects the secondary character set). If the printer fails to select the secondary character set using

the eighth bit mode, the data will be printed in the landscape mode.

A null clear (0) will be issued to force this data to be printed.

52   Refer to Figure 838.4.

Test the Printers ability to print 255 characters/line using a 7 X 11 (dot) secondary character set. Printing will be done in the Landscape Mode (printing is not shifted,same as default mode) using the eighth bit mode t to select the secondary character set. The primary characte set is 14 X 22 dots in the landscape mode.

This step will print the alphabet beginning with ten "A"'s per line and incrementing each line length by ten until the letter "Z" is printed which will only print 255 characters per line. The last line will be repeated ten more times and the alphabet will be decremented by ten characters per line and it will terminate with ten "A" characters on the last line. A null clear will be performed to force the page to be printed.

53   Refer to figures 838.5 through 838.8

Using the default Vertical Forms Control (VFC) test all channels. User will determine if this feature worked properly by checking the printed results. Two forms will be used in this step. The first one provides a border that will indentify the boundaries of each logical page. The second form will provide line numbers (1 - 60) that will be used to verify proper operation of the VFC. Data will be printed in the Landscape Mode (default) using a 7 X 11 (dot) character set. Three logical pages per physical page will be utilized to place more actual data on each physical page. A null clear will be performed to print the last page of data.

The first and third physical page of data will be printed with the printers auto page eject feature on and the second physical page will be printed with this feature off. With VFC's enabled and being utilized, the auto eject feature should have no effect. This step also uses the prespace and postspace mode of printing.

54   Refer to figure 838.9 through 838.11

Print a page of data with multiple character sets and
forms using the multicopy forms overlay feature in the
printer. Data will be printed in each of the various
print directions except the reverse landscape direction
(180 degree shift). A null clear will be performed to
force the last page of data to be printed.

This step will print a three part form consisting of five
individual forms. They are the form itself, a border with
the form number and three forms each describing the title
of the copy being printed. The only data that is
transmitted to the form for printing is the statement
"SAMPLE FORM". This statement is positioned on the form
using the move pen absolute feature in the printer.

55   Refer to figures 838.12 through 838.14

This step tests the shift in and shift out feature of se-
lecting primary and secondary character sets. The header
on the printed page, describing the test being printed,
will only appear on the first page. After the header is
printed the logical page for the header will be turned off
and the only active logical page will be LPT 2 (10.9" X
7.0"). Page switching will be done with the VFC.

The entire ASCII character set, beginning with apostrophe
(%41) and ending with tilde (~ %176), will be printed.
The two character sets used for this test are:

    a. Primary set - (14 x 22 dots) landscape direction
    b. Secondary set - 7 x 11 dots) landscape direction

Each line will have 25 ASCII characters printed in the
primary character set, shift in mode, followed by a
descriptive block of data in the secondary character set,
shift out mode. The same data described in the previous
line will be repeated once more on each line. The test
will continue with the next ASCII character until the last
one has been printed.

56   Refer to Figures 838.15 and 838.16

This step will test the page skipping features of the
write function (parameter P1) and the page control
function (function 140).

This step will begin by activating logical pages 3, 4, and
5, selecting character set zero (0) and performing a
physical page eject (function 140 parameter P1=3). The
step number information will be printed and a skip to the
next logical page (LPT 3) will be issued with a write
function P1 parameter of %61. From LPT 3, character set 2
will be selected, two lines of data will be printed, and

an unconditional skip (write function P1=%63) to LPT4 will
be issued.   Two lines of data will be printed from LPT 4
and a conditional skip (write function P1=%61) will be
issued. On LPT 5, print one line of data identifying the
logical page. A function 3 (file open) is issued to close
the physical page of data and move the pen to the top of
the next physical page.

