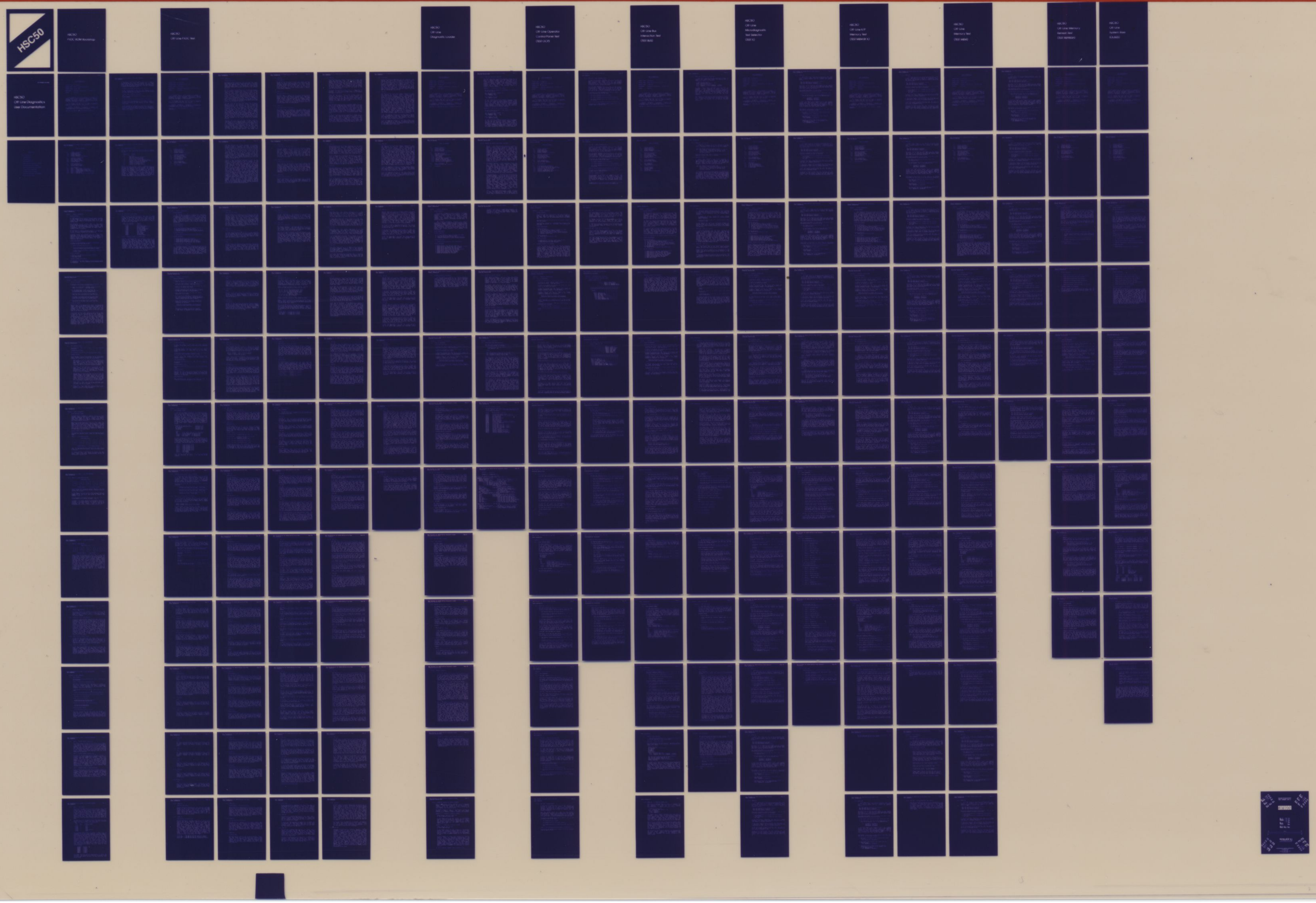
HSC50

OFF-LINE DIAGNOSTIC
USER DOCUMENTATION

EP-OHSC5-UG-002

FICHE 1 OF 1

MAR 1984
digital
MADE IN USA



HSC50

EK-OHSC5-UG-002

HSC50 Off-Line Diagnostics User Documentation

Table of Contents

HSC50

P.IOC ROM Bootstrap

USER DOCUMENTATION

PRODUCT CODE:	XX-XXXXX-XX
PRODUCT NAME:	HSC50 - P.IOC ROM BOOTSTRAP
PRODUCT DATE:	18-OCT-1982
MAINTAINER:	Diagnostic Engineering - Colorado (CX)
AUTHOR:	Mike Hare

TABLE OF CONTENTS

SECTION 1 - GENERAL INFORMATION

1.1 PROGRAM ABSTRACT

The HSC50 P.ioc ROM Bootstrap verifies the basic integrity of the HSC50's P.ioc module, Program memory, and TU58 interface. The goal of the bootstrap tests is to test enough of the HSC50 to allow further tests to be loaded from the TU58.

The bootstrap is the first step in the HSC50 Initialization process, and is run for every bootstrap or reload of the HSC50 operational software (CRONIC). The bootstrap is initiated automatically each time the HSC50 is powered-on, and is also initiated by CRONIC when a software re-boot is required.

The bootstrap is a PDP-11 program, written to execute in an F11 CPU in a stand-alone environment. I.E. No other software processes co-exist with the bootstrap.

Bootstrap failures are reported via the 'Fault Lamp' mechanism, which specifies the module (P.ioc, memory, or TU58) which is the most likely cause of the problem. In addition, certain information relating to TU58 failures is saved in a table in the Program memory, to assist in diagnosing tape errors.

1.2 SYSTEM REQUIREMENTS

In order for the bootstrap to complete successfully, the following hardware must be working:

- o Basic instruction set of the F-11
- o First 2,048 bytes of Program memory, plus 8 Kwords of contiguous Program memory below address 160000.
- o A TU58 drive and controller, containing a tape with a bootable image.

1.3 RELATED DOCUMENTS

1) HSC50 - P.ioc ROM Bootstrap program listing
2) F-11 Users Guide
3) TU58 Users Guide

1.4 HARDWARE ASSUMED TO BE WORKING

This program is the first software executed when the HSC50 is powered-on. No other diagnostics are executed before the bootstrap.

SECTION 2 - OPERATING INSTRUCTIONS

2.1 START-UP PROCEDURE

Follow the steps below to initiate the bootstrap. Refer to section 2.1.1 if this procedure fails.

1) Insert the HSC50 CRONIC SOFTWARE cassette into the TU58's unit 0 drive . (left-hand drive)

2) If the HSC50 power is OFF, turn power ON.
If the HSC50 power is ON, depress the INIT lamp.
The bootstrap will initiate automatically.

4) The INIT lamp on the HSC50's front panel will light when the bootstrap's F-11 tests are done.

5) The TU58's drive-motion LED should light within n seconds, indicating that the cassette's boot blocks are being loaded into Program memory.

6) The bootstrap transfers control to the first instruction of the image just loaded from tape.

2.1.1 IN CASE OF TROUBLE

Most failures in the bootstrap will result in lighting the 'FAULT' lamp on the HSC50's front panel. Depressing the 'FAULT' lamp momentarily will cause a failure code to be displayed in the front panel lamps. Section 3 of this document lists the meaning of each failure code, and specifies which of the HSC50 modules is the most likely cause of the bootstrap failure.

If a failure occurs in the tests of the F-11's basic instruction set, the FAULT lamp mechanism can not be used to report the failure. Instead the F-11 will execute a 'Branch dot' (E.G. "BR ."), and will not continue the bootstrap program. Failures of this type can be identifed by the failure of the INIT lamp to light. (The INIT lamp is lighted immediately after the basic F-11 tests complete.) If a local terminal is connected to the P.ioc, the exact instruction that failed can be determined by depressing the terminal's 'BREAK' key, and noting the address displayed on the terminal. Using a listing of the bootstrap, this address can be used to find the instruction that failed.

2.2 TEST PARAMETER ENTRY

The bootstrap does not accept any user-supplied parameters.

2.3 SETTING/CLEARING FLAGS

The bootstrap does not have any user-modifiable flags.

2.4 PROGRESS REPORTS

The bootstrap does not issue progress reports in the usual sense, however certain indications of the bootstrap's progress are visible to the user, as listed below:

1. LAMPS CLEAR - One of the first operations performed by the bootstrap is to clear all of the HSC50's front panel lamps. If the lamps fail to clear immediately after the bootstrap is initiated, a failure of the P.ioc is probable. (Circuitry on the P.ioc module is responsible for initiating the bootstrap program.)

2. INIT LAMP - The INIT lamp will be lit as soon as the basic tests of the F-11's instruction set are completed. These tests will normally complete within milli-seconds after the bootstrap is initiated. Failure of the INIT lamp to light indicates a failure in the P.ioc's F-11 processor.

3. TAPE MOTION - Following the tests of the F-11 and the Program memory, the bootstrap will try to load the INIT P.IOC TEST from the TU58. The LED on the tape drive containing the HSC50 CRONIC software should begin to blink as the load is proceeding. If the tape motion LED fails to blink within 15 seconds after the boot is initiated, see section 3.2.3 of this document for a discussion of possible causes of this failure.

4. STATE LAMP - When the bootstrap completes and initiates the INIT P.IOC test (or OFFLINE P.IOC TEST) the STATE lamp will be lighted.

5. FAULT LAMP - If the FAULT lamp lights during the boot process, the ROM tests have detected a fatal error. Refer to section 3 of this document.

SECTION 3 - ERROR INFORMATION

3.1 ERROR REPORTING SCHEME

The bootstrap is designed to operate without a local terminal, thus the bootstrap cannot use the terminal as a reporting mechanism for errors. Instead, the HSC50's front panel lamps are used to report errors, and to indicate the module (P.ioc, Memory, or TU58) which is the most likely cause of the error.

When the bootstrap detects an error, it will light the FAULT lamp on the front panel. Depressing the FAULT lamp momentarily will cause the bootstrap to display a failure code in the front panel lamps. The failure code will blink on and off at 1/2 second intervals. The meaning of each failure code is given in sections 3.2.x below. The bootstrap may be re-initiated by momentarily depressing the INIT lamp on the front panel.

3.2 SPECIFIC ERROR CODES

The following sections identify the HSC50 modules that are indicated by specific error codes in the front panel lamps.

3.2.1 CODE 21 - P.IOC MODULE FAILURE

FRONT PANEL LAMPS

```
+------+    +------+    +------+    +------+    +------+
|      |    |      |    |      |    |      |    |      |
|  ON  |    | OFF  |    | OFF  |    | OFF  |    |  ON  |
|      |    |      |    |      |    |      |    |      |
+------+    +------+    +------+    +------+    +------+
```

FAILURE CODE 21 - P.ioc Module Failure

This code specifies the P.ioc module as the most likely cause of the bootstrap failure. The P.ioc module should be replaced.

If a local terminal is connected to the P.ioc, the user can determine which test failed by examining the Test Number. Refer to the Manual Troubleshooting section of this document (section 3.3).

3.2.2 CODE 22 - M.STD MODULE FAILURE

FRONT PANEL LAMPS

```
+------+    +------+    +------+    +------+    +------+
|      |    |      |    |      |    |      |    |      |
|  ON  |    | OFF  |    | OFF  |    |  ON  |    | OFF  |
|      |    |      |    |      |    |      |    |      |
+------+    +------+    +------+    +------+    +------+
```

FAILURE CODE 22 - M.std Module Failure

This code specifies the M.std (memory) module as the most likely cause of a bootstrap failure. Possible causes of this failure include:

a) The memory test of the first 1 Kword (vector area) of Program Memory failed, and using the SWAP BANKS and SWAP BOARDS bits in the P.ioc CSR failed to correct the problem. (Test 2).

b) A contiguous 8 Kword partition could not be found in Program Memory below address 00160000. (Test 3).

In either of the above cases, the M.std module should be replaced. If a local terminal is connected to the P.ioc, the user can determine which of the errors occurred by examining the Test number. Refer to the Manual Troubleshooting Section of this document (section 3.3).

3.2.3 CODE 23 - TU58 FAILURE

FRONT PANEL LAMPS

```
+------+   +------+   +------+   +------+   +------+
|      |   |      |   |      |   |      |   |      |
|  ON  |   | OFF  |   | OFF  |   |  ON  |   |  ON  |
|      |   |      |   |      |   |      |   |      |
+------+   +------+   +------+   +------+   +------+
```

FAILURE CODE 23 - TU58 Failure

This failure code indicates that the TU58 is the most likely cause of the bootstrap failure. This failure can be caused by a hardware problem in the TU58 controller or drives, but can also be caused if none of the HSC50's TU58 drives contains a tape with a bootable image. Before replacing the TU58, try booting from a different drive (the bootstrap will attempt to boot from any drive connected to the P.ioc). If booting from a different drive also fails, try using another tape. If neither of the above two steps succeeds, and no local terminal is connected to the P.ioc, replace the TU58 controller. (If a local terminal is available, refer to section 3.3). If the bootstrap succeeds when a different drive is used, the original drive should be replaced. If the bootstrap succeeds when a different tape is used, replace the original tape.

3.3 MANUAL TROUBLESHOOTING

If a local terminal is connected to the P.ioc module, the ODT program (built into the F-11 microcode) can be used to obtain further information about a bootstrap failure. Refer to section 3.3.1, 3.3.2, or 3.3.3, depending upon the failure symptom.

3.3.1 'INIT' and 'FAULT' ARE BOTH OFF

If the INIT lamp and FAULT lamp are both OFF, either the bootstrap program was not automatically initiated, or the bootstrap's PDP-11 instruction test failed. To determine which failure occurred, depress the BREAK key on the local terminal. The ODT program built into the F-11 should respond by halting the F-11 and displaying the address of the current instruction in the bootstrap. To determine which instruction failed, use the address displayed to locate the corresponding instruction in the listing of the bootstrap program. Some F-11 failures will put the machine in a state such that depressing the BREAK key will have no effect, making it impossible to know which instruction is failing. If spares are available, try replacing the F-11 CPU chip and the F-11 MMU chip (one at a time, in that order). If replacing the chips does not fix the problem, or if spare chips are not available, replace the P.ioc module.

3.3.2 'INIT' IS OFF, 'FAULT' IS LIT

If the INIT lamp is OFF, and the FAULT lamp is on, a failure in the PDP-11 instruction test is indicated. If spares are available, try replacing the F-11 CPU chip, and the F-11 MMU chip (one at a time, in that order). If replacing the chips does not fix the problem, or if spare chips are not available, replace the P.ioc module.

3.3.3 'INIT' and 'FAULT' BOTH LIT

If the INIT lamp and FAULT lamp are both ON, depress the FAULT lamp momentarily and the fault code will be displayed. Then depress the BREAK key on the local terminal to halt the program. Now type: '172340/' (without the quotes). ODT will respond by displaying the contents of address 172340, which is the test number. Use the test number to refer to the proper test description in section 4 of this document.

SECTION 4 - TEST SUMMARIES

4.Ø TEST SUMMARIES

The following sections summarize the tests performed by the bootstrap.

4.1 TEST Ø - BASIC PDP-11 INSTRUCTION SET

This test verifies the correct operation of a subset of the F-11's instruction set. The subset of instructions chosen for testing includes only those instructions required for the completion of the bootstrap. The following instructions are tested:

Single Operand Instructions Tested (both word and byte mode):

```
  ADC,CLR,COM,INC,DEC,NEG,TST,ROR,ROL,ASR,ASL,SWAB,NOP
```

Double Operand Instructions (both word and byte modes):

```
  MOV,CMP,BIT,BIC,BIS,ADD,SUB
```

Branch Instructions Tested:

```
  BR,BNE,BEQ,BPL,BMI,BCC(BHIS),BCS(BLO)
  BGE,BLT,BGT,BLE,BHI,BLOS,BVC,BVS
```

Jump and Miscellaneous Instructions Tested:

```
  JMP,JSR,RTS,SOB,MTPS,MFPS
  CCC,CLN,CLV,CLZ,SEN,SEV,SEZ
```

Addressing Modes Tested:

  All 8 addressing modes are tested.

The F-11 instruction set test uses two methods of reporting errors. During the initial part of the test, errors result in an infinite program loop at the location where the error was detected. During the latter part of the test, (when enough instructions have been tested), the 'FAULT' lamp mechanism is used to report failures (section 2.1.1).

4.2 TEST 1 - PROGRAM MEMORY (SWAP BITS)

The memory modules designed for the HSC50 include special logic that allows the address range of the Program memory to be changed. The address range of the Program memory is controlled by two bits in the P.ioc's Control and Status Register (CSR): the SWAP BANKS bit, and the SWAP BOARDS bit. The purpose of this test is to verify that these two bits can be set and cleared. (The actual memory switching is not tested, just the ability of the bits to set and clear.) A failure in this test indicates that the P.ioc module must be replaced.

4.3 TEST 2 - PROGRAM MEMORY (VECTOR AREA)

In order for the HSC50 Operational Software (CRONIC) to function, the first 2,048 bytes of Program memory must be working (addresses 000000 thru 003777). This test verifies that the first part of Program memory works properly. If the test fails, the SWAP BANKS and SWAP BOARDS features are used in an attempt to swap a good portion of memory into the 000000 thru 0003777 address range. If the test still fails after all combinations of SWAP BANKS and SWAP BOARDS have been tried, a Program memory error is reported via the Fault lamp mechanism (section 2.1.1). A failure in this test indicates that the M.std module must be replaced.

4.4 TEST 3 - PROGRAM MEMORY (8 KWORD PARTITION)

After verifying that the first part of Program memory is working, the bootstrap tries to find a good 8 Kword chunk of Program memory between address 0004000 and address 160000. This partition will be used to load the INIT P.IOC TEST from the TU58. If a large enough chunk of memory can't be found, a Program memory error is reported via the Fault lamp mechanism. A failure in this test indicates that the M.std module must be replaced.

4.5 TEST 4 - TU58 TESTS

The goal of the TU58 test is to find a working TU58 tape drive that contains a cassette with a bootable image. A bootable image is identified by a PDP-11 NOP instruction in the first word of the image. Once a suitable drive is found, the first 8 blocks of the tape are loaded into the 8 Kword partition found in test 3. The 8 blocks loaded consist of the first 5 blocks of the INIT P.IOC TEST, the RT-11 Volume ID block, and the first RT-11 directory segment on the tape. (The directory blocks are loaded at this time to save directory look-up time in the INIT P.IOC TEST).

The TU58 drives are tested in serial order until the first working drive is found that also contains a bootable image. The order of testing is as follows:

CSR	UNIT	(SYMBOLIC NAME)
177520	0	$SL1
177520	1	$SL1
177530	0	$SL2
177530	1	$SL2
177560	0	$SL0 (console)
177560	1	$SL0

If no suitable TU58 drive is found, a Tape error is displayed via the Fault lamp mechanism. An error table is maintained in Program memory addresses 000400 thru 000412, which remembers why each rejected TU58 drive failed. Each word of the table corresponds to one TU58 unit. The lower byte of each word contains a failure code assigned by the bootstrap software, which specifies why the drive was rejected (see definition of software error codes in section 4.5.1 below). If the TU58 controller sent a radial serial end packet with an error code, this code will be stored in the upper byte of the error word for that TU58 drive (see definition of TU58 end packet error codes in section 4.5.2 below). The error table is laid out as follows:

ADDRESS	CSR/UNIT
000400	177520/0
000402	177520/1
000404	177530/0
000406	177530/1
000410	177560/0
000412	177560/1

If a TU58 controller fails to synchronize, or fails its self-test, the software error code will be stored in unit 0's error table entry, and unit 1 of that controller will not be tried.

If the boot fails with a TU58 error, and a local terminal is available, the F-11's ODT feature may be used to examine the TU58 error table to determine why each TU58 drive failed the test (remember that the bootstrap tries all drives before declaring an error). Do the following to examine the TU58 error table:

a) Depress the BREAK key on the local terminal

b) The terminal should type out the address of the current instruction of the bootstrap, and then prompt for input with an '@' character.

c) Type '4ØØ/' (without the quotes)

d) The terminal should print the (octal) contents of address ØØØ4ØØ

e) Type 'linefeed' to examine the rest of the table.

f) The low byte of each word in the table contains a software error code assigned by the bootstrap. Refer to section 4.5.1 below for the meaning of the software error codes.

g) The high byte of each word of the table contains an error code sent by the TU58 controller in an end packet, or a zero if no error code was sent by the controller. Refer to section 4.5.2 below for the meaning of TU58 end packet error codes.

4.5.1 TU58 SOFTWARE ERROR CODES

These error codes are assigned by the bootstrap software to indicate why a particular TU58 drive was not used for booting.

CODE(octal)	MEANING
0	No error
1	NXM trap while accessing CSR's
2	TU58 sync sequence failed
3	TU58 failed its self-test
4	Drive did not contain bootable image
5	Checksum error in received packet (note 1)
6	Timeout on Transmitter Ready bit
7	Timeout on Receiver Done bit
10	Overrun or Framing error
11	Unknown packet type received

NOTE 1 - Checksum Errors - When a checksum error is detected in a received radial serial packet, the bootstrap will save the expected and actual checksum. The expected checksum (as computed from the bytes of the packet) is stored in Program memory location 000414, and the actual checksum (as received at the end of the packet) will be stored in location 000416. If more than one drive produces a checksum error, the checksum words pertain to the last drive tested. A local terminal is required to examine the expected and actual checksum, using the same procedure described in the preceeding section (4.5).

4.5.2 TU58 END PACKET ERROR CODES

The following table lists the error codes that the TU58 sends in radial serial end packets. The codes are listed as they will appear when the TU58 error table words are examined (I.E. the codes are in the upper byte of the word). All values are in the octal radix. The 'xx' portion of each word represents the software error code in the lower byte of each word.

ERROR WORD CONTENTS	MEANING
1774xx	Self-test failed
177Øxx	End of Tape encountered
175Øxx	Hard Read Error
174Øxx	Bad Unit Number Given
1734xx	No Cassette Mounted
1724xx	Write Protected
1674xx	Data Check Error
16ØØxx	Seek Error
1574xx	Motor Stopped
15ØØxx	Bad Op Code
1444xx	Bad Block Number Given

4.6 TEST 5 - TRANSFER CONTROL TO LOADED IMAGE

This part of the bootstrap is not actually a test, but is given a test number in case an error occurs in this section of code. The PDP-11 general registers are loaded with certain parameters (CSR and Unit of load device, base address and size of partition, etc.), then the image loaded from the TU58 is initiated by jumping to the first instruction. Any errors that occur in this part of the bootstrap would probably be unexpected traps or interrupts, caused by intermittent P.ioc or M.std failures. When the loaded image is started, the STATE lamp will be lit, and the INIT lamp will be turned off.

HSC50

Off-Line P.IOC Test

USER DOCUMENTATION

PRODUCT CODE:	XX-XXXXX-XX
PRODUCT NAME:	HSC50 OFFLINE P.IOC TEST
PRODUCT DATE:	12-APR-1983
MAINTAINER:	Diagnostic Engineering - Colorado (CX)
AUTHOR:	Mike Hare

TABLE OF CONTENTS

SECTION 1 - GENERAL INFORMATION

1.1 PROGRAM ABSTRACT

The OFFLINE P.IOC TEST is a REPAIR LEVEL DIAGNOSTIC, designed to test the HSC5Ø P.ioc module. All P.ioc logic not tested by the BOOTSTRAP is covered. The test runs in a STAND-ALONE environment, with no other HSC5Ø processes running. By providing specific error messages, the test aids in isolating P.ioc failures to a specific section of logic. If the entire test runs with no errors, the OFFLINE DIAGNOSTIC LOADER is read to memory from the TU58 and started.