Once a new page has been opened, a function 4 (file close)
and multiple page ejects using write functions with P1
parameter =%61 will be issued, to verify these functions
do not cause multiple page ejects. At this point the step
number information and a description of the preceeding
lines in this paragraph will appear on the top of this
page.   The page skipping feature using fuction code 140
will be tested as follows:

a.   Select LPT 3 (function 140 P2=3) and print using VFC.
b.   Select LPT 5 (function 140 P2=5) and print using VFC.
c.   Select LPT 4 (function 140 P2=4) and print using VFC.
d.   Select LPT 3 and print without VFC (function P1=4).
e.   Select LPT 4 and print without VFC (function P1=4).
f.   Select LPT 5 and print without VFC (function P1=4).

Selecting an LPT will cause all forms associated with it
to be printed.   If the VFC is used, printing will also
begin on line number 1.   If the VFC is not selected,. the
printing should begin on the top of the logical page.

At the completion of this step, a job close will be
issued.   The environmental status will be checked to
confirm the proper number of pages were printed.

Section 6 - Interactive user tests

This section will allow the user to test the remaining
testable status bits by placing the printer into known
states and allowing the Verifier to confirm   proper
detection of these conditions.

60   Ask user to press the halt button on the printer and type
        "GO" when ready.   I/O status word 1 bit 0 is checked for
        proper operation.

61   Ask user to power fail the printer, press run key, and
        type "GO" when ready.   I/O status word 1 bit 2 is checked
        for proper operation

        When the printer starts the power on sequence, the "HP2680
        PAGE PRINTER" message will be displayed, and approximately
        2 seconds later, the "JOB ACTIVE PWR FAIL" message will be
        displayed.

62   This step tests the ability to change page length from the
        printers keyboard, test for error when page length is made
        programmatically that is greater than that just set and
        verify that an attempt to print a larger logical page onto
        the smaller one set by the operator causes truncation of
        the larger logical page.

        This step will cause the printer to start a print
        operation (turn motors on).   Printing will not occur if
        normal operation is achieved.

        Section 7 - Restore users environment
70   Restore original spooled state of the device under test.
        Before this is done the Verifier will check the present
        spooled state to see if it has been altered since the
        diagnostic began. If it has, the user will be notified of
        this occurence and the Verifier will terminate.   If not,
        the original condition will be restored.

        Restore users original environment (allow mask), enable
        terminal break key and disable control Y.

```
+--------------------------------------------------+ +----------+
|                                                  | | APPENDIX |
|                        STATUS AND CONTROL WORDS  | |----------|
|                                                  | |    A     |
+--------------------------------------------------+ +----------+
```

A-1 INTRODUCTION

I/O Status

The 2680A status reports contains 16 data words to indicate the
the condition of the 2680A system. The status report is used to
to diagnose 2680A system faults. The following is an example of
example of an I/O display in response to the OCTAL command.

NOTE

Words 2 through 15 and bits 1,2,3 and 4 of word 1 are
cleared whenever the I/O status block is returned to the
host system.

WORD	I/O STATUS	ENV STATUS
0	%004004	%000020
1	%000000	%027511
2	%000000	%000057
3	%000000	%010100
4	%001000	%070101
5	%000000	%000654
6	%000000	%000000
7	%000000	%000102
8	%000000	%021156
9	%000000	%000000
10	%000000	%000675
11	%000000	%004102
12	%000000	%000000
13	%000001	%000000
14	%000000	%000000
15	%000000	%000000

I/O Status Word 0

Word 0 identifies status words containing valid information.
Each bit, starting with bit one, indicates the status word (1-15)
containing valid information. For example, if bit 4 is set (1),
then word four contains valid status data.

```
-----------------------------------------------------------
| 0   1   2   3   4   5   6   7   8   9   10  11  12  13  14  15 |
-----------------------------------------------------------
     |   |  |   |                              |   |   |   |
     |   |  |   |                              |   |   |   |
     ---------------------------------------------------------
                          |
                          |STATUS WORDS CONTAINING
                           VALID DATA
```