1.2 SYSTEM REQUIREMENTS

Hardware Required:

a) P.ioc (processor) module with HSC5Ø Boot ROMS.
b) At least one M.std (memory) module.
c) TU58 controller with at least one working drive.
d) Terminal connected to P.ioc's console interface.

Software Required:

a) HSC5Ø Offline Diagnostic Cassette (TU58 media).

1.3 RELATED DOCUMENTS

a) HSC5Ø ROM Bootstrap User Document
b) HSC5Ø Offline Diagnostic Loader User Document
c) HSC5Ø Offline Memory Test User Document
d) HSC5Ø Offline K/P Memory Test User Document
e) HSC5Ø Offline K Test Selector User Document

1.4 HARDWARE ASSUMED TO BE WORKING

The Offline P.ioc Test is loaded by the HSC5Ø P.ioc ROM Bootstrap program. The bootstrap performs tests of the basic PDP-11 instruction set, the lower 2Ø48 bytes of Program memory, an 8 Kword partition in Program Memory, and the TU58 used by the bootstrap. Hence when the Offline P.ioc test begins to execute, most PDP-11 logic has been tested, and is assumed to be working. Likewise, the memory occupied by the test, and the TU58 used to load the test, are also assumed to be tested and working.

SECTION 2 - OPERATING INSTRUCTIONS

2.1 START-UP PROCEDURE

Follow the steps below to start the Offline P.ioc test:
(Refer to section 2.1.1 if this procedure fails)

1) Insert the HSC5Ø OFFLINE DIAGNOSTIC cassette into the TU58's Unit Ø drive. (left hand drive)

2) Power-on the HSC5Ø, then depress and release the INIT button on the HSC5Ø Front Panel.

3) The TU58 drive-motion LED should light within 1Ø seconds, indicating that the Bootstrap is loading the OFFLINE P.IOC TEST to Program Memory.

4) After tape motion stops, the P.ioc test begins testing the P.ioc module.

5) If no errors are detected, the tape-motion LED will light within 1Ø seconds, indicating that the test ran successfully and the OFFLINE DIAGNOSTIC LOADER is being read from tape to memory.

6) After about 6Ø seconds of loading, the OFFLINE LOADER will indicate it has loaded properly by displaying the following:

```
   HSC5Ø OFL Diagnostic Loader, Version nn-nn
   Radix=Octal,Data Length=Word,Reloc=ØØØØØØØØ
ODL>
```

Reference the Loader User Guide for further information on Loader features and commands, or type 'HELP' in response to the loader prompt.

2.1.1 IN CASE OF TROUBLE

If the test fails to load when the above Start-Up procedure is followed, refer to the items in the checklist below:

1) Try booting your tape from the TU58's Unit 1 Drive (right-hand drive). If your system has more than 2 drives, try booting your tape from one of the other drives.

2) Try booting another tape. If another tape will boot, the original tape is probably damaged or worn.

3) Inspect the cable between the TU58 and the P.ioc module. Insure the connectors are well seated, and inserted properly.

2.2 TEST PARAMETER ENTRY

The Offline P.ioc Test does not have any user-supplied parameters.

2.3 SETTING/CLEARING FLAGS

The Offline P.ioc Test does not have any user-modifiable flags.

2.4 PROGRESS REPORTS

The Offline P.ioc test does not issue any progress reports. If the test runs to completion with no errors detected, the Offline Diagnostic Loader is loaded and started. If the P.ioc test detects an error, an error report will be displayed.

2.5 TEST TERMINATION

Once initiated, the Offline P.ioc test can only be terminated by halting the machine and rebooting.

SECTION 3 - ERROR INFORMATION

3.1 ERROR MESSAGE FORMAT

All error messages produced by the Offline P.ioc test conform to the HSC5Ø Diagnostic Error Message format. (With 3 EXCEPTIONS as noted in section 3.1.1.). The first line of the error message contains general information concerning the error. The second line is a text message describing the nature of the error. Lines 1 and 2 are mandatory, and appear in all error messages. Line 3, and succeeding lines, of an error message display additional information, and are optional. They are displayed for certain errors that need to provide further information. All numeric information displayed with an error report is in the OCTAL radix. Listed below is a typical error message.

```
OPIO>ØØ:ØØ T#aaa E#bbb U-ØØØ        (mandatory line)
  <Text string describing error>    (mandatory line)
  MA -xxxxxxxx                      (optional line)
  EXP-yyyyyy                        (optional line)
  ACT-zzzzzz                        (optional line)

  WHERE:
   aaa      = Octal number denoting test that failed
   bbb      = Octal number denoting error detected
   xxxxxxxx = Failing address (if appropriate)
   yyyyyy   = Data expected   (if appropriate)
   zzzzzz   = Data found      (if appropriate)
```

The PDP-11 HALTS after printing the error message. R1 contains the address of the routine that loads the Offline Loader (Refer to section 3.1.3). RØ contains a pointer to a 9-word array which holds a copy of the general registers at the time the error was detected, plus the error and test number. The 9-word array is arranged as follows:

```
   word Ø  -  saved contents of RØ
   word 1  -  saved contents of R1
   word 2  -  saved contents of R2
   word 3  -  saved contents of R3
   word 4  -  saved contents of R4
   word 5  -  saved contents of R5
   word 6  -  saved contents of R6
   word 7  -  Test Number
   word 8  -  Error Number
```

After halting, the Offline P.ioc test may be restarted from the beginning by typing a 'P' (proceed) on the terminal.

3.1.1 PRIMITIVE ERROR REPORTS (HALT at 000400)

Certain errors are detected before the entire Offline P.ioc test has been loaded to memory. These errors can not use the normal error reporting method (text on the terminal). Instead, a more primitive error reporting method is used: General Register 0 (R0) is loaded with the error number, and the PDP-11 halts with the PC=000400. The test can only be repeated by rebooting.

The following errors use the primitive error reporting mechanism:

a) Unexpected trap while loading the remainder of the Offline P.ioc Test from the tape. (Error 000)

b) TU58 read error while loading the remainder of the Offline P.ioc Test from the tape. (Error 003)

c) Failure to find a tape directory entry for the Offline P.ioc Test. (Error 002)

3.1.2 WHAT TO DO IN THE EVENT OF A CRASH

Certain P.ioc failures may cause the program to 'Crash', (halt or loop indefinitely), without printing an error report. In the event this happens, use the 'BREAK' key to halt the PDP-11, then use 'PDP-11 ODT' to examine the following locations:

```
000422 (TSTNUM) holds current test number
000424 (ERRNUM) holds current error number
```

The test number should indicate the test causing the crash. The error number will ONLY BE VALID if the crash occurred while trying to report an error. Additionally, the contents of the PDP-11 General Registers (R0 thru R7), and the last few words on the stack, may help determine the exact point where the test is failing.

3.1.3 LOADING ODL AFTER AN ERROR REPORT

Certain memory module failures will be detected by the Offline P.ioc test. (E.G. No working control memory). The Offline Memory test is required to debug these failures. To force the Offline P.ioc test to load the Offline Loader, perform the following steps:

1. When the PDP-11 halts after printing an error report, type "R1/". The contents of R1 will be displayed.

2. Type a carriage return.

3. Type "R7/". The contents of R7 (the PC) will be displayed.

4. Enter the value that was in R1, followed by a carriage return.

5. Type a 'p'.

6. The Offline Loader will be read into memory and started.

7. Type 'Test Memory' followed by carriage return to start the Offline Memory test.

3.2 SPECIFIC ERROR MESSAGES

The following sections (3.2.x) list specific information about each of the errors emitted by the Offline P.ioc Test. Where feasible, hints about the cause of an error are provided. For further clues to the cause of an error, use the test number in the error report to reference one of the test descripitions in section 4.1 of this document.

3.2.1 ERROR 000

'Unexpected Trap or Interrupt'

A trap or interrupt occurred when none was expected, or an expected trap or interrupt used the wrong vector. This error can occur at any point in the Offline P.ioc test. The test number displayed with the error report can be used as a clue to the nature of the failure. For further clues to the cause of the problem, reference the proper test in section 4.1 of this document. If an incorrect vector is being used, the P.ioc's vector ROMs and associated circuitry are suspect.

3.2.2 ERROR 001

'OFL MONITOR look-up failure'

The TU58 used for the boot did not contain a directory entry for the Offline Diagnostic Loader program. The Offline Loader is normally read to memory after the Offline P.ioc test completes. The TU58 media may be corrupted, or may not have been written correctly. Use a different copy of the Offline Diagnostic Tape.

3.2.3 ERROR 002

'OFL P.IOC look-up failure'

The TU58 used for the boot did not contain a directory entry for the Offline P.ioc Test (this program). Only the first 5 blocks (2560 bytes) of the Offline P.ioc test are loaded by the bootstrap, the remaining blocks must be loaded from the tape. When the tape directory was searched, no entry was found for the P.ioc test (OFLPIO.SAV). This error is always reported by the Primitive Error Report mechanism (sec. 3.1.1) since the normal error reporter has not yet been loaded. The TU58 media may be corrupted, or may not have been written properly. Use a different copy of the Offline Diagnostic Tape.

3.2.4 ERROR 003

'Tape Read Error'

An error occurred while reading the TU58. If the error occurs every time the test is run, and always at the same point in the test, the tape media probably contains a bad spot. Try using a different copy of the Offline Diagnostic Tape.

If this error occurs intermittently (not every time the test is run), the TU58 media or the path between the P.ioc and the TU58 may be marginal. The problem may be worn media, a faulty Serial Interface on the P.ioc or TU58, or there may be excessive noise on the serial line between the P.ioc and the TU58. Try a different copy of the Offline Diagnostic Tape. If that cures the problem, worn media is indicated. If a different copy of the tape also fails intermittently, try booting from tape drive 1 (right-hand drive). If the tape reads correctly from drive 1, drive 0 may have dirty heads or an intermittent hardware problem. If the failure still occurs, either there is noise on the serial line between the tape and the P.ioc, or the Serial Interface on the P.ioc or TU58 has an intermittent hardware problem.

3.2.5 ERROR 004

'ASH'

The test of the ASH (Arithmetic Shift) instruction failed. Since all logic associated with this instruction is internal to the PDP-11 chip-set, a failure of the PDP-11 chip-set is probable. Replace the PDP-11 processor chip and try again.

3.2.6 ERROR 005

'ASHC'

The test of the ASHC (Arithmetic Shift Combined) instruction failed. Since all logic associated with this instruction is internal to the PDP-11 chip-set, a failure of the PDP-11 chip-set is probable. Replace the PDP-11 processor chip and try again.

3.2.7 ERROR 006

'DIV'

The test of the DIV (Divide) instruction failed. Since all logic associated with this instruction is internal to the PDP-11 chip-set, a failure of the PDP-11 chip-set is probable. Replace the PDP-11 processor chip and try again.

3.2.8 ERROR 007

'MUL'

The test of the MUL (Multiply) instruction failed. Since all logic associated with this instruction is internal to the PDP-11 chip-set, a failure of the PDP-11 chip-set is probable. Replace the PDP-11 processor chip and try again.

3.2.9 ERROR 010

'XOR'

The test of the XOR (Exclusive Or) instruction failed. Since all logic associated with this instruction is internal to the PDP-11 chip-set, a failure of the PDP-11 chip-set is probable. Replace the PDP-11 processor chip and try again.

3.2.10 ERROR 011

'SBC'

The test of the SBC (Subtract Carry) instruction failed. Since all logic associated with this instruction is internal to the PDP-11 chip-set, a failure of the PDP-11 chip-set is probable. Replace the PDP-11 processor chip and try again.

3.2.11 ERROR 012

'SXT'

The test of the SXT (Sign Extend) instruction failed. Since all logic associated with this instruction is internal to the PDP-11 chip-set, a failure of the PDP-11 chip-set is probable. Replace the PDP-11 processor chip and try again.

3.2.12 ERROR 013

'Tape RCV INT enable'

The test detected a problem with the Receiver Interrupt Enable in the P.ioc's Control and Status Register (CSR) for the tape drive used to boot the test. The interrupt enable either failed to set or clear. The expected and actual data indicate whether the enable failed to set or clear. After the PDP-11 halts, use 'PDP-11 ODT' to examine the contents of location 000412, which holds the address of the CSR that failed.

3.2.13 ERROR 014

'Tape Control and Status Reg'

The test detected a problem with one of the P.ioc's Control and Status Registers (CSR's) for the tape drive used to boot the test. One or more of the unused bit positions in the CSR is not reading as a 'zero'. The ACTual data displayed with the error report indicates which bit or bits failed to read as a 'zero'. After the PDP-11 halts, use 'PDP-11 ODT' to examine location 000412. Location 000412 contains the address of the first CSR for the tape unit in question. Use 'PDP-11 ODT' to examine this address, and the next 2 addresses in sequence. The binary representations below indicate which bits should read as 'zero' in each of the registers examined. The symbol 'x' indicates a bit position that is allowed to be either a 'zero' or a 'one'. The symbol '0' indicates a bit position that should always be a 'zero'.

```
First CSR  - x 000 000 0xx 000 000 (Receiver Status)
Second CSR - x xx0 000 0xx xxx xxx (Receiver Buffer)
Third CSR  - 0 000 000 0xx 000 x0x (Transmitter Status)
```

3.2.14 ERROR Ø15

'PSW Priority field'

The priority field (bits 5,6,7) of the Processor Status Word (PSW) did not contain the expected value. The ACTual data printed with the error report indicates the contents found, the EXPected data indicates the contents that should have been found. The paragraphs below provide additional information depending on the test that detected the error.

Test ØØ7 - This test attempts to set the Priority field to all 8 possible values. A failure in this test probably indicates a bit in the Priority field is stuck in the 'zero' or 'one' state. After the PDP-11 halts, examine the PSW by typing 'RS/'. Using the ACTual data from the error report as a guide, try setting or clearing the bit or bits in error. If this process uncovers a bit that is 'stuck', the PDP-11 CPU or MMU chip is the probable cause of the error, although one of the inputs or outputs of these chips may also be the cause. If the use of 'PDP-11 ODT' fails to exhibit any problems with the PSW's priority field, an interrmittent problem in the PDP-11 CPU or MMU chip may be the cause of the error.

Test Ø1Ø - This test causes a TU58 Transmitter interrupt. The PSW word of the interrupt vector should contain ØØØ34Ø, which sets the Priority to level 7 when the interrupt occurs. After the interrupt, the priority field was not set to level 7. Either the PSW word of the interrupt did not contain ØØØ34Ø, or else the interrupt process failed to load the PSW from the vector area. After the PDP-11 halts, use 'PDP-11 ODT' to examine location ØØØ414. Add 6 to the value in location ØØØ414, and examine the resulting address. (E.G. if location ØØØ414 contains ØØØ3ØØ, examine location ØØØ3Ø6). If the location examined contains the word ØØØ34Ø, the problem is likely to be a failure of the interrupt process to load the PSW from the vector. If the location does contain ØØØ34Ø, the Offline P.ioc test has somehow been corrupted, and the problem is not likely to be in the interrupt process.

Test Ø12 - This test sets up both a TU58 Transmitter interrupt and a Line Clock interrupt. The PSW in the line clock vector is set to ØØØ3ØØ (level 6), and the PSW in the transmitter interrupt vector is set to ØØØ24Ø (level 5). Then the processor priority is set to level Ø to allow interrupts. The Line Clock interrupt has a higher priority that the TU58 transmitter, so the line clock interrupt is expected first. After an interrupt is serviced, the priority should have been set to level 6 by the line clock interrupt service. The ACTual data displayed with the error report indicates the priority that was actually set by the interrupt. If the ACTual priority is level 5 (000240), the TU58 interrupt was serviced before the clock interrupt, indicating a problem in the interrupt priority logic, or a failure of the clock to interrupt. If the ACTual priority is level 0,1,2,3,4, or 7, the PSW may not have been loaded correctly by the interrupt service, or the PSW word of the line clock vector may not contain the correct value. After the PDP-11 halts, use 'PDP-11 ODT' to examine location 102. If location 000102 contains 000300, the interrupt service probably failed to load the PSW correctly. If location 000102 does not contain 000300, the Offline P.ioc test has somehow been corrupted, and the problem may not be in the interrupt processing logic.

Test 013 - This test lowers the priority to level 0, and expects a TU58 transmitter interrupt to occur. The PSW word of the interrupt vector is set to 000240, so the processor priority should be level 5 after the interrupt is serviced. The ACTual data displayed in the error report indicates the priority level that was actually set by the interrupt. If the ACTual priority was 000300 (level 6), a line clock interrupt was probably serviced before the TU58 interrupt. Since a line clock interrupt was just serviced in Test 012, another was not expected so soon. This may indicate that the line clock is interrupting continuously. If the ACTual priority was level 0,1,2,3,4, or 7, the interrupt process either failed to load the PSW from the TU58 transmitter interrupt vector area, or the PSW word of the TU58 interrupt vector was corrupted. After the PDP-11 halts, use PDP-11 ODT to examine location 000414. Compute the address of the TU58 Transmitter interrupt PSW word by adding 6 to the contents of location 000414, then examine the contents of that address. (E.G. If location 000414 contains 000300, examine location 000306). If this location contains 000240, the interrupt service probably failed to load the PSW. If the location does not contain 000240, the Offline P.ioc test has somehow been corrupted, and the problem may not be in the interrupt processing logic.

3.2.15 ERROR 016

'Tape XMIT INT enable'

The test detected a problem with the Transmitter Interrupt Enable in the P.ioc's Transmitter Status Register for the TU58 used to boot the test. The ACTual and EXPected data indicate whether the enable bit failed to set or clear. After the PDP-11 halts, use 'PDP-11 ODT' to examine location 000412. The address of the Transmitter Status Register is equal to the contents of location 000412 plus 4. E.G. If location 000412 contains 177520, the failing register is located at address 177524. Use PDP-11 ODT to set and clear the interrupt enable bit (bit 5) to determine whether the problem is constant or intermittent. The P.ioc logic that implements the interrupt enable bit for the TU58 transmitter is suspect.

3.2.16 ERROR 017

'Trap or Interrupt expected'

A trap or interrupt did not occur when it was expected. The test number supplied with the error message should be used to reference the proper test description in section 4.1 of this document. The test description indicates which interrupt or trap was expected, and supplies clues about the possible cause of the error.

3.2.17 ERROR 020

'RTI'

The test of the RTI (Return from Interrupt) instruction failed. Since all logic associated with this instruction is internal to the PDP-11 chip-set, a failure of the PDP-11 chip-set is probable. Try replacing the following chips (one at a time) : PDP-11 CPU, PDP-11 MMU. If neither of these two chips cures the problem, the interconnect between these chips is probably the cause of the error.

3.2.18 ERROR Ø21

'RTT'

The test of the RTT (Return from Trap) instruction failed. Since all logic associated with this instruction is internal to the PDP-11 chip-set, a failure of the PDP-11 chip-set is probable. Try replacing the following chips one at a time: PDP-11 CPU, PDP-11 MMU. If neither of these chips cures the problem, the interconnect between these two chips is probably the cause of the problem.

3.2.19 ERROR Ø22

'WAIT'

The test of the WAIT (Wait for Interrupt) instruction failed. Since all logic associated with this instruction is internal to the PDP-11 chip-set, a failure of the PDP-11 chip-set is probable. Replace the PDP-11 processor chip and try again.

3.2.2Ø ERROR Ø23

'JMP mode Ø'

The test of the JMP (Jump) instruction, address mode Ø, failed. A JMP mode Ø should trap thru location ØØØØØ4. The instruction either did not trap at all, or trapped thru an incorrect vector. Since all logic associated with this instruction is internal to the PDP-11 chip-set, a failure of the PDP-11 chip-set is probable. Replace the PDP-11 processor chip and try again.

3.2.21 ERROR 024

'Error Address Register'

The test generated a parity trap or NXM trap, and the Error Address Register failed to capture the 22-bit address which caused the trap. The Error Address Register is clocked by the occurrance of a parity trap or timeout, and is loaded directly from the P.ioc's 22-bit 'OUT BUS'. The Error Address is held in two locations:

```
  Address 17770024 - lower 16 bits of address
  Address 17770026 - upper 6 bits of address
     (bits 6 - 15 read as 'zeroes')
```

Either register may have failed to load, or may have been loaded incorrectly. Alternately, there may be a problem which does not allow a register to be read properly. The EXPected and ACTual data displayed with the error report show which bit or bits are failing.

3.2.22 ERROR 025

'MMU PSW mode bits'

The upper 4 bits of the PDP-11 Processor Status Word (PSW) are used by the Memory Management Unit (MMU). Bits 15 and 14 of the PSW hold the Current Mode (00=kernel, 11=user) of the MMU, and bits 13 and 12 hold the Previous Mode (00=kernel, 11=user). Error 025 indicates a failure of these mode bits. Use the test number supplied with the error message as a reference to one of the two paragraphs below.

Test 23 - This test attempts to load all possible combinations of 'ones' and 'zeroes' into the mode bits. One of the mode bits is probably stuck in a 'one' or a 'zero' state. Use the EXPected and ACTual data in the error message to determine the bit or bits in error. Since the logic controlling the 'MODE' bits is within the PDP-11 chip set, the PDP-11 CPU chip and the PDP-11 MMU chip are suspect. Replace these chips (one at a time) and retry the test. If replacing both chips does not cure the problem, the interconnect between the PDP-11 CPU and MMU is the next most likely source of the error.

Test 040 - This test causes a NXM trap while in User mode. Both the Current and Previous mode fields in the PSW saved by the NXM trap should indicate User mode. (I.E. Upper 4 bits of the PSW = 1111). The ACTual data displayed with the error message indicates which bits were not set. The PDP-11 CPU chip and MMU chip are the prime suspects.

3.2.23 ERROR Ø26

'MMU User/Kernel Stack Pointer'

The User and Kernel stack pointers (SP's) are interfering with each other. When the CURRENT MODE bits of the PSW=ØØ, the Kernel SP should be enabled. Likewise, when the CM bits = 11, the User SP should be enabled. The test determined that the User SP was being used when the CM was Kernel mode, or the Kernel SP was being used when the CM was User mode. The PDP-11 CPU chip and the PDP-11 MMU chip are the prime suspects, although the interconnect between these chips may also be the cause of the problem.

3.2.24 ERROR Ø27

'MMU Status Reg Ø'

The test detected a failure in the Memory Management Unit's Status Register Ø (address 17777572). Status Register Ø (SRØ) is used to enable and disable the MMU, and contains error bits that remember the reason for a MMU abort. For the PDP-11 MMU, the following bits are implemented:

```
bit ØØ - enables the MMU when set to '1'

bits Ø1 thru Ø3 - remembers MMU page when error occurs.

bits Ø5 and Ø6 - remembers mode when error occurs
                 (ØØ=Kernel, 11=User)
bit 13 - set when 'Read-Only-Access' abort

bit 14 - set when 'Page-Length-Error' abort

bit 15 - set when 'Non-Resident-Page' abort
```

The EXPected and ACTual data displayed with the error message specify which bit or bits in the register are not working properly. Most of the logic associated with SRØ is contained in the PDP-11 CPU chip and the PDP-11 MMU chip, so these two chips are the prime suspects.

3.2.25 ERROR 030

'MMU Status Reg 2'

The test detected an error in the MMU's Status Register 2 (address 17777576). Status Register 2 (SR2) is loaded with the 16-bit virtual address at the beginning of each PDP-11 instruction fetch, but is not updated if the instruction fetch fails. On the occurrance of any MMU abort, the contents of the register are frozen. The EXPected and ACTual data displayed with the error report indicate which bit or bits of the register are failing. If the ACTual and EXPected data differ by more than a single bit, it is likely that the register failed to 'freeze' when a MMU abort occurred. The PDP-11 MMU chip is the most likely cause of the failure, but the PDP-11 CPU chip could also be at fault. Use the test number from the error report as a reference to the proper test description in section 4.1 for further info.

3.2.26 ERROR 031

'MMU User/Kernel PAR'

This error occurs while performing a bit test of the MMU's Page Address Registers (PAR's). A bit or bits in one of the PAR's is stuck in a 'zero' or 'one' state. The EXPected and ACTual data displayed with the error report indicate which bit or bits are failing. Since the PAR's are contained within the PDP-11 MMU chip, the first thing to replace is the MMU. The PDP-11 CPU chip, or the interconnect between the MMU and CPU are also possible causes of the failure.

3.2.27 ERROR 032

'MMU User/Kernel PDR'

This error occurs while performing a bit test of the MMU's Page Descriptor Registers (PDR's). A bit or bits in one of the PDR's is stuck in a 'zero' or 'one' state. The EXPected and ACTual data displayed with the error report indicate which bit positions are failing. Since the PDR's are contained within the PDP-11 MMU chip, the first thing to replace is the MMU. The PDP-11 CPU chip, or the interconnect between the MMU and CPU chip, are also possible causes of the failure.

3.2.28 ERROR 033

'MMU Relocation Adder'

This error is detected while testing the ability of the MMU to relocate. After using the MMU to write to a test location, the location was read without Memory Management and found to contain incorrect data. The MMU's relocation adder and associated circuitry is probably at fault. The PDP-11's MMU chip is the prime suspect, but the failure could also be caused by a faulty PDP-11 CPU chip, or by the interconnect between the MMU and CPU chips.

3.2.29 ERROR 034

'MMU PDR W bit'

A failure was detected with the 'Write' (W) bit in a MMU Page Descriptor Register (PDR). The bit should clear when the PDR is loaded, and should only set if there is a write access to the page the PDR describes. The bit either failed to clear when the PDR was written, or else failed to set when a location in that page was written. The EXPected and ACTual data in the error report indicate whether the bit failed to set or clear. The 'W' bit circuitry is contained within the PDP-11's MMU chip, thus the MMU chip is the prime suspect for the failure. Other less likely causes of the error include the PDP-11 CPU chip, and the interconnect between the CPU and MMU chips.

3.2.30 ERROR 035

'MMU Access Mode'

A MMU access mode failure occurred. Refer to one of the two paragraphs below, depending on the test number displayed in the error report.

Test 032 - This is a test of the Non-Resident access mode. The Page Descriptor Register for a particular page was set to indicate the page was not resident. (When the access mode is set non-resident, all references to that page should cause a MMU abort.) Then the test performed an 'INC' instruction to a location in the page. Although an abort occurred when the location was referenced, the 'INC' instruction still added one to the location. The PDP-11 MMU chip is the most likely cause of the failure, but the PDP-11 CPU or the interconnect between the CPU and MMU could be at fault. (continued on next page)

3.2.3Ø ERROR Ø35 (continued)

Test Ø33 - This is a test of the 'Read-Only' access mode. The Page Descriptor (PDR) for a particular page was set to indicate the page was 'Read-Only' (I.E. Locations within the page can be read, but writes are prohibited). Then the test performed an INC (increment) instruction to a location within the page, which should cause a MMU trap. However, no MMU trap occurred. Since the access-mode logic is contained within the PDP-11 MMU chip, the MMU chip is the major suspect. The PDP-11 CPU chip, or the interconnect between the MMU and the CPU are also possible causes of the failure.

3.2.31 ERROR Ø36

'MMU RTI in User mode'

The test executed a RTI instruction when the Memory Management Unit (MMU) was in User mode. The mode bits and priority bits in the Processor Status Word (PSW) should not be affected by the RTI. However, after the RTI the mode or priority bits were found to be changed. The EXPected and ACTual data displayed in the error report indicate which bits were affected. Most of the logic involved is contained within the PDP-11 MMU chip, so the MMU chip is the most likely cause of the failure. If replacing the MMU chip does not cure the problem, the PDP-11 CPU chip or the connections between the CPU and MMU may be at fault.

3.2.32 ERROR Ø37

'MFPI'

The test of the MFPI (Move From Previous Space) instruction failed. Since all logic associated with this instruction is internal to the PDP-11 chip-set, a failure of the PDP-11 chip-set is probable. The PDP-11 CPU chip and the PDP-11 MMU chip are suspect.

3.2.33 ERROR Ø4Ø

'MTPI'

The test of the MTPI (Move To Previous Space) instruction failed. Since all logic associated with this instruction is internal to the PDP-11 chip-set, a failure of the PDP-11 chip-set is probable. The PDP-11 CPU and PDP-11 MMU chips are suspect.

3.2.34 ERROR Ø41

'Switch/Display Reg'

The P.ioc's Switch/Display register test failed. Either a bit is stuck in the lower 6 bits of the register, or the upper 3 bits of the register do not read as 'zeroes'. The EXPected and ACTual data displayed in the error message indicate which bits are at fault. The Switch/Display register is located at address 1777ØØ42.

3.2.35 ERROR Ø42

'P-CSR Parity Invert (high byte)'

The 'High-Byte Parity Invert' bit in the P.ioc's Control and Status Register (address 1777ØØ4Ø) either failed to set, or failed to clear after being used. The ACTual and EXPected data given in the error report indicate whether the bit failed to set or clear. If the bit failed to set, check out the input to the P.ioc CSR. If the bit failed to clear, check out the logic that is supposed to clear this bit after a write operation occurs.

3.2.36 ERROR Ø43

'P-CSR Parity Invert (low byte)'

The 'Low-Byte Parity Invert' bit in the P.ioc Control and Status register (address 1777ØØ4Ø) failed to set, or failed to clear after being used. The ACTual and EXPected data displayed in the error message indicate whether the bit failed to set or clear. If the bit failed to set, check out the input to the P.ioc CSR. If the bit failed to clear, check the logic that is supposed to clear this bit after a write operation occurs.

3.2.37 ERROR Ø44

'Window Index Reg'

The test of the Window Index Register (address 1777ØØ2Ø) failed. The test loads all possible combinations of 'ones' and 'zeroes' into the lower 7 bits of the register, and checks that the upper 9 bits always read as 'zeroes'. The EXPected and ACTual data given with the error report indicate which bits are failing.

3.2.38 ERROR Ø45

'Window Address Reg, bits 2 thru 8'

This error indicates a problem with bits 2 thru 8 of the Window Address Registers. The test puts all possible combinations of 'ones' and 'zeroes' into these bits, and insures that they can be read back properly. The EXPected and ACTual data displayed in the error message indicate the failing bit or bits. Refer to the test number given with the error report, then turn to the proper test description in section 4.1 of this document for further clues to the problem.

3.2.39 ERROR Ø46

'Window Address Reg, bits 9 thru 11'

This error indicates a problem with bits 9 thru 11 of the Window Address Registers. The test puts all possible combinations of 'ones' and 'zeroes' in these 3 bits and insures that they are read back properly. The EXPected and ACTual data displayed in the error message indicates which pattern failed. Refer to the test number in the error report, then turn to the proper test description in section 4.1 for further clues to the problem.

3.2.4Ø ERROR Ø47

'Window Memory - Last pattern wrong'

While performing a modified form of the Moving Inversions Memory Test on the Window Memory, checking a location revealed that the previous contents were changed. The EXPected data given with the error report is the pattern that was supposed to be in the window. The ACTual data is the pattern that was found. This could indicate a stuck bit in one of the windows, or there may be a dual addressing problem in the window memory. Use the test number given in the error report to find the proper test description in section 4.1 of this document.

3.2.41 ERROR Ø5Ø

'Window Memory - New pattern wrong'

While performing a modified form of the Moving Inversions Memory test on the Window Memory, checking a location that was just written revealed that the new data was not stored properly. The EXPected and ACTual data given in the error report indicate the bit or bits in error. This either indicates a stuck bit or bits in the window memory, or possibly marginal timing in the window memory circuit. Using the test number from the error report as a guide, refer to the proper test description in section 4.1 for further clues to the cause of the error.

3.2.42 ERROR Ø51

'Control Mem Data Tranceivers'

The test found a problem in the Control Memory Data Tranceivers. The test performed a Non-Memory-Access (NMA) write to Control Memory, causing the data pattern used to be latched into the tranceivers. (When the NMA bit is used, Control Memory itself is not accessed). Then an NMA read was performed to read the pattern back from the tranceivers. The EXPected and ACTual data displayed in the error report indicate which bit or bits were incorrect when read back.

3.2.43 ERROR Ø52

'K Status Reg'

The test found an error in the 'K Status Register' logic. The test loads the P's own status with a pattern, then loads the 'K Init Register' with the proper value to allow reading the P's own status. When the status was read back, it was incorrect. The EXPected and ACTual data displayed in the error report indicates which bit or bits were incorrect.

3.2.44 ERROR Ø53

'P-CSR NMA bit'

The NMA bit in the P.ioc's Control and Status Register failed to set, or failed to clear after being used. The EXPected and ACTual data displayed in the error message indicates whether the bit failed to set or clear. If the bit failed to set, check out the input to the P.ioc's status register. If the bit failed to clear, check the logic that is supposed to clear this bit after a Control or Data memory reference.

3.2.45 ERROR Ø54

'Data Mem Address Reg'

The test found a problem in the Data Memory Address Register. The Data Memory Address Register is supposed to capture the last Data Memory address used. The Data Memory Address Register is located at addresses 1777ØØ5Ø (lower 16 bits of last Dmem address) and 1777ØØ52 (upper 6 bits of Dmem address). The expected and actual data displayed in the error report indicate which bit or bits are failing. Either the register is not being loaded, or is being loaded with the wrong address.

3.2.46 ERROR Ø55

'Data Memory Data Tranceivers'

The test found a problem in the Data Memory Data Tranceivers. The test performed a Non-Memory-Access write to Data Memory, causing the data pattern to be latched into the Data Memory Data Tranceivers. (When a NMA write is performed, the Data Memory itself is not accessed). Then an NMA read was performed to read the pattern back from the tranceivers. The EXPected and ACTual data given with the error message indicate which bit or bits were incorrect when read back.

3.2.47 ERROR Ø56

'P-CSR Control Mem INT enable'

The test found a failure with the 'Control Memory Interrupt Enable' bit in the P.ioc's Control and Status Register (address 1777ØØ4Ø). Either the bit failed to set, or failed to clear itself after an interrupt was generated. The EXPected and ACTual data displayed with the error message indicate whether the bit failed to set or clear. If the bit failed to set, check the input to the P.ioc CSR. If the bit failed to clear, check the circuit that is supposed to clear the bit after an interrupt is generated.

3.2.48 ERROR Ø57

'P-CSR Control Mem Lock bit'

The test found a failure with the 'Control Memory Lock Cycle' bit in the P.ioc's Control and Status Register (address 1777ØØ4Ø). Either the bit failed to set, or failed to clear itself after the lock cycle was generated. The EXPected and ACTual data supplied in the error message indicate whether the bit failed to set or clear. If the bit failed to set, check the input to the P.ioc's CSR. If the bit failed to clear, check the circuit that is supposed to clear the lock bit after a lock cycle is generated.

3.2.49 ERROR Ø6Ø

'Control Mem Lock Cycle'

The test detected an error in the operation of a 'Control Memory Lock Cycle'. Use the test number supplied in the error report to reference the proper test description in section 4.1 for more information about the failure, and possible clues as to its cause.

3.2.5Ø ERROR Ø61

'Tape Sync failure'

The test attempted to synchronize with the TU58 and failed. This failure can indicate a marginal noise problem on the lines between the P.ioc and the TU58, or an intermittent problem in the UART on the P.ioc or the TU58. Changing the TU58 media, or changing to Drive 1 of the TU58 probably will not help, since a media problem is not indicated.

3.2.51 ERROR Ø62

'Test out of sequence'

The program detected an attempt to execute a test out of the normal sequence. The EXPected data in the error report shows the test number that should have preceeded the current test, the ACTual data shows the test number that actually preceeded the current test.

Tests are designed to execute in numerical sequence. I.E. Test Ø,1,2,....66. A variable within the program (TSTNUM) keeps track of the current test number. At the beginning of each test, a sequence check is made to insure that the previous contents of the TSTNUM variable are just one less than the new test number. If not, the program somehow skipped over a test, or tried to re-execute a test that was already performed. The error probably occurred within the test indicated by the ACTual data in the error report. Although this problem is difficult to categorize, the PDP-11 CPU or MMU chip is the probable cause of the failure.

3.2.52 ERROR Ø63

'Unexpected MMU Abort'

The PDP-11 Memory Management Unit (MMU) produced an unexpected abort (trap thru ØØØ25Ø). After the PDP-11 halts, use 'PDP-11 ODT' to examine the MMU status registers to determine the reason for the abort. The paragraphs below explain how to use the contents of the status registers to determine the cause of the trap.

MMSRØ - (address 17777572) - This register contains the error bits that describe the reason for the trap.

```
  Bit 15 - set for 'Non-Resident-Page' abort
  Bit 14 - set for 'Page-Length-Error' abort
  Bit 13 - set for 'Read-Only-Access' abort
  Bits 5 and 6 - MMU mode at time of abort
                ØØ=Kernel, 11=User
  Bits Ø1 thru Ø3 - MMU page causing abort
  Bit Ø - set when MMU is enabled
```

MMSR2 - (address 17777576) - This register contains the virtual PC of the instruction that caused the abort.

MMSR3 - (address 17772516) - This register should contain ØØØØ2Ø, which enables 22-bit memory mapping. If it does not, use PDP-11 ODT to determine if the 22-bit mapping bit is stuck in the 'zero' state.

Each of the MMU tests in this program contain a list of the contents of the Kernel and User Page Address Registers (PAR's) and Page Descriptor Registers (PDR's). Refer to the program listing to determine the expected contents of the PAR/PDR pair for the page causing the trap (see MMSRØ above).

```
  Kernel PAR's - 1777234Ø thru 17772356
  Kernel PDR's - 177723ØØ thru 17772316
  User PAR's   -  1777764Ø thru 17777656
  User PDR's   -  177776ØØ thru 17777616
```

3.2.53 ERROR 064

'MMU wrong PC, NXM in User mode'

The test caused an NXM trap while the MMU was enabled in User mode. The NXM trap is supposed to cause a MMU abort while pushing the NXM trap PC and PSW on the User stack. The MMU abort is supposed to cause the PC and PSW to be pushed on the Kernel stack instead. The test determined that the correct PC was not pushed on the Kernel stack. The logic being tested is mostly within the PDP-11's MMU chip, thus the MMU chip is the most likely cause of the problem. Other less likely causes are the PDP-11 CPU chip, and the inputs and outputs of the MMU chip and the CPU chip.

3.2.54 ERROR 065

'Parity Trap'

An unexpected parity trap occurred. The test number in the error report indicates the test where the trap occurred. After the PDP-11 halts, use 'PDP-11 ODT' to examine the error address register to determine the memory address that caused the trap. The lower 16 bits of the Error Address will be in location 17770024, and the upper 6 bits of the failing address will be in the lower 6 bits of location 17770026. The list below indicates the address ranges of the various HSC50 memories.

```
Program Memory - 00000000 thru 13777777

Data Memory    - 14000000 thru 15777777

Control Memory - 16000000 thru 16777777
```

SECTION 4 - TEST SUMMARIES

4.1 TEST SUMMARIES

The following sections describe each of the tests in the Offline P.ioc Test. Emphasis is on the hardware that is checked by each test.

4.1.1 TEST ØØ1

Test ØØ1 is a test of the PDP-11's ARITHMETIC SHIFT (ASH) instruction. The test executes several ASH instructions and checks for proper condition codes and shift results. The logic being tested is within the PDP-11 CPU chip.

4.1.2 TEST ØØ2

Test ØØ2 is a test of the PDP-11's ARITHMETIC SHIFT COMBINED (ASHC) instruction. The test executes several ASHC instructions and checks for the correct condition codes and shift results. The logic being tested is within the PDP-11 CPU chip.

4.1.3 TEST ØØ3

Test ØØ3 is a test of the PDP-11's DIVIDE (DIV) instruction. Six different combinations of dividends and divisors are used to check out the DIV logic for correct numeric results and proper setting of the condition codes. The logic being tested is within the PDP-11 CPU chip.

4.1.4 TEST ØØ4

Test ØØ4 is a test of the PDP-11's MULTIPLY (MUL) instruction. Five different combinations of multiplicands and multipliers are used to check the MUL logic for correct numeric results and proper setting of the condition codes. The logic being tested is within the PDP-11 CPU chip.

4.1.5 TEST ØØ5

Test ØØ5 contains tests of the PDP-11's EXCLUSIVE OR (XOR), SUBTRACT CARRY (SBC), and SIGN EXTEND (SXT) instructions. The instructions are checked for the correct numeric results and condition code settings. The logic being tested is within the PDP-11 CPU chip.

4.1.6 TEST 006

Test 006 is a test of certain bits in the Control and Status Registers (CSR's) of the TU58 interface used to boot the Offline P.ioc test. First the Receiver Interrupt Enable in the Receiver Status Register is set and cleared to insure this bit is not stuck in either state. Then all the unused bits in the Receiver Status, Receiver Buffer, and Transmit Status registers are checked to insure the unused bits read as 'zeroes'. The logic being tested is all contained on the P.ioc module.

4.1.7 TEST 007

Test 007 is a test of the Priority field in the PDP-11's Processor Status Word (PSW). The 3-bit priority field is loaded with all 8 possible bit combinations to insure that no bits in the field are stuck in a 'zero' or 'one' state, or shorted to another bit. Before the test is started, the Line Clock Interrupt vector is set to point to a Return From Interrupt (RTI) instruction, because the clock will interrupt as soon as the priority is level 5 or lower. The occurrance of the line clock interrupt should be transparent to the test (I.E. causes no harm). The logic being tested is within the PDP-11's CPU chip and MMU chip.

4.1.8 TEST 010

Test 010 sets up a Transmitter Interrupt on the serial interface that controls the TU58 used to boot, and sets the corresponding interrupt enable bit. Then the interrupt enable bit is checked to insure it was set. Next the test checks that the Transmitter Interrupt (Level 4) is not honored when the PDP-11's priority is set to levels 7, 6, 5, or 4. The test then verifies that the interrupt is honored when the PDP-11's priority is level 3. After the interrupt is processed, further checks are made to insure that the interrupt set the PSW correctly, and that the Transmitter Interrupt Enable can be cleared.

If the interrupt occurs at the wrong priority level, the PDP-11 CPU or MMU chip may be at fault. If the interrupt fails to occur at all, the serial interface and associated logic should be checked. If the Transmitter Interrupt Enable fails to set or clear, look for the problem in the P.ioc register that contains the interrupt enable bit. If the transmitter interrupt uses the wrong vector, an ERROR 000 is declared. This problem may be caused by the P.ioc ROMs which generate the interrupt vector address.

4.1.9 TEST 011

Test 011 checks the operation of the Line Clock interrupt, which is a level 6 interrupt. The test first checks that the interrupt is not processed when the PDP-11's priority is set to levels 7 and 6. Then the priority is dropped to level 5, and the test checks that the clock interrupt is honored at that level.

If the line clock interrupts when the PDP-11's priority is set to levels 6 or 7, the problem probably lies within the PDP-11 CPU or MMU chip. If the clock fails to interrupt at level 5, the problem could be in the P.ioc logic which passes interrupt requests to the PDP-11, or the 75 Hertz clock might be faulty, or the PDP-11 CPU or MMU chip may be at fault.

4.1.10 TEST 012

Test 012 tests the interaction of a simultaneous Line Clock Interrupt and TU58 Transmitter Interrupt. The test first sets up a TU58 transmitter interrupt, then waits at least 15 milli-seconds to guarantee that a clock interrupt is generated. The PDP-11's interrupt priority is initially set to level 6, so neither interrupt should be processed. Then the PDP-11's priority is lowered to level 0 to enable all interrupts. The test checks that the clock interrupt (level 6) is honored before the TU58 interrupt (level 4).

If either interrupt is processed when the priority is set to level 6, the test displays an ERROR 00 (unexpected interrupt). This would probably indicate a problem with the PDP-11 CPU or MMU chip. If the clock fails to interrupt, there might be a problem in the P.ioc logic that presents the clock interrupt to the PDP-11 CPU chip.

4.1.11 TEST 013

Test 013 is a continuation of test 012 (see preceeding section). After servicing the Clock Interrupt, the test lowers the PDP-11's priority to level 0 to enable interrupts. The TU58 transmitter interrupt should be processed immediately after the priority is lowered.

If the TU58 fails to interrupt when the priority is set to level 0, the problem may be in the P.ioc logic that presents the interrupt requests to the PDP-11, or might be in the P.ioc's serial line unit circuitry. It is also possible that an PDP-11 CPU or MMU chip failure might cause the interrupt to be ignored. If another clock interrupt is serviced before the transmitter interrupt, the clock interrupt may be stuck in the 'on' state.

4.1.12 TEST 014

Test 014 is a test of the PDP-11's TRAP instruction. A TRAP instruction should result in a trap thru vector 000034.

If the TRAP instruction fails to trap at all, an ERROR 017 is declared. If the TRAP instruction uses the wrong vector, an ERROR 000 is declared. Either problem is most likely caused by a failure of the PDP-11 CPU or MMU chip.

4.1.13 TEST 015

Test 015 is a test of the PDP-11's EMT (Emulator Trap) instruction. An EMT instruction should trap thru vector 000030.

If the EMT instruction fails to trap, an ERROR 017 is declared. If the EMT instruction traps, but uses the wrong vector, an ERROR 000 is declared. Either failure is most likely caused by the PDP-11 CPU or MMU chip.

4.1.14 TEST 016

Test 016 is a test of the PDP-11's BPT (Break Point Trap) instruction. A BPT instruction should trap thru vector 000014.

If the BPT fails to trap, an ERROR 017 is declared. If the BPT traps thru the wrong vector, an ERROR 000 is declared. In either case, the PDP-11's CPU or MMU chip is the most likely cause of the problem.

4.1.15 TEST 017

Test 017 is a test of the PDP-11's IOT (I/O Trap) instruction. An IOT instruction should trap thru vector 000020.

If the IOT fails to trap, an ERROR 017 is declared. If the IOT instruction traps, but uses the wrong vector, an error 000 is declared. In both cases, the PDP-11 CPU chip or MMU chip is the most likely cause of the problem.

4.1.16 TEST 020

Test 020 is a test of the PDP-11's RTI (Return From Interrupt), RTT (Return From Trap), WAIT (Wait for Interrupt), and JMP (Jump) instructions. The RTI and RTT instructions should pop the first word on the stack to the PDP-11's Program Counter (PC) and the second word on the stack should be popped to the PDP-11's Processor Status Word (PSW). The WAIT instruction should hang up the PDP-11 until an interrupt occurs. The JMP instruction is tested with a "mode 0" (register) operand, which should cause a trap thru vector 000004.

If any of these instructions fail, the most probable cause of the failure is the PDP-11's CPU or MMU chip. If the WAIT instruction fails, the PDP-11 may hang until a 'break' is typed on the operator's terminal.