I/O Status Word 1

```
-----------------------------------------------------------
| 0   1   2   3   4   5   6   7   8   9   10  11  12  13  14  15 |
-----------------------------------------------------------
  |   |  |   |   |  |   |   |   |   |   |   |   |   |   |
  |   |  |   |   |  +------------------------------------+
  |   |  |   |   |                  |
  |   |  |   |   |                  |
  |   |  |   |   |               RESERVED
  |   |  |   |   |
  |   |  |   |   |---> TRANSMISSION ERROR DETECTED IN PRINTER
  |   |  |   |        (CHANNEL COMMAND ERRONEOUS OR
  |   |  |   |        DATA LENGTH ERROR)
  |   |  |   |
  |   |  |   |------> PARITY ERROR DETECTED IN HPIB
  |   |  |
  |   |  |
  |   |  |---------> POWER UP COMPLETED SINCE LAST I/O
  |   |             STATUS READ
  |   |
  |   |------------> MESSAGE BEING DISPLAYED ON 2680A PRINTER
  |
  |---------------> 0=ONLINE, 1=OFFLINE
```
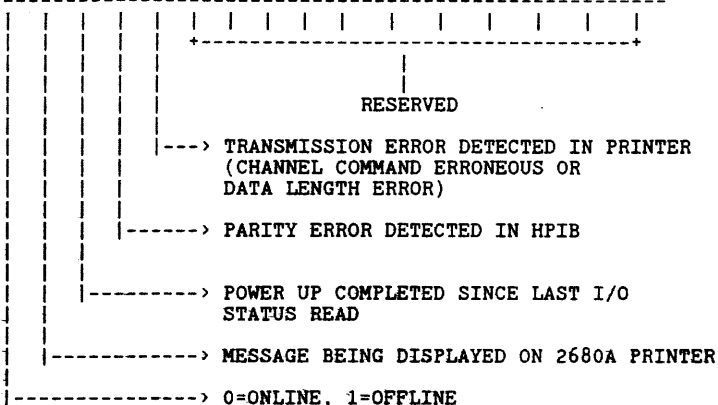
I/O Status Word 3 - Machine Control System (MCS) Fault Member

```
-----------------------------------------------------------
| 0   1   2   3   4   5   6   7   8   9   10  11  12  13  14  15 |
-----------------------------------------------------------
```

Contains octal word indicating a given machine fault (i.e. paper
jam, out-of-paper). The status word is translated to a message
and displayed on the printer readoutLED display.

I/O Status Word 4

| 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 |

BIT	DESCRIPTION
0	No memory available for attempted character set load
1	No memory available for attempted form load
2	No memory available for attempted VFC load
3	An attempt was made to print data without a selected character set
4	An attempt was made to select an undefined form
5	An attempt was made to print data without a selected VFC
6	An attempt was made to print data without a selected logical page table (LPT)
7	An attempt was made to move pen off the logical page.
8	The printer could not process all data before transfer was made to the drum/paper. Data will be lost.
9	Data block contains format error. Invalid function code or record/block size error
10	Missing multi-copy forms table. An attempt was made to use a multicopy forms table that was not loaded for this job.

BIT	DESCRIPTION
11	Maximum number of copies per physical page has been exceeded
12	A command or function code was received without a job in process
13	No user memory available. User memory is loaded with character sets, VFC's, forms and data. The current data transmitted cannot be processed and will be lost.
14	A VFC is selected by a logical page table entry which has word ten (line spacing on page) less than or equal to zero.
15	A skip was made to a non-existent VFC

I/O Status Word 5

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15															

BIT	DESCRIPTION
0	Logical page was truncated to fit on the physical page.
1	Page size requested by programmer does not match page length set by operator. The operator-set page length will be used.
2	No character set selected when print record was processed. Record was skipped.

NOTE

I/O status words 12,13,14, and 15 are double
word integers.

I/O Status Word 12

```
--------------------------------------------------------
| 0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 |
--------------------------------------------------------
```

Contains error record number defined by word 4.
Information is reported during a JOB function.

I/O Status Word 13

```
--------------------------------------------------------
| 0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 |
--------------------------------------------------------
```

Contains error record number defined by word 4.
Information is reported during a JOB function.