4.1.17 TEST 021

Test 021 creates a Non-Existent-Memory (NXM) trap and insures that a trap thru vector 000004 is produced. The test also checks that the address which caused the NXM is latched into the P.ioc's ERROR ADDRESS REGISTER (addresses 17700024 and 17700026).

If the NXM fails to cause a trap an ERROR 017 is declared. The PDP-11 CPU chip is the most likely cause of this problem.

If the NXM traps, but uses the wrong vector, an ERROR 000 is declared. This problem is also most likely to be caused by a faulty PDP-11 CPU chip.

If the NXM trap does not latch the proper address in the ERROR ADDRESS REGISTER, an ERROR 024 is declared. This problem is most likely caused by the P.ioc logic associated with the ERROR ADDRESS REGISTER.

4.1.18 TEST 022

Test 022 is a test of the PDP-11's Stack Overflow Trap. The PDP-11 should cause a trap thru vector 000004, whenever the stack pointer (SP) is decremented below address 000400. The test sets the SP to 000376, and pushes one word on the stack to cause a trap.

If the PDP-11 fails to trap an ERROR 017 is declared. If the PDP-11 traps, but uses the wrong vector, an ERROR 000 is declared. In both cases, the PDP-11 CPU chip is the prime suspect.

4.1.19 TEST 023

Test 023 checks the MMU MODE bits in the PDP-11's Processor Status Word (PSW), and tests for interaction between the User and Kernel Stack Pointers. The mode bits (PSW bits 15 thru 12) are tested by loading and reading back all 16 possible bit-patterns (0000 thru 1111). The User and Kernel stack pointers are tested for interaction by setting the kernel SP and user SP, then checking the kernel SP to insure it was not changed when the user SP was set.

If the mode bits fail, an ERROR 025 is declared. The EXPected and ACTual data displayed with the error report show which mode bits failed. If the user and kernel SP interaction test fails, an ERROR 026 is declared. Errors 025 and 026 are both most likely to be caused by a problem in the PDP-11 MMU chip, but the PDP-11 CPU chip is also a possible cause.

4.1.20 TEST 024

Test 024 checks all the MMU registers for time-outs. Each of the MMU registers is read to insure that each register will respond to its address. The test checks the MMU status registers (SR0,SR1,SR2,SR3), the User and Kernel Page Descriptor Registers (PDR's), and the User and Kernel Page Address Registers (PAR's).

If any of these registers fails to respond to its address, a Non-Existent-Memory (NXM) trap will occur, and an ERROR 000 will be reported. The PDP-11 MMU chip is the most probable cause of this failure, but the PDP-11 CPU chip or the connections between the MMU and CPU could be at fault.

4.1.21 TEST 025

Test 025 checks the 3 error bits in the MMU's Status Register 0 (SR0), and tests that Status Register 2 (SR2) tracks the virtual address until one of the error bits in SR0 is set. First all three error bits are set in SR0, and a RESET instruction is issued to insure that RESET clears the error bits. Then SR2 is checked to insure that it is tracking the virtual address of each instruction executed when no error bits are set in SR0. Next each of the three error bits in SR0 is set in turn. The test checks that the error bit can be set, and tests that SR2 locks-up (stops updating) when the error bit is set.

(Continued on following page)

4.1.21 (Continued)

If the test detects a problem with the error bits in Status Register 0, an ERROR 027 is declared. The EXPected and ACTual data displayed with the error report indicate which bit or bits are failing. If the test detects a problem with Status Register 2, an ERROR 030 is displayed. In either case, the PDP-11 MMU chip is the most likely cause of the error. Other less likely causes of the error include the PDP-11 CPU chip, and the connections between the MMU and CPU chips.

4.1.22 TEST 026

Test 026 uses a WALKING ZERO pattern (15 'ones' and 1 'zero') to check out the MMU Kernel and User Page Address Registers (PAR's). Each of the 16 PAR's is loaded with all 16 possible patterns and read back to insure that no bits are picked up or dropped.

If any of the PAR's picks-up or drops a bit, an ERROR 031 is displayed. The EXPected and ACTual data indicate which bits are at fault. The PDP-11 MMU chip is the most probable cause of any failures detected, since the PAR's are contained within the MMU chip.

4.1.23 TEST 027

Test 027 uses a WALKING ZERO pattern (15 'ones' and 1 'zero') to check out the MMU Kernel and User Page Descriptor Registers (PDR's). Each of the 16 PDR's is loaded with all 16 possible patterns and read back to insure that no bits were picked-up or dropped. Bits 15,7,6,5,4, and 0 are not loaded with 'ones' since these bits are not implemented.

If any of the PDR's pick-up or drop bits. an ERROR 032 is reported. The EXPected and ACTual data in the error message indicate the bits that are failing. The PDP-11 MMU chip is the most likely cause of any failures, since the PDR's are contained within the MMU chip.

4.1.24 TEST 030

Test 030 enables the Memory Management Unit (MMU) and tests the relocation logic. The Kernel and User Page Address Registers (PAR's) and Page Descriptor Registers (PDR's) are set to map the test itself, and to map areas of memory to be tested. Then relocation is tested in both User and Kernel mode. As part of the test, a write is performed to each page that is mapped, to test the 'WRITE' (W) bit in the PDR's. The W bit should clear when the PDR is loaded, and should set if there is any write to the page the PDR describes.

If any of the relocation tests fail, an ERROR 033 is declared. If the W bit tests fail, an ERROR 034 is reported. In either case, the PDP-11 MMU chip is probably the cause of the error.

NOTE: This test has a better than average chance of 'crashing' the program. In order to test relocation, the MMU must also be used to map the program itself. All possible tests of the MMU that do not require mapping enabled have already been performed before this test is started, but when the MMU is used to map the program, failures in the MMU may cause a crash.

4.1.25 TEST 031

Test 031 tests special cases of the MMU's WRITE (W) bit. Each of the 16 Page Descriptor Registers (PDR's) contains a W bit that sets whenever any location within that page is written. Normal operation of the W bit has already been tested in test 030, but this test checks out two special cases. First the W bit in Kernel PDR 7 is checked to NOT SET when writing to MMU Status Register 0 (which is in the I/O page). Then the W bit is checked to SET when a write access is aborted by a Non-Existent-Memory (NXM) trap.

If the W bit fails to work as expected, an ERROR 034 is reported. The EXPected and ACTual data indicate whether the bit failed to set, or set when not expected to set. If the PDP-11 fails to trap when a Non-Existent-Memory address is referenced, an ERROR 017 is declared. The PDP-11 MMU chip is the most probable cause of both types of failure. Other less likely causes are the PDP-11 CPU chip, and the interconnect between the MMU and CPU chips.

4.1.26 TEST Ø32

Test Ø32 checks for the proper operation of the MMU's Non-Resident-Access mode. The access mode bits in one of the Page Descriptor Registers (PDR's) are set to the 'Non-Resident' state. Then the page is accessed, which should cause a MMU trap. After the trap, Status Register Ø (SRØ) should have the 'NON-RESIDENT' error bit set, and should indicate the proper page and mode (user/kernel). Status Register 2 (SR2) should have locked up with the virtual address of the instruction that caused the trap.

If the access to the non-resident page fails to cause a trap, an ERROR Ø17 is declared. If the access caused an abort, but still accessed the page, an ERROR Ø35 is displayed. If SR2 failed to lock-up at the proper address, an ERROR Ø3Ø is reported. If SRØ did not contain the proper error bit, mode, or page, an ERROR Ø27 is declared. In all cases the PDP-11 MMU chip is the most likely cause of the failure. Other possible, but less likely, causes of the error are the PDP-11 CPU chip or the connections between the MMU and CPU chips.

4.1.27 TEST Ø33

Test Ø33 checks for the proper operation of the MMU's Read-Only-Access mode. The access mode bits in one of the Page Descriptor Registers (PDR's) are set to the 'Read-Only' state. Then the test attempts a write to the page, expecting a MMU trap to occur. After the trap, the MMU's Status Register Ø (SRØ) should have the 'Read-Only' error bit set, and should indicate the proper page and mode (user/kernel). Status Register 2 (SR2) should have locked-up the address of the instruction that caused the trap.

If the write access to a read-only page fails to cause a trap, an ERROR Ø17 is declared. If SR2 failed to lock-up with the proper address, an ERROR Ø3Ø is displayed. If SRØ did not contain the proper error bit, mode, or page, an ERROR Ø27 is declared. In all cases, the PDP-11 MMU chip is the most likely cause of the failure. Other possible, but less likely, causes of the error include the PDP-11 CPU chip and the connections between the MMU and CPU chips.

4.1.28 TEST Ø34

Test Ø34 tests the proper operation of a MMU Page-Length-Fault (PLF) when the page expansion is upwards ('ED' bit in PDR = Ø). First the test makes a reference within the legal limits of the page being tested, to insure no trap occurs. Then a reference is made outside the legal page limits and a MMU trap is expected.

If the access within legal limits results in a MMU trap, an ERROR ØØØ is reported. If the access outside the legal page limits fails to trap, an ERROR Ø17 is declared. If Status Register Ø (SRØ) does not have the proper error bit set after the trap, an ERROR Ø27 is displayed. In all cases, the PDP-11 MMU chip is the most probable cause of the failure. Other possible causes are the PDP-11 CPU chip, and the interconnect between the MMU and CPU chips.

4.1.29 TEST Ø35

Test Ø35 tests the proper operation of a MMU Page-Length-Fault (PLF) when the page expansion direction is downwards ('ED' bit in PDR = 1). First the test makes a reference within the legal limits of the page being tested to insure no trap occurs. Then a reference is made outside the legal page limits and a MMU trap is expected.

If the access within the legal limits results in a MMU trap, an ERROR ØØØ is declared. If the access outside the legal page limits fails to trap, an ERROR Ø17 is reported. If Status Register Ø (SRØ) doesn't have the proper error bit set after the trap, and ERROR Ø27 is displayed. In all cases, the PDP-11 MMU chip is the most probable cause of the error. Other possible, but less likely, causes of the failure include the PDP-11 CPU chip and the connections between the MMU and CPU chips.

4.1.3Ø TEST Ø36

Test Ø36 checks that a RTI (Return From Interrupt) instruction executed while the MMU is in User mode does not change the MODE or PRIORITY bits in the Processor Status Word (PSW). A RTI instruction is executed while in User mode, then the mode and priority bits in the PSW are checked to be unchanged.

If the RTI instruction fails, an ERROR Ø2Ø is declared. This would most likely be caused by a failure within the PDP-11 CPU chip. If the RTI works, but changes the PSW mode or priority bits, the PDP-11 MMU chip is the most likely cause of the problem.

4.1.31 TEST 037

Test 037 checks for the proper operation of a simultaneous Non-Existent-Memory (NXM) trap and MMU abort. The MMU abort should take precedence over the NXM trap, and the MMU's Status Register 0 (SR0) should indicate the MMU error. The test references a non-existent location outside the legal limits of the MMU page being tested. A MMU abort should occur, and SR0 should indicate a Page Length Fault (PLF).

If the NXM trap takes precedence over the MMU abort, an ERROR 000 will be displayed. If SR0 does not indicate a Page Length Fault with the proper page and mode (kernel/user), an ERROR 027 is reported. In both cases, the PDP-11 MMU chip is the most likely cause of the error. Other possible causes of the error are the PDP-11 CPU chip or the interconnect between the MMU and CPU chips.

4.1.32 TEST 040

Test 040 tests the operation of a Non-Existent-Memory trap that causes a MMU Page Length Fault (PLF) while the MMU is in User mode. Conditions are set to cause a PLF while the NXM PC and PSW are being pushed onto the User stack. The PLF should cause the NXM PC and PSW to be pushed onto the Kernel stack instead of the User stack.

If the Non-Existent-Memory (NXM) trap fails to occur, an ERROR 000 is declared. If the MMU's Status Register 2 (SR2) fails to lock-up at the address of the instruction that caused the trap, an ERROR 030 is displayed. If the NXM trap PC was not pushed onto the Kernel stack, an ERROR 064 is reported. If the NXM trap PSW was not pushed on the Kernel stack, an ERROR 025 is declared. If the MMU's Status Register 0 does not contain the proper error bit, page, and mode (user), an ERROR 027 is reported. If the test detects that operations on the User stack affect the Kernel stack, or vice versa, an ERROR 026 is displayed. All of these errors are most likely the fault of the PDP-11 MMU chip, although the PDP-11 CPU chip and the connections between the MMU and CPU are possible causes of these problems.

4.1.33 TEST 041

Test 041 tests the operation of the MTPI and MFPI instructions when the MMU's current mode is Kernel mode, and the previous MMU mode is User mode. Under these conditions, the contents of the User PAR's and PDR's should be used to perform the mapping done by the MTPI and MFPI instructions. In order to tell which mapping (user or kernel) is being used by these instructions, User mapping is set up differently than Kernel mapping in the PAR/PDR pairs used for the test.

If the MFPI instruction uses Kernel mapping instead of User mapping, an ERROR 037 is displayed. If the MTPI instruction uses the wrong mapping information, an ERROR 040 is declared. The EXPected data displayed with the error report indicates the pattern that was expected if the correct mapping is used, the ACTual data displayed indicates the data pattern that was actually found. In either case the most likely cause of the error is the PDP-11 MMU chip. Other possible causes include the PDP-11 CPU chip, and the connections between the MMU and CPU chips.

4.1.34 TEST 042

Test 042 tests the operation of the MTPI and MFPI instructions when the MMU's current mode is User and the previous mode is Kernel. Under these conditions, the contents of the Kernel PAR's and PDR's should be used to perform the mapping for the MTPI and MFPI instructions. In order to tell which mode is being used for mapping (user/kernel), the PAR/PDR pairs used for the test are set differently for User and Kernel modes.

If the MFPI instruction uses User mapping instead of Kernel mapping, an ERROR 037 is reported. If the MTPI instruction uses the wrong mapping, an ERROR 040 is declared. The EXPected data displayed with the error report shows the pattern that was expected if the correct mapping was used. The ACTual data indicates the data pattern that was actually found. In either case, the PDP-11 MMU chip is the most likely cause of the error. Other possible causes include the PDP-11 CPU chip and the interconnect between the MMU and CPU chips.

4.1.35 TEST Ø43

Test Ø43 tests the operation of the MTPI and MFPI instructions when the MMU's current mode is Kernel and the previous mode is Kernel. Under these conditions, the contents of the Kernel PAR's and PDR's should be used to perform the mapping for the MTPI and MFPI instructions. In order to tell which mode is being used for mapping (user/kernel), the PAR/PDR pairs used for the test are set differently for User and Kernel modes.

If the MFPI instruction uses User mapping instead of Kernel mapping, an ERROR Ø37 is reported. If the MTPI instruction uses the wrong mapping, an ERROR Ø4Ø is declared. The EXPected data displayed with the error report shows the pattern that was expected if the correct mapping was used. The ACTual data indicates the data pattern that was actually found. In either case, the PDP-11 MMU chip is the most likely cause of the error. Other possible causes of the failure include the PDP-11 CPU chip and the interconnect between the MMU and CPU chips.

4.1.36 TEST Ø44

Test Ø44 tests the operation of the MTPI and MFPI instructions when the MMU's current mode is User and the previous mode is User. Under these conditions, the contents of the User PAR's and PDR's should be used to perform the mapping for the MTPI and MFPI instructions. In order to tell which mode is being used for mapping (user/kernel), the PAR/PDR pairs used for the test are set differently for User and Kernel modes.

If the MFPI instruction uses Kernel mapping instead of User mapping, an ERROR Ø37 is reported. If the MTPI instruction uses the wrong mapping, an ERROR Ø4Ø is declared. The EXPected data displayed with the error report shows the pattern that was expected if the correct mapping was used. The ACTual data indicates the data pattern that was actually found. In either case, the PDP-11 MMU chip is the most likely cause of the error. Other possible causes include the PDP-11 CPU chip and the interconnect between the MMU and CPU chips.

4.1.37 TEST 045

Test 045 tests that a HALT instruction does not halt the PDP-11 when the Memory Management Unit is in User mode. When in User mode, a HALT instruction should result in a trap thru vector 000010.

If the PDP-11 halts instead of trapping thru 000010, no error report will be produced. If the procedure listed in section 3.1.2 is followed, the user will note that the contents of TSTNUM (test number) are equal to 000045, indicating that the HALT occurred in test 045. If the HALT instruction does not halt the PDP-11, or cause a trap thru 000010, an ERROR 017 is declared. If the HALT instruction traps, but uses an incorrect vector, an ERROR 000 is reported. The PDP-11 MMU chip is the most likely cause of any of these errors, since most of the logic is within the MMU chip. The next most likely cause of an error is the PDP-11 CPU chip, and the interconnect between the MMU and CPU chips.

4.1.38 TEST 046

Test 046 is a test of the P.ioc's Switch/Display Register (address 17770042). The test checks that the lower 6 bits of the register can be set and cleared, and also checks that the upper 3 bits of the register read as binary 'zeroes'. Note that the other 6 bits of the register can only be tested manually, since someone must depress the front panel switches to test these bits.

If this test fails, an ERROR 041 will be reported. The EXPected and ACTual data displayed with the error report will indicate the bit or bits that are failing. If the lower 6 bits are failing, the Switch/Display register is most likely the cause of the error. If the upper 3 bits do not read as zeroes, check out the special logic that is supposed to supply zeroes to these bits when the register is read.

4.1.39 TEST 047

Test 047 checks out the parity error detection circuitry. The test uses the 'Invert High Byte Parity' bit in the P.ioc's Control and Status Register (address 17770040) to write bad parity to the upper byte of a test location. Then the location is read and a parity trap (vector 000114) is expected to occur. After the trap, the Error Address Register (addresses 17770024 and 26) is checked to insure that the address of the bad parity location was latched by the trap. The test also insures that the parity-invert bit was automatically cleared when a location was written with bad parity. As a final step in the test, the test location is rewritten with good parity.

If the parity-invert bit fails to set, or fails to clear when a write is performed, an ERROR 042 is declared. The EXPected and ACTual data displayed with the error report indicate whether the bit failed to set or clear. The most likely cause of this problem is the logic that implements the high-byte parity invert bit in the P.ioc's CSR.

If no parity trap occurs, an ERROR 017 is declared. This error would most likely be caused by a failure in the parity error detection logic.

If the wrong address is latched in the Error Address Register, an ERROR 024 is declared. The EXPected and ACTual data displayed with the error report indicate which bit or bits are failing to be latched. This error is most likely caused by a failure in the P.ioc logic that implements the Error Address Register.

NOTE: In all the above cases, the test location is rewritten with good parity before the error is reported.

4.1.40 TEST 050

Test 050 checks out the P.ioc's parity error detection logic. The test uses the 'Invert Low Byte Parity' bit in the P.ioc's Control and Status Register (address 17770040) to write bad parity to the lower byte of a test location. Then the test location in read, and a parity trap (vector 000114) is expected. After the trap, the Error Address Register (addresses 17770024 and 26) is checked to insure that the address causing the trap was latched when the trap occurred. The test also checks that the parity invert bit was automatically cleared when a location was written with bad parity. As a final step in the test, the test location is rewritten with good parity.
(Continued on next page)

4.1.40 (Continued)

If the parity invert bit fails to set, or fails to clear when a write is performed, an ERROR 042 is reported. The EXPected and ACTual data displayed with the error report indicate whether the bit failed to set or clear. The most probable cause of this error is the logic that implements the low-byte parity invert bit in the P.ioc's CSR.

If no parity trap occurs, an ERROR 017 is declared. This error would most likely be caused by the P.ioc's parity error detection logic.

If the wrong address is latched in the Error Address Register, an ERROR 024 is displayed. The EXPected and ACTual data displayed with the error report indicate the bit or bits that failed to latch. This error would most likely be caused by the P.ioc logic that implements the Error Address Register.

NOTE: In all the above cases, the test location is rewritten with good parity before the error is reported.

4.1.41 TEST 051

Test 051 tests the P.ioc's Window Index Register (address 17770020). The test loads the window index with all octal values between 000 and 177. This checks out the path from the P.ioc's 'OUT' bus, to the window index register, thru the output latches, and back to the P.ioc's 'IN' bus.

If the test fails, an ERROR 044 is reported. The EXPected and ACTual data displayed with the error report indicate which bit or bits of the index register are failing. The error is likely to be in the P.ioc logic that implements the window index register.

4.1.42 TEST Ø52

Test Ø52 checks out the P.ioc's Window Address Registers. The first part of the test checks bits 2 thru 8 of the Window Address Register. Window Ø of Window Set Ø is used. All 128 possible combinations of 'ones' and 'zeroes' are written to the window, then read back and compared. This checks for a complete path from the P.ioc's 'OUT' bus, thru the Window Register, and back to the P.ioc's 'IN' bus. The second part of the test checks bits 9,1Ø, and 11 of the Window Address Register by addressing windows 1,2,3,4,5,6, and 7 of Window Set Ø. (Window Ø of Window Set Ø was tested in the first part of the test).

If bits 2 thru 8 of the windows fail, an ERROR Ø45 is reported. If bits 9 thru 11 fail, an ERROR Ø46 is declared. The EXPected and ACTual data displayed with the error report indicate the individual bit or bits that are failing. The problem is most likely in the P.ioc's Control Memory Window logic.

4.1.43 TEST Ø53

Test Ø53 tests the RAM memories that implement the P.ioc's Control Memory Windows. The test uses a modified form of the Moving Inversions Memory Test Algorithm. The windows are initially all loaded with binary 'zeroes'. Beginning with Window Ø of Window Set Ø, each window of each set is checked to insure it still contains Ø, the window is written with the pattern ØØØØØ1, and the window is read to insure it contains ØØØØØ1. After Window 7 of Window Set 7 is reached, the pattern is updated to ØØØØØ3 and the same cycle is repeated (I.E. read window and insure it contains ØØØØØ1 from last time thru, then write ØØØØØ3, then read and check for ØØØØØ3). On each pass thru the test, the pattern is updated by adding a single additional 1 bit, until the pattern 177777 has been used.

If the test finds that the previous pattern is no longer in the window, an ERROR Ø47 is reported. If the ACTual data in the error report contains a single additional 'one' bit when compared to the EXPected data, and the additional 'one' occurs immediately to the left of the leftmost 'one' in the EXPected data, a dual-addressing problem may exist in one of the RAM's. Otherwise one of the RAM's has a data problem.

If the test finds that the new pattern was not written correctly, an ERROR Ø5Ø is reported. The EXPected and ACTual data displayed with the error report indicate the bit or bits that are failing. The most likely cause of the error is a data problem in one of the window memory RAM's.

4.1.44 TEST 054

Test 054 tests the RAM memories that implement the P.ioc's Control Memory Windows. The test uses a modified form of the Moving Inversions Memory Test Algorithm. The windows all contain 177777 at the beginning of the test (as a result of test 053). Beginning with Window 7 of Window Set 7, (and progressing downwards to Window 0 of Window Set 0) read each window to insure it contains the pattern 177777, then write the window with the pattern 077777, and read the window back to check that it contains 077777. After Window 0 of Window Set 0 is reached, the pattern is updated by adding a single additional 'zero' bit (pattern becomes 037777 after 077777), and the same cycle is repeated. (I.E. read each window and insure the last pattern is still there, write the new pattern, check that the new pattern got written correctly). When the pattern 000000 has been used in each window, the test terminates.

If the test finds that the previous pattern is no longer in a window, an ERROR 047 is declared. If the EXPected data in the error report contains a single additional 'one' bit when compared to the ACTual data, and the additional 'one' occurs immediately to the left of the leftmost 'one' in the ACTual data, a dual-addressing problem may exist in one of the RAM's. Otherwise one of the window memory RAM's has a data problem.