I/O Status Word 14

```
--------------------------------------------------------
| 0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 |
--------------------------------------------------------
```

Contains sheet number where error occured as defined
by word 4. Information is reported during a job
function.

I/O Status Word 15

```
--------------------------------------------------------
| 0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 |
--------------------------------------------------------
```

Contains sheet number where error occured as defined by word 4.
Information is reported during a job function.

Environmental Status

The environmental status report contains 16 data words indicating current configuration, print job, and printer mode of the 2680A page printer. Data is supplied to assist in the interpretation of diagnostic data.

Environmental Status Word 0

```
------------------------------------------------------
| 0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 |
------------------------------------------------------
   |  |  |  |  |  |  |  |  |  |   |   |   |   |   |   |
   ---------------------         ---------------------------
            |                                |
            |                                |
 Number of data blocks          Size of incoming data buffer
 in the incoming data           in 512 word blocks
 buffer
```

Environmental Status Word 1

```
------------------------------------------------------
| 0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 |
------------------------------------------------------
```

Number of twenty word buckets available.

Environmental Status Word 2

```
------------------------------------------------------
| 0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 |
------------------------------------------------------
```

Maximum number of buckets used since last job open.

Environmental Status Word 3

```
-----------------------------------------------------
| 0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 |
-----------------------------------------------------
   |  |  |  |  |  |  |  |  |  |   |   |   |   |   |   |
   ----------  ----------------  --------------------
      |              |                     |
      |              |           Number of forms loaded in
      |              |           printer
      |              |
      |              |---------->Number of VFC's loaded in
      |                          printer
      |
      |------------------------->Amount of Printer Memory
                                 00=128K words
                                 01=256K words
```

Environmental Status Word 4

```
-----------------------------------------------------
| 0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 |
-----------------------------------------------------
   |  |  |  |  |  |  |  |  |  |   |   |   |   |   |   |
   ----------  ----------------  --------------------
      |              |                     |
      |              |           Character sets loaded in
      |              |           printer
      |              |
      |              |---------->Number of active logical
      |                          pages
      |
      |------------------------->HPIB address (dev 0-7)
```
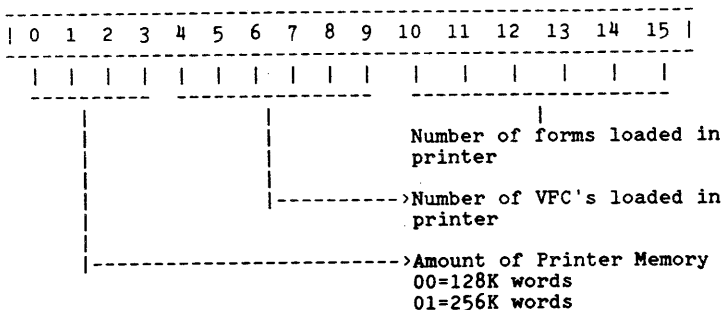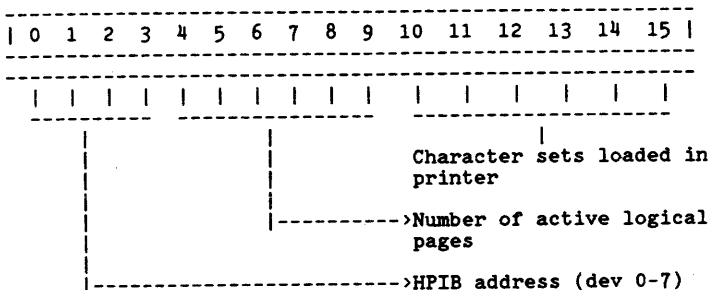
Environmental Status Word 5
```
-----------------------------------------------------
| 0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 |
-----------------------------------------------------
```

Number of character set dot/bit image (words+3)/4 plus the number
of proportional spacing (words used plus 3)/4.