If the test finds that the new pattern was not written correctly, an ERROR 050 is reported. The EXPected and ACTual data displayed with the error report indicate the bit or bits that are failing. The most likely cause of the error is a data problem in one of the window memory RAM's.

4.1.45 TEST 055

Test 055 tests the Control Memory Data Tranceivers on the P.ioc, without actually referencing the Control Memory. This is achieved by using the P.ioc's Non-Memory-Access (NMA) feature to perform a write to the control memory. When the NMA bit in the P.ioc's CSR is set, the write to the control memory does not actually enable a memory cycle, but the data is latched in the control memory tranceivers on the P.ioc. After the NMA write, a NMA read is performed, which simply reads the contents of the tranceivers without activating the Control Memory. This process allows the data path to and from the tranceivers to be tested, although it can not test whether the path from the tranceivers out to the Control Memory Bus is working (an actual read and write is necessary to test that). The test writes and reads two different patterns to check for opens and shorts in the path to and from the tranceivers.

If the test fails, an ERROR 051 is reported. The EXPected and ACTual data displayed with the error report indicate which bit or bits of the tranceivers are failing. Look for the problem in the control memory tranceivers, or in the data path to and from the tranceivers.

4.1.46 TEST 056

Test 056 tests the P.ioc's 'K-Status Register' (address 17770046). Each 'P' and 'K' in the HSC50 has a status register which can be read by the P.ioc. To read the status of a particular 'K' or 'P', the 'K-Init Register' (address 17770044) is loaded with the correct 'Slot Number', which enables the status from that slot to be read from the P.ioc's 'K-Status Register'. The test consists of writing and reading the P.ioc's own status to test the part of the status logic that resides on the P.ioc module. The K-Init register is cleared to enable the status from slot 0, which is the P.ioc's slot. Then the P.ioc's status is set to 000125, and the K-Status register is read. Reading the K-Status register should return the value 000325 (P.ioc's status plus parity bit for odd parity). Next the P.ioc's status is set to 000052, and the K-Status register is read again. This time the K-Status register should read as 000052 (parity bit is clear).
(Continued on next page)

4.1.46 (Continued)

If the test fails, an ERROR Ø52 will be reported. The expected and actual data displayed with the error report indicate which bit or bits are failing. If the lower 8 bits of the ACTual data are correct, but the upper 8 bits are not zeroes, check out the special circuit that is supposed to supply zeroes to these upper bits. If the ACTual and EXPected data only differ by one or two bits, the most likely cause is a faulty bit position in the register or its output latch. If the EXPected and ACTual data differ by many bits, check that the 'STAT ENA Ø' signal is being generated, and that the signals 'LD KST REG' and 'RD KST REG' are working.

4.1.47 TEST Ø57

Test Ø57 is a test of the Data Memory Tranceivers (data and address) on the P.ioc module, without actually referencing the Data Memory. This is accomplished by using the P.ioc's Non-Memory-Access (NMA) feature to perform a write to the Data Memory. When the NMA bit in the P.ioc's Control and Status Register (CSR) is set, the write to the Data Memory does not actually enable a memory cycle, but the data used is latched in the Data Memory data tranceivers on the P.ioc module. After the NMA write, a NMA read is performed to read the data from the tranceivers without activating the Data Memory. This process allows the data path to and from the tranceivers to be tested, although it can not test that the path from the tranceivers to the Data Bus is working (an actual read or write is necessary to test that). Test Ø56 also tests the Data Memory Address tranceivers via the Data Memory Address Register (addresses 1777005Ø and 52). The Data Memory Address register latches the last address sent to the Data Memory. After the NMA write is performed, the Data Memory Address register is read to insure that the path to the Data Memory Data tranceivers is working.

If the test detects a failure in the Data Memory Address lines, an ERROR Ø54 is reported. The EXPected and ACTual data supplied with the error report indicate the address bit or bits that are failing. The problem is likely to be in the Data Memory Address tranceivers.

If the test detects a failure in the Data Memory data path, an ERROR Ø55 is reported. The EXPected and ACTual data displayed with the error report indicate the data bit or bits that are failing. The problem is likely to be in the Data Memory Data tranceivers.

4.1.48 TEST Ø6Ø

Test Ø6Ø is a test of the Data Memory Tranceivers (data and address) on the P.ioc module, without actually referencing the Data Memory. This is accomplished by using the P.ioc's Non-Memory-Access (NMA) feature to perform a write to the Data Memory. When the NMA bit in the P.ioc's Control and Status Register (CSR) is set, the write to the Data Memory does not actually enable a memory cycle, but the data used is latched in the Data Memory data tranceivers on the P.ioc module. After the NMA write, a NMA read is performed to read the data from the tranceivers without activating the Data Memory. This process allows the data path to and from the tranceivers to be tested, although it can not test that the path from the tranceivers to the Data Bus is working (an actual read or write is necessary to test that). Test Ø56 also tests the Data Memory Address tranceivers via the Data Memory Address Register (addresses 1777ØØ5Ø and 52). The Data Memory Address register latches the last address sent to the Data Memory. After the NMA write is performed, the Data Memory Address register is read to insure that the path to the Data Memory Data tranceivers is working.

If the test detects a failure in the Data Memory Address lines, an ERROR Ø54 is reported. The EXPected and ACtual data supplied with the error report indicate the address bit or bits that are failing. The problem is likely to be in the Data Memory Address tranceivers.

If the test detects a failure in the Data Memory data path, an ERROR Ø55 is reported. The EXPected and ACTual data displayed with the error report indicate the data bit or bits that are failing. The problem is likely to be in the Data Memory Data tranceivers.

4.1.49 TEST 061

Test 061 tests the P.ioc's 'Control Memory Interrupt' feature. The test checks that a 'priority 7' interrupt is honored before a priority 4,5, or 6 interrupt is honored. A priority 7 interrupt is generated by setting the Control Memory Interrupt enable in the P.ioc's Control and Status Register (address 17770040), then writing the pattern 000010 to the Control Memory. When the Control Memory Interrupt enable is set, a write to the Control Memory does perform a normal write cycle. Instead the write data is used as an interrupt mask to the P.ioc's Control Memory interrupt logic. The pattern 000010 will generate a priority 7 interrupt to the P.ioc, resulting in an interrupt thru the vector 000134 in Program Memory.

If the Control Memory Interrupt enable bit in the P.ioc's CSR fails to set, an ERROR 056 is reported. The problem is most likely in the CSR bit, or in the path to and from the CSR for that bit.

If the priority 7 interrupt does not occur, an ERROR 017 is declared. The problem could be in the P.ioc logic that receives the interrupt mask from the Control Memory Data lines, or in the registers that latch the mask. The problem may also be in the circuitry that actually generates the priority 7 interrupt.

If the interrupt occurs, but uses the wrong interrupt vector, an ERROR 000 is reported. The problem may be in the P.ioc ROM's that generate the interrupt vector addresses.

4.1.50 TEST 062

Test 062 tests the P.ioc's 'Control Memory Interrupt' feature. The test checks that a 'priority 6' interrupt is honored before a priority 4 or 5 interrupt is honored. A priority 6 interrupt is generated by setting the Control Memory Interrupt enable in the P.ioc's Control and Status Register (address 17770040), then writing the pattern 000004 to the Control Memory. When the Control Memory Interrupt enable is set, a write to the Control Memory does perform a normal write cycle. Instead the write data is used as an interrupt mask to the P.ioc's Control Memory interrupt logic. The pattern 000004 will generate a priority 6 interrupt to the P.ioc, resulting in an interrupt thru the vector 000130 in Program Memory.

If the priority 6 interrupt does not occur, an ERROR 017 is declared. The problem could be in the P.ioc logic that receives the interrupt mask from the Control Memory Data lines, or in the registers that latch the mask. The problem may also be in the circuitry that actually generates the priority 6 interrupt.

If the interrupt occurs, but uses the wrong interrupt vector, an ERROR 000 is reported. The problem may be in the P.ioc ROM's that generate the interrupt vector addresses.

4.1.51 TEST 063

Test 063 tests the P.ioc's 'Control Memory Interrupt' feature. The test checks that a 'priority 5' interrupt is honored before a priority 4 interrupt is honored. A priority 5 interrupt is generated by setting the Control Memory Interrupt enable in the P.ioc's Control and Status Register (address 17770040), then writing the pattern 000002 to the Control Memory. When the Control Memory Interrupt enable is set, a write to the Control Memory does perform a normal write cycle. Instead the write data is used as an interrupt mask to the P.ioc's Control Memory interrupt logic. The pattern 000002 will generate a priority 5 interrupt to the P.ioc, resulting in an interrupt thru the vector 000124 in Program Memory.

If the priority 5 interrupt does not occur, an ERROR 017 is declared. The problem could be in the P.ioc logic that receives the interrupt mask from the Control Memory Data lines, or in the registers that latch the mask. The problem may also be in the circuitry that actually generates the priority 5 interrupt.

If the interrupt occurs, but uses the wrong interrupt vector, an ERROR 000 is reported. The problem may be in the P.ioc ROM's that generate the interrupt vector addresses.

4.1.52 TEST Ø64

Test Ø64 tests the P.ioc's 'Control Memory Interrupt' feature. The test checks that a 'priority 4' control memory interrupt is honored. A priority 4 interrupt is generated by setting the Control Memory Interrupt enable in the P.ioc's Control and Status Register (address 1777ØØ4Ø), then writing the pattern ØØØØØ1 to the Control Memory. When the Control Memory Interrupt enable is set, a write to the Control Memory does perform a normal write cycle. Instead the write data is used as an interrupt mask to the P.ioc's Control Memory interrupt logic. The pattern ØØØØØ1 will generate a priority 4 interrupt to the P.ioc, resulting in an interrupt thru the vector ØØØ12Ø in Program Memory.

If the Control Memory Interrupt enable bit in the P.ioc's CSR fails to clear, an ERROR Ø56 is reported. The problem is most likely in the CSR bit, or in the path to and from the CSR for that bit.

If the priority 4 interrupt does not occur, an ERROR Ø17 is declared. The problem could be in the P.ioc logic that receives the interrupt mask from the Control Memory Data lines, or in the registers that latch the mask. The problem may also be in the circuitry that actually generates the priority 4 interrupt.

If the interrupt occurs, but uses the wrong interrupt vector, an ERROR ØØØ is reported. The problem may be in the P.ioc ROM's that generate the interrupt vector addresses.

4.1.53 TEST Ø65

Test Ø65 checks out the operation of a 'Control-Memory-Lock-Cycle'. When the Control Memory Lock Cycle bit is set in the P.ioc's Control and Status Register (CSR), the next read from the Control Memory results in a lock cycle. A lock cycle returns the contents of the selected Control Memory location, and replaces the contents of the location with the pattern 1ØØØØØ (the 'lock value'). The test first writes a Control Memory location with a particular pattern, then generates a lock cycle to that location. The test then checks that the original pattern was returned by the lock cycle. Next the location is read to insure that the lock value (1ØØØØØ) was stored by the lock cycle. Then another lock cycle is performed, and this time the returned data is checked to be 1ØØØØØ. Then the location is written with the pattern ØØØØØØ, and the location is read twice to insure that the lock cycle circuitry is not stuck in the 'ON' state (If the lock cycle is stuck 'ON', the second read will return the lock value instead of ØØØØØØ).

If the Lock-Cycle bit in the P.ioc CSR fails to set, or fails to clear after the lock cycle is generated, an ERROR Ø57 is reported. The EXPected and ACTual data displayed with the error report indicate whether the bit failed to set or clear. The failure is probably in the CSR bit that controls the lock function, or in the path to and from the CSR for that bit.

If any of the lock cycles do not return the expected data, an ERROR Ø6Ø is displayed. The EXPected and ACTual data provided in the error report indicate the pattern that should have been read, and the pattern that was actually returned. Check the decoder that translates the 'CCYCLE Ø,1,and 2' lines, and the circuitry that generates the lock cycle signals (CDATA LO PAR L, CDATA 15 L, CWRTL L, and CWRTH L).

A failure in this test, (especially a parity or NXM trap), may be caused by a bad control memory. After the error message is displayed, follow the directions in section 3.1.3 to load the Offline Loader and the Memory test. If the control memory passes the Offline Memory test, the P.ioc's Lock-Cycle circuitry is the probable cause of the failure in Test 65. If the control memory fails the Offline Memory test, the control memory is the probable cause of the failure in test 65.

4.1.54 TEST Ø66

Test Ø66 is not actually a test, but the test numbering scheme is useful for error reporting purposes. This part of the Offline P.ioc Test loads the HSC5Ø OFFLINE DIAGNOSTIC LOADER (ODL) from the TU58 to Program Memory, and sets up the necessary memory mapping for the loader. Then control is transferred to the loader.

If the read from the TU58 fails, an ERROR ØØ3 is declared. The problem is likely to be a bad spot on the TU58 media, or an intermittent problem with the serial interface between the P.ioc and the TU58. The problem is unlikely to be a 'HARD' (persistent) failure, since the same TU58 drive and media were used to load the Offline P.ioc Test previously. Refer to the section of this document that discusses ERROR ØØ3 for further information.

HSC50 Off-Line Diagnostic Loader

USER DOCUMENTATION

PRODUCT CODE:	XX-XXXXX-XX
PRODUCT NAME:	HSC5Ø Offline Diagnostic Loader
PRODUCT DATE:	12-APR-1983
MAINTAINER:	Diagnostic Engineering - Colorado (CX)
AUTHOR:	Mike Hare

TABLE OF CONTENTS

SECTION 1 - GENERAL INFORMATION

1.1 PROGRAM ABSTRACT

The Offline Diagnostic Loader provides a software environment for the HSC5Ø Offline Diagnostics. The loader supports a command language that permits an offline diagnostic to be loaded (from the HSC5Ø's TU58) into Program memory and executed. The loader's command language also permits displaying and modifying the contents of any location in the HSC5Ø's Program, Data, or Control memories.

The software environment provided for Offline Diagnostics includes a TU58 driver and a terminal driver. These drivers supply a standard software interface between diagnostics and these devices, and make it unnecessary for each diagnostic to contain routines to interface to the tape and terminal. The loader also maintains a timer that keeps track of the relative time since the loader was last booted. This allows diagnostic error messages to be time-stamped.

1.2 SYSTEM REQUIREMENTS

Hardware Required:

a) P.ioc (processor) module with HSC5Ø Boot ROMS.
b) At least one M.std (memory) module.
c) TU58 controller with at least one working drive.
d) Terminal connected to P.ioc's console interface.

Software Required:

a) HSC5Ø Offline Diagnostic Tape (TU58 Media)

1.3 RELATED DOCUMENTS

a) HSC5Ø Offline P.ioc Test User Document
b) HSC5Ø Offline K Test Selector User Document 9.break
c) HSC5Ø Offline K/P Memory Test User Document
d) HSC5Ø Offline Memory Test User Document
e) HSC5Ø Offline Bus Interaction Test User Document
f) HSC5Ø Offline System Sizer User Document
g) HSC5Ø Offline OCP Test User Document
h) HSC5Ø K Microdiagnostic User Document
i) HSC5Ø ROM Bootstrap User Document

1.4 HARDWARE ASSUMED TO BE WORKING

In the process of loading the Offline Diagnostic Loader, several diagnostics are run. The ROM Bootstrap tests the basic PDP-11 instruction set, tests a partition in Program Memory, and tests the TU58 used for the boot. Then the bootstrap loads the Offline P.ioc Test, which completes the PDP-11 tests and tests the remainder of the P.ioc module. After these tests, the Offline Diagnostic Loader is loaded from the TU58 to memory, and control is passed to the Loader. Due to the sequence of tests that precede the loader, the loader assumes that the P.ioc module and the TU58 are tested and working.

SECTION 2 - OPERATING INSTRUCTIONS

2.1 START-UP PROCEDURE

Follow the steps below to start the Offline Loader:

1) Insert the HSC50 OFFLINE TESTS cassette into the TU58's Unit 0 drive. (left hand drive)

2) Power-on the HSC50, or depress and release the INIT button on the HSC50 front panel.

3) The TU58 drive-motion LED should light within 10 seconds, indicating that the Bootstrap is loading the OFFLINE DIAGNOSTIC LOADER to Program Memory.

4) After about 60 seconds of loading, the OFFLINE DIAGNOSTIC LOADER will indicate that it has loaded properly by displaying the following:

```
      HSC50 OFL Diagnostic Loader, Version Vnn-nn
      Radix=Octal,Data Length=Word,Reloc=00000000
  ODL>
```

5) The Offline Loader is now ready to accept commands. Refer to section 2.2 for information on the Loader's command language.

2.2 COMMAND LANGUAGE

This section describes the commands recognized by the Offline Loader. Appendix A of this document is a copy of the Offline Loader's Help file.

2.2.1 HELP COMMAND

The HELP command provides an abbreviated list of all commands that the Loader recognizes. In response to the HELP command, the Loader reads the file OFLLDR.HLP from the TU58 and displays the contents of this file on the HSC50's operator terminal. See appendix A of this document for a listing of the Loader Help file.

2.2.2 SIZE COMMAND

The Offline System Sizer is invoked by the SIZE command. The Sizer will determine the sizes of the HSC50's Program, Control, and Data memories, and the type of K in each HSC50 requestor position. (The requestor position refers to the priority of a particular K on the Data and Control memory Buses. It does not match up with the numbering of module slots.) Refer to the Offline System Sizer User Document for further information.

2.2.3 INVOKING OFFLINE DIAGNOSTICS

The Loader's TEST command is used to invoke the various offline diagnostics available on the HSC50. The following sections list the particular form of the TEST command used to invoke each diagnostic. In general, the format of the TEST command is designed so that an operator specifies the system component that should be tested. E.G. The TEST MEMORY command invokes the Offline Memory Test.

2.2.3.1 MEMORY TEST

The Offline Memory test is invoked by the TEST MEMORY command. The Offline Memory test uses the P.ioc to test the Program, Control, or Data memories. Refer to the Offline Memory Test User Document for further information. NOTE: For a faster test of the Control or Data memories, use the TEST MEMORY BY K command as described in the following section.

2.2.3.2 K/P MEMORY TEST

The Offline K/P Memory test is invoked by the TEST MEMORY BY K command. The K/P Memory test uses one of the HSC5Ø K's to test either the Data or Control memory. This test runs faster than the Offline Memory Test, since a K is roughly 7 times faster than the P.ioc. The Program memory can not be tested using the K/P memory test, since the K's do not have an interface to the Program memory bus. Refer to the Offline K/P Memory Test User Document for further information.

2.2.3.3 K TEST SELECTOR

The Offline K Test Selector is invoked by the TEST K command. The K Test Selector allows an operator to run specific K microdiagnostics. Refer to the Offline K Test Selector User Document for further information.

2.2.3.4 BUS INTERACTION TEST

The Offline Bus Interaction Test is invoked by the TEST BUS command. The Bus Interaction test generates contention on the HSC5Ø's Data and Control memory buses, by having two or more K's simultaneously test different sections of the Control and Data memories. NOTE: Two or more working K's are required to run this test. Refer to the Offline Bus Interaction Test User Document for further information.

2.2.3.5 OCP TEST

The Offline OCP Test is invoked by the TEST OCP command. The OCP (Operator Control Panel) test allows the HSC5Ø's lights and switches to be tested. The test requires manual intervention by an operator.

2.2.3.6 MEMORY REFRESH TEST

The Offline Memory Refresh Test is invoked by the TEST REFRESH command. The Memory Refresh test allows the refresh feature of the memories to be tested.

2.2.4 LOAD AND START COMMANDS

2.2.4.1 LOAD COMMAND

The LOAD command allows a program to be loaded into HSC5Ø Program memory without being started. The format of the command is 'LOAD <filename>', where '<filename>' is the name of any file on the HSC5Ø's Offline cassette. The Loader will find the specified file and load it into the Program memory. This command is useful when the operator wishes to patch a program image before starting execution. After a patch has been made, the program can be initiated via the START command as described in the following section.

2.2.4.2 START COMMAND

The START command directs the Loader to initiate the program that is currently loaded in Program memory. The START command can be used in conjunction with the LOAD command (see preceeding section), or it may be used to reinitiate the last offline diagnostic. This saves the time required to reload the program from the TU58. E.G. Operator has previously typed SIZE to initiate the Offline System Sizer program. After the Sizer completes, the operator wishes to run the Sizer again. Typing 'START<CR>' will restart the Sizer without having to reload the program again from the TU58, saving many seconds of load time.

2.2.5 EXAMINE AND DEPOSIT COMMANDS

The EXAMINE and DEPOSIT commands can be used to display or modify the contents of any location in the HSC5Ø's Program, Control, and Data memories. Qualifiers (switches) may be used with these commands to display bytes, words, long words, or quad words, and the radix (octal, decimal, hex) of the displayed data can also be controlled by qualifiers. Alternately, the SET DEFAULT command can be used to set the default data length and radix for all Examine and Deposit commands. (see section 2.2.6).

2.2.5.1 EXAMINE COMMAND

The EXAMINE command is used to display the contents of any location in the HSC5Ø's Program, Data, or Control memories. The format of the command is: EXAMINE <address>. The <address> can be a string of digits in the current (default) radix, but certain symbolic addresses are also permitted (see section 2.2.5.3).

EXAMPLE:

```
ODL> E 14Ø17776 <CR>
  (D) 14Ø17776  125252
```

In the example the user entered a command to examine the contents of location 14Ø17776. Notice that the EXAMINE command can be abbreviated to a single 'E'. When the loader displays the contents of location 14Ø17776, the address is preceeded by a '(D)' to indicate that the location is within the HSC5Ø's Data memory. The display indicates that the location contains the value 125252.

2.2.5.2 DEPOSIT COMMAND

The DEPOSIT command is used to modify the contents of any location in the HSC5Ø's Program, Control, or Data memories. The format of the command is: DEPOSIT <address> <data>. The <address> can be a string of digits in the current (default) radix, but certain symbolic addresses are also permitted (see next section).

EXAMPLE:

```
ODL> D 14Ø17776 123456 <CR>
```

In the example above, the user entered a command to store the value 123456 into the contents of address 14Ø17776. The previous contents of this Data memory location will be replaced with the value specified in the Deposit command (123456).

2.2.5.3 SYMBOLIC ADDRESSES

1. '*' - The symbol '*' can be used as an <address> in a Deposit or Examine command. The symbol '*' means that the loader is to use the same address as was used in the last Examine or Deposit command. For example: If the user just examined the contents of address 16Ø12344, and now wishes to deposit the value 1234 into the same address, the user can type 'DEPOSIT * 1234', instead of typing 'DEPOSIT 16Ø12344 1234'.

2. '+' - The symbol '+' can be used as an address in a Deposit or Examine command. The symbol '+' means that the loader is to use the address following the last address used by an Examine or Deposit command. When the loader sees a '+' as an address, the loader will take the last address used by Examine or Deposit and add an offset which depends on the current default data length (see section 2.2.6). If the current default data length is a byte, the loader will add 1 to the last address. If the default was a word, the loader will add 2 to the last address. The offset will be 4 for longword data length, and 8 for quadword. This feature is useful when examining a number of items that are stored in successive locations. For example: If the user is examining a table of words beginning at address 14125234, the user would examine the first location by typing 'EXAMINE 14125234'. The next location could now be examined by typing 'EXAMINE +', instead of typing 'EXAMINE 14125236'.