Environmental Status Word 6

```
-----------------------------------------------------
| 0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 |
-----------------------------------------------------
```

Number of form dots per bit (words + 3)/4 plus the number of form triplet (words plus 3)/4.

Environmental Status Word 7
```
-----------------------------------------------------------
| 0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 |
-----------------------------------------------------------
```

Number of VFC words loaded.

Environmental Status Word 8
```
-----------------------------------------------------------
| 0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 |
-----------------------------------------------------------
   |  |  |  |  |  |  |  |  |  |   |   |   |   |   |
   ----------------------        ---------------------------
            |                                  |
            |                                  |
   Page length in the          Page width in direction of laser
   direction of of paper       scan, in 0.1 inch increments
   motion, in 0.25"
   increments
```

Environmental Status Word 9

```
-----------------------------------------------------------
| 0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 |
-----------------------------------------------------------
   |  |  |  |  |  |  |  |  |  |  |   |   |    |   |   |
   -------------------------------------------   |   |   |
                                                 |   |   |
                          |                      |   |   |
                          |                      |   |   |
                          ~                      |   |   |
                       NOT USED                  |   |   |
                                                 |   |   |
                                                 |   |   |
   End of job encountered while printer          |   |   |
   was in the silent running mode      <------   |   |
                                                     |   |
                                                     |   |
   Form not printed on page as form started         |   |
   before the quarter inch margin on the top        |   |
   or bottom of page.  Error was caused either <--|  |
   by programmer error, or operator used the        |
   registration switches to locate the form off     |
   the page.                                         |
                                                     |
                                                     |
                                                     |
   Data truncated from top or bottom of page.        |
   Programming error occurred or operator moved      |
   the print of the page with the registration  <----|
   switches.
```

Environmental Status Word 10

```
-----------------------------------------------------------
| 0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 |
-----------------------------------------------------------
```

Number of USER AREA words actually loaded, plus 3
divided by 4.

Environmental Status Word 11

```
-----------------------------------------------------------
| 0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 |
-----------------------------------------------------------
```
Date code of DCS firmware currently installed

Environmental Status Word 12

```
------------------------------------------------------------
| 0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 |
------------------------------------------------------------
```

Number of non blank characters clipped (not printed) on this job.

Environmental Status Word 13

```
------------------------------------------------------------
| 0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 |
------------------------------------------------------------
```


Reserved

Environmental Status Word 14 and 15

```
------------------------------------------------------------
| 0  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 |
------------------------------------------------------------
```

Number of physical pages printed since last job open (signed double integer). Indicates total number of physical pages printed for this job since the environmental status block read function.

```
+----------------------------------------------+ +----------+
|                                              | | APPENDIX |
|                           ENVIRONMENT FILE|  |----------|
|                                              | |    B     |
+----------------------------------------------+ +----------+
```

B-1 INTRODUCTION

The environment file (D467ENV) used by verifier is described
below.

CHARACTER SETS

CHAR SET NO.	CELL HEIGHT	WIDTH	ORIENTATION	CHARACTER CODE BASE- LOWEST	HIGHEST	LINE
0	22"	14"	0 DEGREES	32	126	6
1	11"	7"	90 DEGRESS	32	126	3
2	11"	7"	0 DEGRESS	32	126	3

LOGICAL PAGE TABLE

LOGICAL PAGE TABLE NUMBER	PAGE WIDTH	HEIGHT	ORIENTATION	DISTANCE FROM LEFT	TOP	MARGIN	BASE CHAR SET
0	10"	1"	0 DEGREES	0.5"	0.5"	0	0
1	6.5"	10"	90 DEGRESS	1.5"	1.0"	0	1
2	10.9"	6.5"	0 DEGREES	0.1"	1.5"	0	2
3	3.25"	4.06"	0 DEGREES	.25"	2.5"	0.2	2
4	3.25"	4.06"	0 DEGREES	3.75"	2.5"	0.2	2
5	3.25"	4.06"	0 DEGREES	7.25"	2.5"	0.2	2
6	11"	7"	0 DEGREES	0"	1.0"	0	0