3. '-' - The symbol '-' can be used as an address in a Deposit or Examine command. The symbol '-' means that the loader is to use the address preceeding the last address used by an examine or deposit command. When the loader sees a '-' as an address, the loader will take the last address used by an Examine or Deposit and subtract an offset which depends on the current default data length (see section 2.2.6). If the current default data length is a byte, the loader will subtract 1 from the last address. If the default was a word, the loader will subtract 2 from the last address. The loader will subtract 4 for longword data length, and 8 for quadword. This feature is useful in the same way as the '+' symbol (see above), but is designed for examining a table starting at the highest address and proceeding down to lower addresses. For example: If the user wants to examine a table of words that ends at address 14Ø12346, the user would examine the last location of the table by typing 'EXAMINE 14Ø12346'. The preceeding location in the table could now be accessed by typing 'EXAMINE -',

instead of having to type 'EXAMINE 14012344'.

4. '@' - The symbol '@' can be used as an address in a Deposit or Examine command. The symbol '@' means that the loader is to use the data from the last Examine or Deposit command as an address. This feature is useful when following linked lists. For example: The user first examines location 123434, which contains a pointer to a linked list. Now the user can type 'EXAMINE @' to examine the location to which the first location pointed.

2.2.5.4 REPEATING EXAMINE AND DEPOSIT COMMANDS

When troubleshooting memory problems, it is sometimes useful to continuously execute an Examine or Deposit command. While the command is being repeated, an oscilloscope can be used to detect errors in the memory data, address, and timing circuits.

The REPEAT command is used to continuously execute an Examine or Deposit command. The user types REPEAT followed by the Examine or Deposit command to be repeated.

EXAMPLE # 1 - Repeating a Deposit command

```
REPEAT DEPOSIT 14017776 125252 <CR>
or   RE D 14017776 125252 <CR>
```

In the example above, the user wishes to continuously deposit the value 125252 into address 14017776. Notice that the format of the Deposit command does not change. The Deposit command is just preceeded by the word REPEAT. Notice also that the REPEAT command can be abbreviated to 'RE'.

EXAMPLE # 2 - Repeating an Examine command

```
REPEAT EXAMINE 14017776 <CR>
or   RE E 14017776 <CR>
```

In the example above, the user wishes to continuously examine the contents of address 14017776. Notice that the format of the Examine command does not change. The Examine command is just preceeded by the word REPEAT. Also notice that the REPEAT command can be abbreviated to 'RE'.

In the example as shown, the contents of location 14017776 would be displayed continuously on the terminal. On hard copy devices this wastes paper, and in any case it slows down the repetition of the command which may make it difficult to sync a scope on the memory signals. The output to the terminal may be stopped by typing a control-o (^O), but the loader's language also provides a special qualifier for the Examine command which suppresses output to the terminal. This qualifier (/INHIBIT) is discussed in section 2.2.5.6.

To stop a repeated command, type a control-c (^C).

2.2.5.5 USING THE RELOCATION REGISTER

The loader provides a relocation register which can be used to reduce the number of address digits typed for an Examine or Deposit command when all addresses are in either the Control or Data memories. The contents of the relocation register are added to the address given with an Examine or Deposit command. The relocation register contains a zero when the loader is initiated, so it normally has no effect on the addresses typed in an Examine or Deposit command.

EXAMPLE # 1 - Relocation to Data memory

The user wishes to examine a large number of locations in the Data memory.

```
ODL> SET RELOCATION:14000000 <CR>
ODL> EXAMINE 0 <CR>
  (D) 14000000 123432

ODL> EXAMINE 1234 <CR>
  (D) 14001234 154323
```

The user loads the relocation register with the address of the first location in Data memory (14000000). Then when the user issues an examine command with an address of 0, the loader adds the relocation register to the address given by the user, resulting in address 14000000 being examined. Likewise, when the user issues an examine command with an address of 1234, the loader will display the contents of location 14001234.

EXAMPLE # 2 - Relocation to Control memory

The user wishes to examine a large number of locations in the Control memory.

```
ODL> SET RELOCATION:16000000 <CR>
ODL> EXAMINE 0 <CR>
  (C) 16000000 125252

ODL> EXAMINE 4320 <CR>
  (C) 16004320 125432
```

The user loads the relocation register with the address of the first location in the Control memory (16000000). Then when the user issues an examine command with an address of 0, the loader adds the relocation register to the address given by the user, resulting in the contents of address 16000000 being displayed. Likewise, when the user issues an examine command with an address of 4320, the loader displays the contents of location 16004320.

2.2.5.6 EXAMINE AND DEPOSIT QUALIFIERS (SWITCHES)

1. /NExt:# - The /NEXT qualifier allows an Examine or Deposit command to work on successive addresses. When used with a valid Examine command, it specifies that after the location specified in the command has been displayed, the loader should also display the next '#' locations following the first. For example, the command 'E 1000/NEXT:5' would result in the display of location 1000, 1002, 1004, 1006, 1010, and 1012 (assuming the default data length is a word). The '#' argument can be any value in the current default radix which can be contained in 15 binary bits or less. I.E. If the default radix is octal, the '#' argument can be any value between 1 and 77777. The /NEXT qualifier works the same way for Deposit commands, except that the data given with the Deposit will be stored in the location specified and the next '#' locations following.

2. /Byte/Word/Long/Quad - These qualifiers can be used to control the data-length of Examined or Deposited data. Normally, the loader uses the default data-length (see section 2.2.6) when data is examined or deposited, however the data-length qualifiers can be used to override the default for a single examine or deposit. I.E. Assume the default data-length is currently a word, and the user wishes to examine a byte quantity at address 16001234. The command 'Examine 16001234/Byte <CR>' would display the proper byte without affecting the default data-length.

3. /Octal/Decimal/Hex - The qualifiers can be used with an Examine command to control the radix of the address and data displayed. They are NOT used to control the radix of the address supplied in the Examine command. The radix of the address and data displayed by an examine command is usually controlled by the current Default Radix (see section 2.2.6), but the /Byte/Word/Long/Quad qualifiers can be used to override the default radix for a single examine command. E.G. Assume the default radix is octal. The command 'Examine 14001234/Hex<CR>' will display the contents of address 14001234(8) in the Hexadecimal radix. The examine display would be as follows: (D) 30029C HHHH. 'HHHH' represents the contents of the location displayed in hex. Notice that the address is also displayed in hex.

4. /INH - The /INHIBIT (abbreviated to /INH) qualifier is used when repeating an Examine command to inhibit the display of examined data. This is useful both for saving paper on hardcopy devices, and for speeding up

the Examine operation for scope-loop purposes. For example: the command 'REPEAT EXAMINE 16012346/INH' will result in the loader continuously reading the contents of location 16012346, without displaying anything on the console.

2.2.6 SETTING AND SHOWING DEFAULTS

The Set Default command is used to change the default radix and/or data length. The default radix controls the radix of parameters supplied with Examine and Deposit commands, and the radix of data displayed by the Examine command. The default data length controls the length (byte,word,long,quad) of data displayed by the Examine command, or data stored by a Deposit command.

The default radix may be set to Octal, Decimal or Hexadecimal. When the Offline Loader is first started, it sets the default radix to Octal. To set the default radix to Hexadecimal, the user would type 'Set Default Hex<CR>'. (See Loader Help file for abbreviation rules for the Set Default command). Once the default radix has been set, it will remain as set until another Set Default command is issued, or the Loader is rebooted.

The default data length may be set to Byte, Word, Longword, or Quadword. When the loader is first started, it sets the default data length to word (16 bits). To set the default data length to Longword (32 bits), the user would type: 'Set Default Long <CR>'. (See Loader help file for abbreviation rules for the Set Default command). Setting the default data length to Longword will cause an Examine command to display longword quantities, and will cause the Deposit command to store longword quantities. (Since the Loader is executing in a PDP-11, longwords are stored and retrieved as two successive 16-bit words). Once the default data length has been set, it will remain as set until changed by another Set Default, or until the Loader is rebooted.

2.2.7 EXECUTING INDIRECT COMMAND FILES

The Loader is capable of executing indirect command files stored on the TU58. These command files consist of valid Offline Loader commands terminated by a Carriage Return (<CR>) and a Line Feed (<LF>). Comments may also be placed in indirect command files by preceeding a comment line with an exclamation mark (!). Comment lines must also be terminated with a <CR> and <LF>. E.G. The Offline Loader Help file is an indirect command file that contains only comments.

Indirect command files cannot be created by the Loader or Cronic. The command files must be created in RT-11 format and stored on the Offline Diagnostic cassette. Any editor that does not insert line numbers in the output files can be used to create command files.

SECTION 3 - ERROR INFORMATION

3.1 UNEXPECTED TRAPS AND INTERRUPTS

When the loader detects an unexpected trap or interrupt, the following message is displayed:

```
    Unexpected trap thru www, VPC=xxx, PSW=yyy
    Error Address = zzz
```

Where:

```
www = The address of the trap or interrupt vector
xxx = Virtual PC of loader at time of trap
yyy = Contents of PSW at time of trap
zzz = Address of location causing NXM or Parity trap
```

The first line of the unexpected trap report is issued for all unexpected traps or interrupts. The second line is only issued if the trap was through vector addresses ØØØØØ4 (NXM trap) or ØØØ114 (parity trap). The address of the vector is a direct clue to the cause of the trap. Refer to the following section for a list of the devices and error conditions associated with each vector. The Virtual PC (VPC) of the instruction that was being executed when the trap occurred is sometimes useful in determining the cause of the trap. The VPC can be referenced in the listing to find the instruction that caused the trap. Remember that the VPC will be the address of the instruction following the instruction that was being executed when the trap occurred.

NXM traps can be caused by Examine or Deposit commands, if a user specifies an address that is not contained in a particular HSC5Ø. E.G. If an HSC5Ø only contains data memory from addresses 14ØØØØØØ thru 14177776, and a user tries to examine or deposit address 142ØØØØØ, the Loader will report an NXM trap. In this example, the NXM trap would NOT represent an error condition.

Parity traps can be caused by an Examine command, if a user examines an address that has not been initialized with good parity. When the HSC5Ø memories are powered-on, the parity bits are in random states. Thus if a user examines a location that has not been written since power-on, the location may generate a parity error. This would not constitute an error condition. However, if a location produces a parity error, and that location has been written since power-on, a memory error is indicated. (Also note that the P.ioc and K's have bits which allow them to write bad parity to allow the parity circuit to be tested. These bits are never supposed to be used except by diagnostics.)

3.1.1 TRAP AND INTERRUPT VECTORS

The following is a list of trap and interrupt vectors for various devices and error conditions recognized by the P.ioc's PDP-11 processor.

Vector	Device or Error Condition
ØØØØØ4	Non-Existent-memory, stack overflow
ØØØØ1Ø	Illegal instruction, F11 control chip error
ØØØØ14	BPT instruction
ØØØØ2Ø	IOT instruction
ØØØØ24	Power Fail Interrupt
ØØØØ3Ø	EMT instruction
ØØØØ34	TRAP instruction
ØØØØ6Ø	Console Terminal - receiver interrupt
ØØØØ64	Console Terminal - Transmitter interrupt
ØØØ1ØØ	Line Clock Interrupt
ØØØ114	Parity Trap
ØØØ12Ø	Control Bus Interrupt - Level 4
ØØØ124	Control Bus Interrupt - Level 5
ØØØ13Ø	Control Bus Interrupt - Level 6
ØØØ134	Control Bus Interrupt - Level 7
ØØØ25Ø	MMU Abort (Trap)
ØØØ3ØØ	TU58 #1, Receiver Interrupt
ØØØ3Ø4	TU58 #1, Transmitter Interrupt
ØØØ31Ø	TU58 #2, Receiver Interrupt
ØØØ314	TU58 #2, Transmitter Interrupt

SECTION 4 - APPENDICES

APPENDIX A - LOADER HELP FILE

```
!  HSC50 OFL Diagnostic Loader Help File - Vnn-nn
!  Capital letters = required input, lower case = optional
!COMMANDS (terminated by CR):
!  'Examine <address>'                ;display data at <address> specified
!  'Deposit <address> <data>'         ;deposit <data> to <address>
!  <address> = digit string in current default radix, or:
!          '*' = use same address as last Ex or De
!          '+' = use address following last address
!          '-' = use address preceeding last address
!          '@' = use <data> from last Ex or De as <address>
!  'HElp'                     ;print this file
!  '@filename'                        ;execute indirect command file
!  'Load filename'            ;load file to diagnostic partition
!  'REpeat <command>'                 ;repeat specified command until ^C
!  'SEt Default <option>'             ;set default radix or data length
!     <option> = Byte,Word,Long,Quad,Hex,Octal,Decimal
!  'SEt Relocation:#'                 ;set relocation register to #
!  NOTE:  Relocation register is 22-bit positive # added to address
!    of all Examine and Deposit commands.
!  'SHow'                     ;display defaults and loader version #
!  'SIze'                     ;Size HSC50 memories and display K status
!  'Start'                    ;start program in diagnostic partition
!  'Test Bus'                         ;load and start the OFL Bus Test
!  'Test MEmory'                      ;load and start the OFL Memory Test
!  'Test MEmory By K'                 ;load and start the OFL K/P Memory Test
!  'Test K'                           ;load and start the OFL K Test Selector
!  'Test Ocp'                         ;load and start the OFL OCP Test
!  'Test Refresh'             ;load and start the OFL Memory Refresh Test
!
! QUALIFIERS (switches) for 'Ex' and 'De'
!  '/Next:#'                          ;repeat Ex or De on next '#' addresses
!  '/Byte,/Word,/Long,/Quad'          ;use specified length vs.  default
!  '/Octal,/Decimal,/Hex'  ;use specified radix for Examine display
!  '/INHibit'                         ;inhibit display of examined data
!  <end of help file>
```

HSC50 Off-Line Operator Control Panel Test (TEST OCP)

USER DOCUMENTATION

PRODUCT CODE:	XX-XXXXX-XX
PRODUCT NAME:	HSC50 Offline OCP Test
PRODUCT DATE:	12-APR-1983
MAINTAINER:	Diagnostic Engineering - Colorado (CX)
AUTHOR:	Mike Hare

TABLE OF CONTENTS

SECTION 1 - GENERAL INFORMATION

1.1 PROGRAM ABSTRACT

The Offline Operator Control Panel (OCP) test provides a means to test the operation of the HSC5Ø's lamps and switches. Testing includes the 5 OCP lamps and switches, the STATE LED, and the SECURE/ENABLE switch and ENABLE LED.

The Offline OCP test is initiated by the Offline Loader's TEST OCP command.

This document includes a troubleshooting section that describes procedures for localizing faults detected by this test.

1.2 SYSTEM REQUIREMENTS

Hardware Required:
a) P.ioc (processor) module with HSC5Ø Boot ROMS.
b) At least one M.std (memory) module.
c) TU58 controller with at least one working drive.
d) Terminal connected to P.ioc's console interface.
e) Operator Control Panel

Software Required:
a) HSC5Ø Offline Diagnostic Tape

1.3 RELATED DOCUMENTS

a) HSC5Ø Offline Diagnostic Loader User Document
b) HSC5Ø Offline P.ioc Test User Document
c) HSC5Ø ROM Bootstrap User Document

1.4 HARDWARE ASSUMED TO BE WORKING

In the process of loading the Offline Diagnostic Loader, several diagnostics are run. The ROM Bootstrap tests the basic F-11 instruction set, tests a partition in Program Memory, and tests the TU58 used for the boot. Then the bootstrap loads the Offline P.ioc Test, which completes the F-11 tests and tests the remainder of the P.ioc module. After these tests, the Offline Diagnostic Loader is loaded from the TU58 to memory, and control is passed to the Loader. When the operator instructs the Loader to 'TEST OCP', the Offline OCP Test is loaded and started. Due to the sequence of tests that precede this test, the P.ioc, Program memory, and TU58 are assumed to be tested and working.

SECTION 2 - OPERATING INSTRUCTIONS

2.1 STA T-UP PROCEDURE

Follow the steps below to start the OCP test:

1) Insert the HSC50 OFFLINE TESTS cassette into the TU58's Unit 0 drive. (left hand drive)

2) Power-on the HSC50, then depress and release the INIT button on the HSC50 front panel.

3) The TU58 drive-motion LED should light within 10 seconds, indicating that the Bootstrap is loading the OFFLINE DIAGNOSTIC LOADER to Program Memory.

4) After about 60 seconds of loading, the OFFLINE DIAGNOSTIC LOADER will indicate that it has loaded properly by displaying the following:

```
     HSC50 OFL Diagnostic Loader, Version Vnn-nn
     Radix=Octal,Data Length=Word,Reloc=00000000
ODL>
```

5) Type 'TEST OCP' in response to the Loader prompt (ODL>). The TU58 drive-motion LED will light as the test is loaded.

6) The test will indicate it has been loaded properly by displaying the following:

```
     HSC50 OFL OCP Test, Version Vnn-nn
```

7) The test now prompts for parameters. See section 2.2 below.

2.2 TEST PARAMETER ENTRY

The test first checks the position of the SECURE/ENABLE switch, via a bit in the P.ioc's Control and Status Register (address 17ØØ4Ø). If the switch is in the SECURE position, the following prompt is issued, otherwise the test skips to the next prompt.

```
Put SECURE/ENABLE switch into ENABLE position
---------------------------------------------
```

If the SECURE/ENABLE switch is in the ENABLE position, and the above prompt is issued anyway, there is a problem with the bit in the P.ioc CSR that monitors the SECURE/ENABLE switch. Refer to the troubleshooting procedures in section 5.5. The program waits until the SECURE/ENABLE switch is changed to the ENABLE position, then issues the following message:

```
(ENABLE LED should be lit, STATE LED should be blinking)
--------------------------------------------------------
```

Check that the ENABLE LED is lit, and STATE LED is blinking. There are two STATE LEDs, one is to the left of the INIT switch on the HSC5Ø's front panel, the other is located on the P.ioc module (the fourth LED from the bottom of the rightmost module in the HSC5Ø card cage). If either of the LEDs is not blinking, refer to the troubleshooting procedures in section 5. The test will now prompt for a lamp test:

```
Press FAULT (All OCP lamps should light) (Y/N) [Y] ?
----------------------------------------------------
```

Press the FAULT lamp and observe that all OCP lamps light. If none of the lamps light, there may be a problem in the lamp test logic on the OCP assembly. (Refer to OCP schematics). If one or two lamps fail to light, replace those lamps before proceeding with the test. If all lamps light properly, type a carriage-return to continue the test.

Now the program checks that all OCP switches are OFF (out position). If any switch bits in the P.ioc's Switch/Display register read as 'ones' (on), the program lights the lamps for those switches, and prompts the operator:

```
Put all lit switches in OFF (out) position (Y/N) [Y] ?
------------------------------------------------------
```

If the FAULT or INIT lamps are lit (non-locking switches), there is a problem with the wiring in those switches, or with their respective bits in the Switch/Display register.

Otherwise press all lit switches to release their locks, and type a carriage return. If the message repeats, and one or more lamps remain lit even thought the switches are OFF (out position), refer to the troubleshooting procedures in section 5.3.

The program will now test each of the OCP switches, one at a time. A switch will be lit, then the operator will be prompted:

```
Press and release the lit switch
--------------------------------
```

Press the switch that is lit. The program will allow about 1 second for the switch to be released after it is pressed, and then will continue to the next prompt. If the program fails to respond when a switch is pressed, refer to the troubleshooting procedures in section 5.3. For those switches that lock in the ON position (ONLINE and the two unmarked switches), the program will prompt:

```
Press and release the lit switch again
--------------------------------------
```

Press the switch again to return it to the OFF (out) position. If the ONLINE switch or either of the unmarked switches fails to lock in the ON position, the switch is defective and should be replaced.

After the OCP switch tests are completed, several features of the SECURE/ENABLE switch are tested. The program will begin these tests by prompting:

```
Put SECURE/ENABLE switch into SECURE position
---------------------------------------------
```

The program will wait until the SECURE/ENABLE switch is in the proper position before continuing. If the program fails to respond when the switch is moved to the SECURE position, refer to the troubleshooting procedures in section 5. Once the program detects that the switch is in the SECURE position, it will prompt:

```
(ENABLE LED should turn off)
----------------------------
```

Check that the ENABLE LED is off. If this LED fails to turn off when the switch is in the SECURE position, a short or wiring problem is likely.

The program will next prompt:

```
Press INIT (HSC should not re-boot) (Y/N) [Y] ?
-----------------------------------------------
```

Press the INIT switch. When the SECURE/ENABLE switch is in the SECURE position, pressing the INIT switch should have no effect. (Do not press any other switch or an error message will result.) If the HSC50 starts to perform a bootstrap, (INIT lamp turns on, green LED on P.ioc turns off, TU58 starts moving), the SECURE/ENABLE switch is not disabling the action of the INIT switch. Refer to the circuit schematics for the OCP assembly. After pressing INIT, type a carriage-return to continue. The test will respond with the following prompt:

```
Press terminal BREAK key (HSC should not halt) (Y/N) [Y] ?
----------------------------------------------------------
```

Press the BREAK key as directed. When in SECURE mode, the BREAK key should not cause the P.ioc's F-11 processor to halt (enter ODT). If the terminal displays the "@" character when BREAK is pressed, the SECURE/ENABLE switch is not disabling the action of the BREAK key. Refer to the troubleshooting procedures in section 5. After pressing the BREAK key, type a carriage-return to continue the test. The final prompt of the test is:

```
Put SECURE/ENABLE switch into ENABLE position
---------------------------------------------
```

The test will wait until the SECURE/ENABLE switch is returned to the ENABLE position, then the test will terminate and return to the Offline Loader.

The test may be aborted at any time by typing a control-c.

2.3 SETTING CLEARING FLAGS

N/A

2.4 PROGRESS REPORTS

N/A

SECTION 3 - ERROR INFORMATION

3.1 ERROR MESSAGE FORMAT

All error messages produced by the this test conform to the HSC5Ø Diagnostic Error message format. The first line of an error message contains general information concerning the error, and is mandatory. The second line of an error message consists of text describing the error, and is also mandatory. The third and succeeding lines of an error message are used to display additional information when required, and are optional. Listed below is a typical error message.

```
OOCP>hh:mm T#aaa E#bbb U-ØØØ
  <Text describing error>
  MA -xxxxxxxx
  EXP-yyyyyy
  ACT-zzzzzz
```

```
WHERE:
   hh       = Elapsed hours since last bootstrap
   mm       = Elapsed minutes
   aaa      = decimal number denoting test
   bbb      = decimal number denoting the error detected
   xxxxxxxx = Address of location causing the error
              (always 17ØØ42, the Switch/Display register)
   yyyyyy   = Data that was expected
   zzzzzz   = Data that was actually found
```

3.2 SPECIFIC ERROR MESSAGES

The following sections (3.2.x) describe the nature of the failure indicated by each error number.

3.2.1 ERROR ØØØ - WRONG BIT SET

This error occurs when the test detects a switch bit set in the P.ioc's Switch/Display register other than the switch bit being tested. The error can be caused by:

1. The operator pressing the wrong switch.
2. A short causes an additional switch bit to set along with the expected one.
3. A wiring error causes the wrong bit to set when a switch is pressed.

The MA (Media Address) field of the error report gives the address of the P.ioc's Switch/Display register. The EXPected and ACTual data in the error report show the switch bit the program expected to find set, and the bit or bits that were actually found to be set.

If the EXPected data and the ACTual data each consist of only one bit, then the failure was either caused by the operator pressing the wrong switch, or by a wiring error. If the ACTual data consists of two or more set bits, then a short between switches is likely. Refer to the troubleshooting procedures in section 5.3.