FORMS TABLE

FILE NAME	FORM NAME	NUMBER OF SUBFORMS	ASSOCATED WITH LOGICAL TABLE NO
NEWFORMS	COPY1	1	6
NEWFORMS	COPY2	1	6
NEWFORMS	COPY3	1	6
NEWFORMS	HEADBORDER	1	0
NEWFORMS	LOGPAGENUM1	2	3
NEWFORMS	LOGPAGENUM2	2	4
NEWFORMS	LOGPAGENUM3	3	5
NEWFORMS	NEWLINENUMS	1	3,4,5
NEWFORMS	NEWREQFORM	13	6
NEWFORMS	REQBORDER	1	6

```
+----------------------------------------------------+ +---------+
|                                                    | |APPENDIX |
|                                              DATA  | |---------|
|                                                    | |    C    |
+----------------------------------------------------+ +---------+
```

C-1 INTRODUCTION

Appendix C contains copies of the test printout generated during
the performance of test section 5.   Data is supplied to show the
normal response to the section 5 diagnostic test.

SECTION 5 STEP 50 (Print ASCII character set using printer default set)
           (Test REPEAT PAGE FEATURE by duplicating this page 3 times)

Figure C-1.   Repeat Page Feature Test
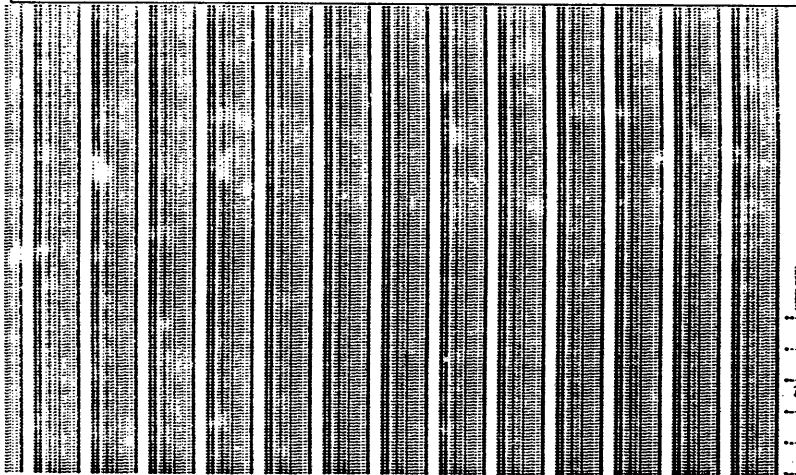
Figure C-2. Maximum Data Per Page Test (20K Chars+5%)

SECTION 5 STEP 52   (Print letters A-Z Start with 10/line and complete with 255)
(Print 10 more lines of (Z) and continue with letters Y-A at 250/line decrementing by 10/line

Figure C-3, Maximum Number of Characters (255) Per Line Test

SECTION 5 STEP S3 (Print using downloaded character set - no shift)
    (Test all VFC's - Use 3 logical pages/physical page)

LOGICAL PAGE 1

LOGICAL PAGE 2

LOGICAL PAGE 3

Figure C-4.   VFC CHannel (1,2,3,4 &5) Test

SECTION 5 STEP S3 (Print using downloaded character set - no shift)
(Test all VFC's - Use 3 logical pages/physical page)

LOGICAL PAGE 1

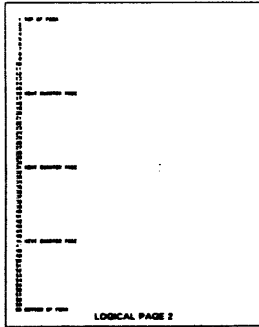LOGICAL PAGE 2

LOGICAL PAGE 3

Figure C-5. VFC Channel (1,2,9,11,12,13,14 & 15) Test

Figure C-6. VFC Channel (1,2,9,11,12 &16) Test

SECTION 5 STEP 53 (Print using downloaded character set - no shift)
(Test all VFC's - Use 3 logical pages/physical page)