3.2.2 ERROR ØØ1 - BIT SET WHEN INIT PRESSED

This error occurs when the INIT switch is pressed while the HSC5Ø is in the SECURE mode (test ØØ8). This error can be caused by one of the following:

1. The operator presses some switch other than the INIT switch.
2. Pressing the INIT switch causes a switch bit in the P.ioc's Switch/Display register to set.

The MA (Media Address) field of the error report gives the address of the P.ioc's Switch Display register. The EXPected data is always zero (no bit is expected to set). The ACTual data shows the bit or bits that read as a '1' when the INIT switch was pressed. Refer to the troubleshooting procedures in section 5.3.

SECTION 4 - TEST SUMMARIES

4.1 TEST SUMMARIES

4.1.1 TEST 000 - OBSERVE ENABLE AND STATE LEDS

This test is performed by the operator, since the program can't tell whether the ENABLE or STATE LEDs are lit. If the ENABLE LED is off, there may be a wiring problem (LED not connected to power/ground source), or the LED may be faulty.

If the STATE LED on the front panel fails to blink, check the STATE LED on the P.ioc module. (Fourth LED from the bottom of the rightmost module in the HSC50 card cage). If both STATE LEDs are not blinking, the problem is likely to be caused by the bit in the P.ioc CSR register that controls the STATE LED (Refer to section 5.6). If one of the STATE LEDs is blinking, but the other is not, the non-blinking LED is probably wired wrong, or is faulty.

4.1.2 TEST 001 - LAMP TEST VIA FAULT SWITCH

The OCP's FAULT switch performs an automatic lamp test. When the FAULT switch is pressed, all lamps should light until the switch is released.

If none of the lamps light when FAULT is pressed, the problem is likely to be in the lamp test circuitry on the OCP assembly. (There is also a chance that all lamps are bad, or that the lamps are not installed). Refer to the schematics for the OCP assembly to localize the problem.

If some lamps light when FAULT is pressed, but others do not, replace the lamps that do not light, then press FAULT again. If the lamps still fail to light, there is a wiring problem with the non-illuminated lamps. Refer to the schematics for the OCP assembly to localize the problem.

4.1.3 TEST 002 - CHECK ALL SWITCHES OFF

In order to complete the remaining tests properly, all switches must start out in the OFF (out) position. The program reads the P.ioc's Switch/Display register to see if any of the switch bits read as ON (switch bit is a 'one'). If the bit for any switch reads as ON, the corresponding lamp is lit, and the operator is prompted to turn off any switch that is lit. The program will not proceed until all switch bits read as OFF.

If a lamp remains ON, even though the corresponding switch is OFF (out position), the switch is either wired incorrectly, or the bit in the P.ioc's Switch/Display register for that switch is faulty. Refer to section 5.3 to localize the problem.

4.1.4 TEST 003 - FAULT SWITCH

In this test the FAULT lamp is lit, and the operator is directed to press the lit switch. The program then monitors the switch bits in the P.ioc's Switch/Display register, and waits for the FAULT switch bit to set. If any other switch bit sets, an error is reported and the program terminates.

If pressing the FAULT switch has no effect, one of the following could be the cause:

1. The FAULT switch is broken.

2. The FAULT switch is not wired properly.

3. The FAULT switch bit in the P.ioc's CSR can not be set.

Refer to the troubleshooting procedures in section 5.3.

If pressing the FAULT switch results in a error message, refer to section 3.2.1.

4.1.5 TEST 004 - ONLINE SWITCH

In this test the ONLINE lamp is lit, and the operator is directed to press the lit switch. The program then monitors the switch bits in the P.ioc's Switch/Display register, and waits for the ONLINE switch bit to set. If any other switch bit sets, an error is reported and the program terminates.

Once the ONLINE switch bit sets in the P.ioc's Switch/Display register, the program directs the operator to press the lit switch again, to return it to the OFF (out) position. Then the program waits until the switch bit reads as OFF (0) before proceeding to the next test.

If pressing the ONLINE switch has no effect, one of the following could be the cause:

1. The ONLINE switch is broken.

2. The ONLINE switch is not wired properly.

3. The ONLINE switch bit in the P.ioc's CSR can not be set.

Refer to the troubleshooting procedures in section 5.3.

If pressing the ONLINE switch results in a error message, refer to section 3.2.1.

4.1.6 TEST ØØ5 - FIRST UNMARKED SWITCH

In this test the FIRST UNMARKED lamp is lit, and the operator is directed to press the lit switch. The program then monitors the switch bits in the P.ioc's Switch/Display register, and waits for the FIRST UNMARKED switch bit to set. If any other switch bit sets, an error is reported and the program terminates.

Once the FIRST UNMARKED switch bit sets in the P.ioc's Switch/Display register, the program directs the operator to press the lit switch again, to return it to the OFF (out) position. Then the program waits until the switch bit reads as OFF (Ø) before proceeding to the next test.

If pressing the FIRST UNMARKED switch has no effect, one of the following could be the cause:

1. The FIRST UNMARKED switch is broken.

2. The FIRST UNMARKED switch is not wired properly.

3. The FIRST UNMARKED switch bit in the P.ioc's CSR can not be set.

Refer to the troubleshooting procedures in section 5.3.

If pressing the FIRST UNMARKED switch results in a error message, refer to section 3.2.1.

4.1.7 TEST 006 - SECOND UNMARKED SWITCH

In this test the SECOND UNMARKED lamp is lit, and the operator is directed to press the lit switch. The program then monitors the switch bits in the P.ioc's Switch/Display register, and waits for the SECOND UNMARKED switch bit to set. If any other switch bit sets, an error is reported and the program terminates.

Once the SECOND UNMARKED switch bit sets in the P.ioc's Switch/Display register, the program directs the operator to press the lit switch again, to return it to the OFF (out) position. Then the program waits until the switch bit reads as OFF (0) before proceeding to the next test.

If pressing the SECOND UNMARKED switch has no effect, one of the following could be the cause:

1. The SECOND UNMARKED switch is broken.

2. The SECOND UNMARKED switch is not wired properly.

3. The SECOND UNMARKED switch bit in the P.ioc's CSR can not be set.

Refer to the troubleshooting procedures in section 5.3.

If pressing the SECOND UNMARKED switch results in a error message, refer to section 3.2.1.

4.1.8 TEST 007 - ENABLE LED OFF

The test begins by prompting the operator to put the SECURE/ENABLE switch into the SECURE position. The program waits until bit 15 of the P.ioc's Control and Status register reads as a '0', indicating that the switch has entered the SECURE position. Then the program tells the operator to observe that the ENABLE LED has turned off.

If ENABLE LED fails to turn off when the switch is in the SECURE position, check the wiring of the ENABLE LED.

4.1.9 TEST ØØ8 - INIT SWITCH IN SECURE MODE

This test checks that the INIT switch has no effect when the SECURE/ENABLE switch is in the SECURE position. The operator is prompted to press the INIT switch, while the program monitors the switch bits in the P.ioc's Switch/Display register to insure that pressing the INIT switch doesn't cause any switch bits to set.

If pressing the INIT switch causes the HSC5Ø to re-boot, the SECURE position of the SECURE/ENABLE switch is not disabling the INIT switch. Refer to the circuit schematics for the OCP assembly to localize the problem.

If pressing the INIT switch causes one of the switch bits in the Switch/Display register to set, an error message will be displayed. Refer to section 3.2.2 for further information.

4.1.1Ø TEST ØØ9 - BREAK KEY IN SECURE MODE

This test checks that the terminal's BREAK key has no effect when the SECURE/ENABLE switch is in the SECURE position. (Normally the BREAK key causes the P.ioc's F-11 CPU to halt and enter ODT). The operator is prompted to press the BREAK key, and to observe that the HSC5Ø does not halt.

If pressing the BREAK key causes the terminal to print "@", the SECURE position of the SECURE/ENABLE switch is not disabling BREAK from halting the F-11 CPU. Refer to the CREF listing at the back of the P.ioc circuit schematics, and find the signal named "TERM ENA H". This signal should disable BREAK (called "TERM FRM ERR H" in the schematics).

SECTION 5 - TROUBLESHOOTING PROCEDURES

5.1 P.IOC SWITCH DISPLAY REGISTER LAYOUT

ADDRESS 170042 VIA ODT

```
             +--------- 4000(8) FAULT SWITCH
             |  +------ 2000(8) ONLINE SWITCH
             |  |  +--- 1000(8) 1st UNMARKED SWITCH
             |  |  |  +- 400(8) 2nd UNMARKED SWITCH
             |  |  |  |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|15|14|13|12|11|10| 9| 8| 7| 6| 5| 4| 3| 2| 1| 0|
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
 |  |  |  |              |  |  |  |  |  |  |  |
 ( UNUSED )              |  |  |  |  |  |  |  |
                         |  |  |  |  |  |  |  |
      200(8) GREEN LED --+  |  |  |  |  |  |  |
      100(8) CMEM/DMEM NXM -+  |  |  |  |  |  |
       40(8) INH PARITY TRAP --+  |  |  |  |  |
       20(8) INIT LAMP -----------+  |  |  |  |
       10(8) FAULT LAMP -------------+  |  |  |
        4(8) ONLINE LAMP ---------------+  |  |
        2(8) FIRST UNMARKED LAMP ----------+  |
        1(8) SECOND UNMARKED LAMP ------------+
```

5.2 P.IOC CONTROL AND STATUS REGISTER LAYOUT

ADDRESS 170040 VIA ODT

```
 +--------------------- 100000(8) 0 WHEN SECURE
 |  +------------------- 40000(8) ALWAYS 0
 |  |  +---------------- 20000(8) MARGIN ENABLE
 |  |  |  +------------- 10000(8) MARGIN HIGH
 |  |  |  |  +----------- 4000(8) SWAP BOARD
 |  |  |  |  |  +-------- 2000(8) SWAP BANK
 |  |  |  |  |  |  +----- 1000(8) SELECT P 1
 |  |  |  |  |  |  |  +--- 400(8) SELECT P 0
 |  |  |  |  |  |  |  |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|15|14|13|12|11|10| 9| 8| 7| 6| 5| 4| 3| 2| 1| 0|
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
                         |  |  |  |  |  |  |  |
200(8) ENA CMEM ARB -----+  |  |  |  |  |  |  |
100(8) I/O CLEAR -----------+  |  |  |  |  |  |
 40(8) HI BYTE PARITY TEST ----+  |  |  |  |  |
 20(8) LO BYTE PARITY TEST -------+  |  |  |  |
 10(8) STATE LED --------------------+  |  |  |
  4(8) NON-MEMORY-ACCESS (NMA) ---------+  |  |
  2(8) CONTROL MEMORY INTERRUPT ENABLE ----+  |
  1(8) CONTROL MEMORY LOCK CYCLE ENABLE ------+
```

5.3 CHECKING A SWITCH BIT VIA ODT

To check the operation of one of the HSC50's switches, do the following:

1. With the SECURE/ENABLE switch in the ENABLE position, press the terminal's BREAK key.
2. The P.ioc's F-11 CPU should halt and display a '@'.
3. Type: 170042/
4. The contents of address 170042 (the P.ioc's Switch Display register) will be displayed in octal. Refer to the layout of Switch Display register in section 5.1 to locate the switch bits. Each bit will be in the '1' state when the associated switch is ON (pressed in).
5. Type a carriage-return.
6. The user may now type just a '/' to re-examine the Switch Display register.
7. To re-start the Offline Loader (or the diagnostic that was interrupted), type a carriage-return, then type a 'P' followed by another carriage-return.

Using the method described above, the switch bits of the Switch/Display register can be monitored when various switches are in the ON or OFF position.

5.4 CHECKING A LAMP BIT VIA ODT

To check the operation of the lamp control bits in the P.ioc's Switch Display register, use the following method:

1. With the SECURE/ENABLE switch in the ENABLE postion, press the terminal's BREAK key.

2. The P.ioc's F-11 CPU should halt and display a '@'.

3. Type 170042/

4. The contents of the Switch/Display register will be displayed in octal.

5. Use the chart in section 5.1 of this document to locate the bits that control the OCP lamps.

6. When a lamp bit is set, the corresponding lamp should be lit.

7. To light a lamp, type the octal value that corresponds to the proper lamp, then type a carriage return. The lamp should light.

8. The user may now type just a '/' to re-examine the contents of the Switch/Display register.

9. To re-start the Offline Loader (or the diagnostic that was interrupted), type a carriage-return, then type a 'P' followed by another carriage-return.

Using the method given above, various lamps can be manually enabled or disabled.

5.5 CHECKING SECURE/ENABLE SWITCH VIA ODT

To manually check the operation of the SECURE/ENABLE bit in the P.ioc's Control and Status register, use the following procedure:

1. With the SECURE/ENABLE switch in the ENABLE position, press the terminal's BREAK key. (If the HSC5Ø is stuck in the SECURE mode, this method can not be used, since BREAK will be disabled.)

2. The P.ioc's F-11 CPU will halt and display a '@'.

3. Type 17ØØ4Ø/

4. The contents of the P.ioc's Control and Status register will be displayed in octal. Refer to the chart in section 5.2 to identify the various bits of this register.

5. When the SECURE/ENABLE switch is in the ENABLE position, the contents of the register should be 1xxxxx. When in the SECURE position, the contents should be Øxxxxx.

6. To re-examine the register, type a carriage-return and a '/'.

7. To re-start the Offline Loader (or the diagnostic that was interrupted), type a carriage-return, then type a 'P' followed by another carriage-return.

Using the method given above, the SECURE/ENABLE bit in the P.ioc's CSR can be checked with the SECURE/ENABLE switch in both positions.

5.6 CHECKING STATE LED VIA ODT

There are two STATE LEDs in the HSC50. One is on the front panel, to the left of the INIT switch. The other STATE LED is on the P.ioc module (rightmost module in the HSC50 card cage, fourth LED from the bottom of the module). Both LEDs are controlled by a bit in the P.ioc's Control and Status register (see chart in section 5.2 for a layout of this register). To manually control the STATE LED, use the following procedure:

1. With the SECURE/ENABLE switch in the ENABLE position, press the terminal's BREAK key.

2. The P.ioc's F-11 CPU should halt and display a '@'.

3. Type 170040/

4. The contents of the Control and Status register will be displayed in octal.

5. Use the chart in section 5.2 to find the octal value that corresponds to the STATE LED.

6. To light the STATE led, type the octal value that corresponds to the STATE LED, followed by a carriage-return. To extinguish the STATE LED, put a zero in the same bit position. See CAUTION below.

7. The contents of the P.ioc's CSR can be re-examined by just typing a '/'.

8. To re-start the Offline Loader (or the diagnostic that was interrupted), type a carriage-return, then type a 'P' followed by another carriage-return.

CAUTION: Bit 7 of the P.ioc's CSR must be set to allow the HSC50's K's to access Control memory. The setting of other bits in the CSR can result in strange side-effects. Be careful not to set any bits except the STATE LED bit, and leave bit 7 set when you are done.

HSC50 Off-Line Bus Interaction Test (TEST BUS)

USER DOCUMENTATION

PRODUCT CODE:	XX-XXXXX-XX
PRODUCT NAME:	HSC50 Offline Bus Interaction Test
PRODUCT DATE:	12-APR-1983
MAINTAINER:	Diagnostic Engineering - Colorado (CX)
AUTHOR:	Mike Hare

TABLE OF CONTENTS

SECTION 1 - GENERAL INFORMATION

1.1 PROGRAM ABSTRACT

The Offline Bus Interaction Test creates Control and Data bus contention among the K's in the HSC5Ø subsystem. The contention is generated by simultaneously testing different portions of the same memory (control and/or data) from different K's. In the process of testing the memories, the various K's in the subsystem will contend with each other for the use of the control and data buses.

In addition to the bus contention generated by the K's, the user can select P.ioc interaction with the Program, Control, and Data memories. The user can also select P.ioc interaction with the Operator Control Panel (OCP), and/or the TU58. If P.ioc interaction is selected, it will occur simultaneously with the bus contention generated by the K's.

This test requires a minimum of two working K's in order to operate, and will use a maximum of 7 K's if they are available. The more K's that are available for use by this test, the greater the amount of bus contention. A larger number of K's makes it easier to isolate failures to a particular source. The run time of this test increases linearly as the number of K's is increased.

Errors are reported on the console terminal as they occur.

1.2 SYSTEM REQUIREMENTS

Hardware Required:

a) P.ioc (processor) module with HSC5Ø Boot ROMS.
b) At least one M.std (memory) module.
c) TU58 controller with at least one working drive.
d) Terminal connected to P.ioc's console interface.
e) At least two working 'K's (K.sdi, K.sti, or K.ci).
f) Working Control and Data Memory

Software Required:

a) HSC5Ø Offline Diagnostic Tape (TU58 Media)

1.3 RELATED DOCUMENTS

a) HSC5Ø Offline Diagnostic Loader User Document
b) HSC5Ø Offline P.ioc Test User Document
c) HSC5Ø Offline K Test Selector User Document
d) HSC5Ø Offline K/P Memory Test User Document
e) HSC5Ø Offline System Sizer User Document
f) HSC5Ø K Microdiagnostic User Document
g) HSC5Ø ROM Bootstrap User Document
h) TU58 User's Guide

1.4 HARDWARE ASSUMED TO BE WORKING

In the process of loading the Offline Diagnostic Loader, several diagnostics are run. The ROM Bootstrap tests the basic F-11 instruction set, tests a partition in Program Memory, and tests the TU58 used for the boot. Then the bootstrap loads the Offline P.ioc Test, which completes the F-11 tests and tests the remainder of the P.ioc module. After these tests, the Offline Diagnostic Loader is loaded from the TU58 to memory, and control is passed to the Loader. When the operator instructs the Loader to 'TEST BUS', the Offline Bus Interaction test is loaded and started. Due to the sequence of tests that precede the memory test, the memory test assumes that the P.ioc module and the TU58 are tested and working. This test also assumes that the Control and Data memories were previously tested and found to be working via the Offline Memory Test or the Offline K/P Memory Test.

SECTION 2 - OPERATING INSTRUCTIONS

2.1 START-UP PROCEDURE

Follow the steps below to start the Bus Interaction test:

1) Insert the HSC5Ø OFFLINE TESTS cassette into the TU58's Unit Ø drive. (left hand drive)

2) Power-on the HSC5Ø, then depress and release the INIT button on the HSC5Ø front panel.

3) The TU58 drive-motion LED should light within 1Ø seconds, indicating that the Bootstrap is loading the OFFLINE DIAGNOSTIC LOADER to Program Memory.

4) After about 6Ø seconds of loading, the OFFLINE DIAGNOSTIC LOADER will indicate that it has loaded properly by displaying the following:

```
      HSC5Ø OFL Diagnostic Loader, Version Vnn-nn
      Radix=Octal,Data Length=Word,Reloc=ØØØØØØØØ
  ODL>
```

5) Type TEST BUS in response to the Loader prompt (ODL>). The TU58 drive-motion LED will light as the bus interaction test is loaded.

6) The test will indicate it has been loaded properly by displaying the following:

```
      HSC5Ø OFL Bus Interaction Test, Version Vnn-nn
```

7)The test now sizes the Program, Control and Data memories, and determines the number of K's available for testing. See next section.

2.2 TEST PARAMETER ENTRY

After displaying the program name and version, the Program, Control, and Data memories are sized. The bounds of each memory are displayed on the terminal. Then the user is prompted to select the K's that should be used for the test, as follows:

```
Use requestor #001, K.ci (Y/N) [Y] ?
```

Answer with a carriage-return, (or a Y followed by a carriage-return) if the K should be used. Answer with an N followed by a carriage-return if the K should not be used. (See section 5.4.2 for a discussion of how failures are isolated by leaving certain K's out of the test configuration).

At least two working K's must be used to run the bus contention test, since one K cannot generate bus contention by itself. The program will display the following error message if less than 2 K's remain after the user has indicated which K's should be used:

```
Not Enough K's Available for Test
```

Next the program will prompt for the type of P.ioc interaction desired:

```
P.ioc Memory Interaction desired (Y/N) [Y] ?
```

Answer the prompt with a carriage-return if P.ioc interaction with memory is desired. Answer with an 'N' followed by a carriage-return if P.ioc interaction with memory is not desired. If the prompt is answered with an 'N', the next three prompts below will be skipped. If the prompt is answered with a carriage-return, the following prompts are displayed:

```
Interact with Program Memory (Y/N) [Y] ?
Interact with Control Memory (Y/N) [Y] ?
Interact with Data Memory (Y/N) [Y] ?
```

For each prompt, answer with a carriage-return if you want the P.ioc to interact with the specified memory while the K's are generating contention on the Control and Data buses. Answer with an 'N' followed by a carriage-return if you do not want the P.ioc to interact with the specified memory. (If P.ioc interaction is selected, the P.ioc will interact with the memory at the same time the K's are generating Control and Data bus contention.)

The program now prompts for OCP interaction:

```
OCP Interaction Desired (Y/N) [Y] ?
```

If P.ioc interaction with the Operator Control Panel (OCP) is desired, answer with a carriage-return. If OCP interaction is not desired, answer with a 'N', followed by a carriage-return. The test will now prompt for TU58 interaction:

```
Interact with TU58 (Y/N) [Y] ?
```

If P.ioc interaction with the TU58 is desired, answer with a carriage-return. If TU58 interaction is not desired, answer with an 'N', followed by a carriage-return. The program will now prompt:

```
Number of passes to perform (D) [1] ?
```

Enter a decimal number between 1 and 2,147,483,647 (omit commas), which specifies the number of times the bus interaction test should be repeated. (Entering a 0, or just a carriage-return, will result in 1 pass being performed.) After the number of passes is entered, the bus contention test will begin. The test can be aborted at any time by typing a control-c. (The test may continue running for a few seconds after control-c is typed.)

NOTE: For any of the above prompts, use the DELETE key to delete mis-typed parameters before the terminating carriage-return is typed. If you note an error in a parameter that has already been terminated with a carriage-return, type a control-c to return to the Offline Loader. Then type 'START', followed by carriage-return, to re-start the test from the beginning.

2.2.1 REUSING PARAMETERS

After the number of passes specified have been completed, the following prompt will be issued:

```
Re-use parameters (Y/N) [Y] ?
```

Answering this prompt with a carriage-return, or a 'Y' followed by a carriage-return, results in repeating the last test specified, using the parameters that were previously used. Answering the prompt with a 'N' followed by a carriage-return will cause the test to prompt for new parameters. Answering the prompt with a control-c will return control to the Offline Loader.