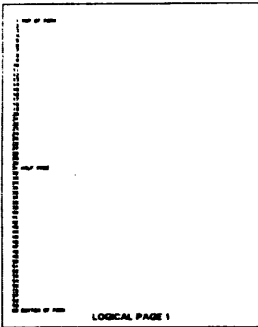LOGICAL PAGE 1   LOGICAL PAGE 2   LOGICAL PAGE 3
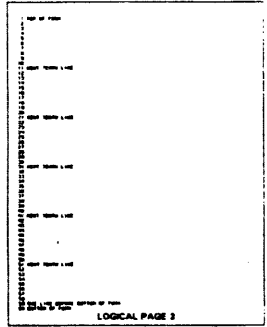
Figure C-7. VFC Channel (1,2,3,6,7,8,9,10, & 12) Test

SECTION 5   STEP 54   (Print form with no shift)
This step tests the multi copy overlay feature

Figure C-8. Multicopy Overlay Test - User's Copy

SECTION 5 STEP 54 (Print form with no shift)
This step tests the multi copy overlay feature

Figure C-9. Multicopy Overlay Test - Billing Copy

SECTION 5 STEP S4 (Print form with no shift)
This step tests the multi copy overlay feature

HEWLETT PACKARD OFFICE SUPPLY REQUISITION 97528

INITIATED BY: _____ DATE _____ EXT _____
DELIVER TO: _____ BLDG: _____
APPROVED BY: _____

ACCOUNT NO. ☐☐☐☐ LOCATION: ☐☐ ☐☐ ☐☐☐

PHONE ORDER
MAIL ORDER
COUNTER ORDER

HP PART NUMBER	QUANTITY REQUIRED	UNIT MEA	ITEM DESCRIPTION	QUANTITY ISSUED	QUANTITY B O	UNIT PRICE	TOTAL
			SAMPLE FORM				
						TOTAL	

SHIPPING COPY

Figure C-10. Multicopy Overlay Test - Shipping Copy

C-11

SECTION 5 STEP 55 (TEST SHIFT IN AND SHIFT OUT MODE OF CHARACTER SET SELECTION)

Each line will have 25 ASCII characters (Shift In) followed by data in the Shift Out mode. The same
data will be repeated once more on each line. This test will go through the entire ASCII character set (33-126)
This logical page will be turned off and the WFC will control printing on the next 3 pages

Figure C-11, Character Set Selection Using Shift In/Shift Out

Figure C-12. Character Set Selection Using Shift In/Shift Out

Figure C-13. Character Set Selection Using Shift In/Shift Out

SECTION 5 STEP 56 (Test page skipping features using the Write Function)

Figure C-14. Write Function Skip Test

SECTION 5 STEP S6 [Test page skipping features using the LPT Skip Function(140)]

A file open (function code 3) was used to close the previous page of data.
An FCLOSE and multiple page ejects (write using %61) were issued to confirm that
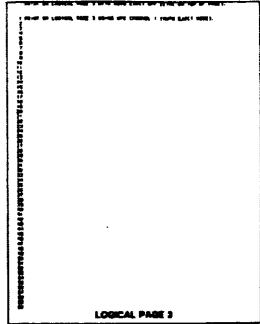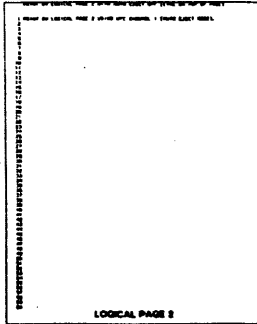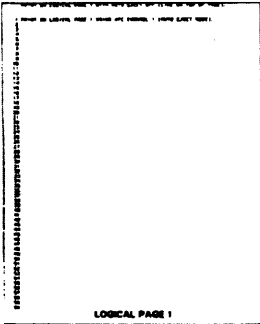multiple page ejects do not occur without any data being transmitted.

LOGICAL PAGE 1

LOGICAL PAGE 2

LOGICAL PAGE 3

Figure C-15. LPT Function Skip Test