2.3 SETTING CLEARING FLAGS

N/A

2.4 PROGRESS REPORTS

Each time the program completes one full set of bus contention tests, an end of pass report is displayed. A pass consists of completing a full set of contention tests, including: Control Bus Tests, Data Bus Tests, and Combined Control and Data Bus Tests. The end of pass message is displayed as follows:

```
End of Pass nnnnnn, xxxxxx errors, yyyyyy total errors.
```

Where:

1. 'nnnnnn' = decimal count of the number of passes completed

2. 'xxxxxx' = decimal count of the number of errors detected on current pass

3. 'yyyyyy' = decimal count of the total number of errors detected since the test was initiated

SECTION 3 - ERROR INFORMATION

3.1 ERROR MESSAGE FORMAT

All error messages produced by this test conform to the HSC50 Diagnostic Error message format. The first line of an error message contains general information concerning the error, and is mandatory. The second line of an error message consists of text describing the error, and is also mandatory. The third and succeeding lines of an error message are used to display additional information when required, and are optional. Listed below is a typical error message.

```
OBIT>hh:mm T#aaa E#bbb U-000
  Memory Test Error
  Detected By K.sdi, requestor 006
  MA -xxxxxxxx
  EXP-yyyyyy
  ACT-zzzzzz
  <K-Error-Summary-Info>
  Memory Test Configuration:
    K.ci , requestor 001, M.ctl 16000700 - 16100274
    K.sdi ,requestor 006, M.ctl 16100300 - 16177674
```

```
WHERE:
   hh       = Hours since Offline Loader was last booted
   mm       = Minutes since Offline Loader was last booted
   aaa      = decimal number denoting test
   bbb      = decimal number denoting the error detected
   xxxxxxxx = Address of location causing the error
   yyyyyy   = Data that was expected
   zzzzzz   = Data that was actually found
   <K-Error-Summary-Info>    = Refer to section 3.1.1.
   Memory Test Configuration = Refer to section 3.1.2
```

3.1.1 K ERROR SUMMARY INFORMATION

When the K reports a memory test failure to the P.ioc, the following information is supplied:

1. The address of the failing memory location.

2. The data expected and data actually found.

3. Error summary information.

The error summary information is supplied as a 3-bit field, which includes the following:

1. A bit indicating a Parity error occurred while reading the location.

2. A bit indicating a NXM error occurred while accessing the location.

3. A bit indicating Control Bus (CBUS) error occurred while accessing the location.

When a memory error report is issued for an error detected by the K, the last line of the error report will include a list of the error summary bits that were set (if any).

A Control Bus (CBUS) Error indicates that the K asserted an illegal combination of the three "CCYCLE" lines when accessing control memory. Since these lines were previously tested from the P.ioc (in the OFL P.IOC TEST), a Control Bus Error is most likely to be caused by a problem with the K's drivers that assert the "CCYCLE" lines.

3.1.2 MEMORY TEST CONFIGURATION

The memory test configuration lists each K being used for bus interaction tests, along with the section of memory that each K was testing when the failure occurred. The configuration information consists of:

1. The type of K (K.ci, K.sdi, K.sti), and requestor #.

2. The memory being tested by the K (M.ctl = control memory, M.data = data memory).

3. The first address of the chunk of memory being tested.

4. The last address of the chunk of memory being tested.

3.2 SPECIFIC ERROR MESSAGES

The following sections (3.2.x) describe the nature of the failure indicated by each error number.

3.2.1 ERROR 000 - MEMORY TEST ERROR

Error 000 indicates that one of the K's detected a memory error in the Control or Data memories. The following is a sample error report:

```
Memory Test Error
Detected by K.ci , requestor 001
MA -16010234
EXP-000177
ACT-000377
Parity error
Memory Test Configuration:
  K.ci  ,requestor 001, M.ctl 16000700 - 16100274
  K.sdi ,requestor 007, M.ctl 16100300 - 16177674
```

MA = The 22-bit address of the failing location.
EXP= The data pattern expected by the K
ACT= The data pattern found by the K
MEMORY CONFIGURATION = other K's enabled when failure occurred.

This sample error report indicates that the K.sdi detected a memory parity error while reading address 16010234 of the Control memory (M.ctl). The K expected to find the value 000177 in the location, but instead found the value 000377. At the time the error occurred, the K.ci in requestor 1 was testing addresses 16000700 thru 16100274 of the Control memory, and the K.sdi in requestor 7 was testing addresses 16100300 thru 16177674 of the Control memory.

3.2.2 ERROR 001 - K TIMED-OUT DURING INIT

This error is displayed when a K fails to complete its Init sequence in time. This error usually indicates that the specified K failed one of its internal microdiagnostics. A sample error report follows:

```
K Timed-out During Init
K.ci , requestor 001, Status = 104
Other K's Enabled:
  K.sdi, requestor 6
  K.sdi, requestor 7
```

This sample error report indicates that the K.ci in requestor 1 did not finish its initialization diagnostics in the required time. The K Status displayed with the error report indicates that the K failed test 4 of its microdiagnostics. (1XX in status = failed test XX). Two other K's were enabled at the time the K.ci timed-out. (One of these K's may be responsible for K.ci's time-out).

When the P.ioc enables the K to perform the memory test, the K begins its Init sequence (which includes executing certain microdiagnostics). At the end of the K's Init sequence, the K indicates that it found the K-Control-Area by complementing a pointer word in the Control memory. If the K fails to complement this pointer word within 50 milli-seconds (4.2 seconds for K.ci) of being enabled, an error 001 is reported.

The contents of the 'K Status Register' are displayed with the error report. Refer to the 'K Microdiagnostic User Document' (or the appendices of the 'Offline K Test Selector User Document) for further information on K status values.

3.2.3 ERROR 002 - K Timed-out During Test

This error indicates that the specified K failed to complete its memory test within the expected time. A sample error report is as follows:

```
K Timed-out During Test
K.sdi, requestor 007, Status = 002
Memory Test Configuration:
K.ci , requestor 1, M.ctl 16000700 - 16100274
K.sdi, requestor 7, M.ctl 16100300 - 16177674
```

The sample error report indicates that the K.sdi in requestor 7 never completed the memory test it was assigned. (K's are allowed up to 1 minute to complete a memory test). The memory configuration displayed with the error report shows all K's that were testing at the same time the K.ci timed-out. In this example, the K.ci in requestor 1 was also testing at the time the K.sdi timed-out.

Test time-out failures may be caused by a failure in the K that timed-out, but they may also be caused by a failure in one of the other K's that were testing at the same time. Refer to section 5.4 for failure isolation techniques.

3.2.4 ERROR 003 - Parity Trap

ERROR 003 indicates the P.ioc detected a Parity error. The 22-bit address of the location that caused the error is displayed as the 'MA' data in the error report.

```
 MA = The address causing the parity trap.
VPC = The Virtual PC of the memory test at the time the
      trap occurred.  Reference this address in the
      listing to locate the area of the test where
      the error occurred.
```

Since the data is lost when a parity trap occurs, no EXPected or ACTual data can be displayed.

3.2.5 ERROR 004 - NXM Trap

Error 004 indicates the P.ioc detected a Non-Existent-Memory error. An NXM error is caused when no memory responds to a particular address. The 'MA' data in the error report indicates the address which produced the NXM trap. After the trap is reported, the program will attempt to restart the test from the beginning.

```
 MA = The address causing the NXM trap.
VPC = The virtual PC of the memory test at the time the
      trap occurred.  Reference this address in the
      listing to locate the area of the test where
      the error occurred.
```

If this error occurs at a memory address that should be in your memory configuration, the memory in question is not supplying an 'ACK' to the P.ioc when the specified address is presented on the memory bus. The most probable point of failure is the logic on the memory module that compares addresses on the memory bus to the range of addresses to which the module should respond. The comparitor itself could be faulty, or the [C IN, C OUT], [D IN, D OUT], or [P IN, P OUT] lines on the backplane could be in error.

3.2.6 ERROR ØØ5 - Memory Test Error (P.ioc Detected)

This error report indicates that the P.ioc detected an error while testing the Program memory. This error can only occur if P.ioc interaction with the Program memory is selected. P.ioc interaction with the Program memory consists of:

1. a series of PDP-11 instructions that perform Read/Modify/Write (RMW) cycles to selected Program memory locations.

2. quick-verify tests of the entire Program memory, (done 6 kwords at time).

An error ØØ5 can be caused by cross-talk between the Program memory bus and either the Control or Data bus. This error can also be caused by a failure in the Program memory logic which inhibits refresh cycles in the middle of a RMW cycle.

3.2.7 ERROR ØØ6 - TU58 Sync Failure

Error ØØ6 indicates that the P.ioc failed in an attempt to synchronize with the TU58. The TU58 sync sequence consists of the P.ioc sending a <BREAK> to the TU58, followed by sending the code ØØ4(8) twice. The P.ioc then allows a maximum of 2 seconds for the TU58 to respond with the code Ø2Ø(8). An error ØØ6 indicates that the TU58 failed to respond to the sync sequence within 2 seconds. This error may be caused by cross-talk between the TU58 signal wires and the wires connected to the Operator Control Panel (OCP), particularly if OCP interaction is enabled. This error can also be caused by a marginal problems in the P.ioc's UART, the TU58's UART, or in the signal path between the two.

3.2.8 ERROR ØØ7 - UART Error

Error ØØ7 indicates an error detected by the P.ioc's UART. The third line of the error report specifies whether the error was a Framing error or a Parity error.

Framing errors can be caused by a noise-burst on the signal wire which connects the TU58's transmitter to the P.ioc's receiver, or by a failure in the P.ioc's or the TU58's UART.

Parity errors should never occur, since the P.ioc's UART is designed to receive 8 data-bits and no parity bit. If a parity error occurs, there is a failure in the P.ioc's UART circuit.

3.2.9 ERROR 008 - TU58 Packet Checksum Error

Error 008 indicates that the P.ioc detected a checksum error in a command or data packet from the TU58. Packets are a series of bytes in the Radial Serial Protocol format (refer to TU58 User's Guide for further detail concerning the Radial Serial Protocol). Each packet contains two checksum bytes at the end. The packet checksum is supposed to equal the sum of all the preceeding bytes in the packet. An error 008 indicates that the sum of all the bytes in a packet was not equal to the checksum at the end of the packet. This error can be caused by:

1. Dropping or picking up a bit in one of the bytes of a packet.

2. One of the bytes in a packet being lost during transmission.

3. A spontaneous TU58 initialize in the middle of a packet transmission.

This error can only occur if TU58 interaction is selected. The possible causes of this error include a faulty UART in either the P.ioc or the TU58 controller, a poorly connected serial line, or crosstalk on the serial line. If this error only occurs when TU58 and OCP interaction are both selected, the serial line is probably picking up noise from the OCP wires (the TU58 serial lines and the OCP conductors share a common cable harness).

3.2.10 ERROR 009 - TU58-Detected Error

Error 009 indicates that the TU58 detected an error in a command packet transmitted by the P.ioc, or the TU58 detected an error while reading tape. The third line of the error report indicates the type of failure that was detected. Following is a list of error messages and the possible causes of each error:

1. SPONTANEOUS INITIALIZE - The TU58 will initialize itself if it detects a checksum error in a command packet from the P.ioc. It will also perform a spontaneous initialize if it detects a protocol error (refer to the TU58 User's Guide). Refer to section 3.2.9 for the possible causes of checksum errors.

2. END OF MEDIUM - The TU58 encountered the end of tape while performing a read command. Since the P.ioc does not direct the TU58 to read beyond the end of tape, the TU58 probably lost its position on the tape. The probable cause is a failure in the TU58 controller or drive electronics.

3. NO CASSETTE MOUNTED - The TU58 detected no cassette in a drive while trying to read from tape. This error can be caused by removing the TU58 cartridge while the program is exercising the drive. It can also be caused by a failure in the micro-switch that detects that a cartridge is inserted in a drive.

4. HARD READ ERROR - The TU58 detected an unrecoverable read error while reading from tape. The probable cause is a bad spot on the TU58 media. The tape cassette should be replaced.

5. MOTOR STOPPED - The TU58 stopped its drive motor because the tape either broke or came unwound from the spool in the cassette. This error can also be caused by the failure of one of the wires that supply power to the drive motor.

6. SEEK ERROR - The TU58 could not position the tape to the block specified in a read command. The probable cause is defective TU58 media. The TU58 cassette should be replaced.

7. BAD RECORD NUMBER - The TU58 detected an illegal block number in a read command from the P.ioc. The probable cause is a garbled read command resulting from noise on the serial line between the P.ioc and the TU58.

8. BAD OP-CODE - The TU58 detected an illegal op-code in a read command sent by the P.ioc. The probable cause is a garbled read command resulting from noise on the

serial line between the P.ioc and the TU58.

9. BAD UNIT NUMBER - The TU58 detected an illegal unit number in a read command from the P.ioc. The only legal unit numbers are 0 and 1. The probable cause is a garbled read command caused by noise on the serial line between the P.ioc and the TU58.

10. WRITE PROTECTED - The TU58 tried to do a write operation to a cassette that was write locked. Since the Bus Interaction test does not do any write operations, the TU58 must have received a read command that was interpreted as a write command. This can be caused by noise on the serial line between the TU58 and the P.ioc, or by an internal TU58 error which caused a read command to be misinterpreted.

11. FAILED SELF-TEST - The TU58 reported a failure in its self-test routine. Refer to the TU58 User's Guide for further information. The Bus Interaction test does not command the TU58 to perform its self-test.

12. DATA CHECK ERROR - The TU58 detected an error while reading back data that was just written to tape. This error can only occur during a write command with the read-verify modifier. The Bus Interaction test does not send any write commands to the TU58, so the TU58 must have received a garbled read command and interpreted the command as a write with read-verify. The TU58 may also have received a valid read command and mis-interpreted the command as a write due to internal errors in the controller.

13. UNKNOWN STATUS VALUE, STATUS = xxx - The P.ioc received an end packet from the TU58 which contained an unknown status value. The xxx in the message is the status value received. The probable cause is an internal error in the TU58 controller. This error might also be caused by noise on the serial line between the P.ioc and the TU58.

SECTION 4 - TEST SUMMARIES

4.1 TEST SUMMARIES

The test numbers displayed in error reports fall into one of the following ranges:

1. Tests 010 thru 017 are Control Bus contention tests.

2. Tests 020 thru 027 are Data Bus Contention tests.

3. Tests 030 thru 037 are Combined Control and Data Bus contention tests.

In addition to the numbered tests, the P.ioc can also be directed to perform various interaction tests while the K's are generating bus contention. The following sections (4.1.x) summarize the various tests.

4.1.1 TESTS 010 thru 017 - CONTROL BUS CONTENTION

<TBS> (Refer to section 5.3 for now)

4.1.2 TESTS 020 thru 027 - DATA BUS CONTENTION

<TBS> (Refer to section 5.3 for now)

4.1.3 TESTS 030 thru 037 - COMBINED BUS CONTENTION

<TBS>

4.1.4 P.IOC Program MEMORY INTERACTION

4.1.5 P.IOC CONTROL MEMORY INTERACTION

4.1.6 P.IOC DATA MEMORY INTERACTION

4.1.7 P.IOC OCP INTERACTION

4.1.8 P.IOC TU58 INTERACTION

4.2 K MEMORY TEST ALGORITHM

The Moving Inversions Memory Test (MOVI) is used to generate bus contention among the K's. Each K in an HSC50 contains the Moving Inversions test as part of its microdiagnostic firmware set.

The Moving Inversions RAM test is used to detect data and addressing problems in dynamic semiconductor memories.

MOVING INVERSIONS ALGORITHM

1) Write 000000 in each location being tested.

2) Read all locations in order from lowest to highest. After reading a location and checking for a zero, rewrite the same location with a single one in the least significant bit. Then reread the location and verify that the write worked correctly.

3) Again read all locations in order from lowest to highest, checking each location to contain the data previously written, and rewriting the data found with a single additional 'one' bit, and rereading to check that the write worked properly.

4) Repeat step 3 until the test pattern consists of a word containing all 'ones' (pattern 177777).

5) Repeat steps 1 thru 4, but this time start at the highest memory address each time and work down to the lowest, but instead of adding an additional '1', add an additional '0'. This will work each memory location from all 'ones' back to all 'zeroes'

6) End of test. All memory is cleared to 000000.

SECTION 5 - TEST PHILOSOPHY

The basic idea behind the Bus Interaction Test is to exercise the Control and Data Bus arbitration logic. Bus activity is generated in such a manner to guarantee that various K's will contend with each other for the use of the Control and Data buses. In the process of testing contention for the buses, the ability of the buses to handle back-to-back cycles is also tested.

5.1 HARDWARE TESTED

1. P.IOC's CONTROL AND DATA BUS ARBITRATION LOGIC
2. CONTROL AND DATA BUS SETTLING DURING BACK TO BACK CYCLES
3. M.STD LOGIC DURING BACK-TO-BACK MEMORY CYCLES
4. INDEPENDENCE OF OCP AND TU58 SIGNAL WIRES

5.2 FIRMWARE TESTED

1. K MEMORY TEST FIRMWARE - The K's memory test firmware is tested to insure that it can function properly when contention exists on the Control and/or Data Bus.

5.3 TEST METHOD

After the HSC50 has been sized to determine available memory and K's, memory bus contention is generated as follows:

1. Control Bus Contention - The available Control memory (minus K Control Areas) is divided into as many 'chunks' as there are available K's. For the initial test, the K with the lowest requestor number is assigned to test the chunk of memory with the lowest starting address. The K with the next highest requestor number is assigned to test the next chunk of memory, and so on until each K has been assigned a chunk of Control memory to test. The parameters for each K are stored in its Control Area, and the K's are enabled (released from the initialized state). When all K's have indicated that they have found their parameters and are ready to test, all K's are commanded to begin testing. Since all K's will now be making requests to use the Control Bus, bus contention is generated. When all K's have completed their test, each K is assigned a different chunk of Control memory, and the test is repeated. When all K's have tested each chunk once, the Control Bus contention tests are complete.

2. Data Bus Contention - The available Data memory is divided into as many 'chunks' as there are available K's. For the initial test, the K with the lowest requestor number is assigned to test the chunk of memory with the lowest starting address. The K with the next highest requestor number is assigned to test the next chunk of memory, and so on until each K has been assigned a chunk of Data memory to test. The parameters for each K are stored in its Control Area, and the K's are enabled (released from the initialized state). When all K's have indicated that they have found their parameters and are ready to test, all K's are commanded to begin testing. Since all K's will now be making requests to use the Data Bus, bus contention is generated. When all K's have completed their test, each K is assigned a different chunk of Data memory, and the test is repeated. When all K's have tested each chunk once, the Data Bus contention tests are complete.

3. Combined Control and Data Bus Contention - These tests are similar to 1 and 2 above, but both the Control and Data memories are tested simultaneously. Half of the available K's test the Control memory at any given time, while the other half test the Data memory.

5.4 FAILURE ISOLATION TECHNIQUES

The following sections describe several methods that may be used to localize the cause of a test failure.

5.4.1 RUNNING THE K/P MEMORY TEST

When the Bus Interaction test fails, the user must first determine if the failure was caused by an interaction problem. This can be determined by running the Offline K/P Memory Test ('Test Memory By K'). When the test prompts for parameters, specify the requestor number of the K that detected the failure in the Bus Interaction Test. Also specify the same starting and ending addresses that were displayed with the error report from the Bus Interaction Test. If the K also fails the Offline K/P Memory Test, the original problem was not an interaction problem and should be localized in the same manner as any ordinary memory failure. (The Offline Bus Interaction Test assumes that Control and Data memory are working). If the K does not fail the Offline K/P Memory Test, the original failure was an interaction problem and should be localized as described in the following sections.

5.4.2 INSPECTING ERROR REPORTS

1. A particular K always fails, other K's work properly.

2. The K in the lowest requestor position tends to fail the most.

3. One section of memory always fails, no matter which K tests that section.

5.4.3 REMOVING K's FROM THE TEST

<TBS>

HSC50 Off-Line Microdiagnostic Test Selector (TEST K)

USER DOCUMENTATION

PRODUCT CODE: XX-XXXXX-XX

PRODUCT NAME: HSC50 Offline K Test Selector

PRODUCT DATE: 12-APR-1983

MAINTAINER: Diagnostic Engineering - Colorado (CX)

AUTHOR: Mike Hare

TABLE OF CONTENTS

SECTION 1 - GENERAL INFORMATION

1.1 PROGRAM ABSTRACT

The Offline K Test Selector allows a user to command a K to perform one of its internal microdiagnostic tests. The Offline K Test Selector executes from the P.ioc, and uses the HSC5Ø 'K Control Area' to instruct one of the sub-system K's to test itself. The operator selects the K to be used, and selects the test number of the microdiagnostic test to be executed. Errors are reported on the console terminal as they occur.

1.2 SYSTEM REQUIREMENTS

Hardware Required:

a) P.ioc (processor) module with HSC5Ø Boot ROMS.
b) At least one M.std (memory) module.
c) TU58 controller with at least one working drive.
d) Terminal connected to P.ioc's console interface.
e) At least one working 'K' (K.sdi, K.sti, or K.ci).
f) A working section of Control memory for use as a 'K Control Area'.

Software Required:

a) HSC5Ø Offline Diagnostic Tape

1.3 RELATED DOCUMENTS

a) HSC5Ø Offline Diagnostic Loader User Document
b) HSC5Ø Offline P.ioc Test User Document
c) HSC5Ø ROM Bootstrap User Document
d) HSC5Ø K Microdiagnostic User Document
e) HSC5Ø Offline System Sizer User Document
f) HSC5Ø Offline Bus Interaction Test User Document

1.4 HARDWARE ASSUMED TO BE WORKING

In the process of loading the Offline Diagnostic Loader, several diagnostics are run. The ROM Bootstrap tests the basic F-11 instruction set, tests a partition in Program Memory, and tests the TU58 used for the boot. Then the bootstrap loads the Offline P.ioc Test, which completes the F-11 tests and tests the remainder of the P.ioc module. After these tests, the Offline Diagnostic Loader is loaded from the TU58 to memory, and control is passed to the Loader. When the operator instructs the Loader to 'TEST K', the Offline K Test Selector is loaded and started. Due to the sequence of tests that precede this test, the P.ioc, Program memory, and TU58 are assumed to be tested and working.

SECTION 2 - OPERATING INSTRUCTIONS

2.1 START-UP PROCEDURE

Follow the steps below to start the K Test Selector:

1) Insert the HSC5Ø OFFLINE TESTS cassette into the TU58's Unit Ø drive. (left hand drive)

2) Power-on the HSC5Ø, or depress and release the INIT button on the HSC5Ø front panel.

3) The TU58 drive-motion LED should light within 1Ø seconds, indicating that the Bootstrap is loading the OFFLINE DIAGNOSTIC LOADER to Program Memory.

4) After about 6Ø seconds of loading, the OFFLINE DIAGNOSTIC LOADER will indicate that it has loaded properly by displaying the following:

```
     HSC5Ø OFL Diagnostic Loader, Version Vnn-nn
     Radix=Octal,Data Length=Word,Reloc=ØØØØØØØØ
 ODL>
```

5) Type 'TEST K' in response to the Loader prompt (ODL>). The TU58 drive-motion LED will light as the test is loaded.

6) The test will indicate it has been loaded properly by displaying the following:

```
     HSC5Ø OFL K Test Selector, Version Vnn-nn
```

7) The test now prompts for parameters. See section 2.2 below.

2.2 TEST PARAMETER ENTRY

The Offline K Test Selector first prompts:

```
K requestor # (1 thru 7) [] ?
```

Answer this question with single digit (1 thru 7), that specifies the requestor # of the K to be used. Terminate the response by typing a carriage-return. After the requestor # is supplied, a K-Control-Area is located in the Control memory and tested. This area is required for communicating with the K that will its microdiagnostics. The test then prompts:

```
Test # (1 thru 11) (O) [] ?
```

Legal test numbers are octal numbers between 1 and 11(8), except for 5. (Test 5 is the K's Control and Data memory test, which is supported by the OFL K/P Memory Test). Terminate the test # with a carriage-return. The test will then prompt:

```
# of passes to perform (D) [1] ?
```

Enter a decimal number between 1 and 2,147,483,647 (omit commas), which specifies the number of times the memory test should be repeated. (Entering a Ø, or just a carriage-return, will result in 1 pass being performed.)

The P.ioc now instructs the K to perform the selected test. The P.ioc allows up to 4.2 seconds for the K to complete its test. If the K completes the test within this time, the P.ioc will display an end of pass message. If the K fails to complete within 4.2 seconds, the P.ioc will display a K Time-Out Error (Error ØØ9). The K microdiagnostics are designed to 'hang' when an error is detected, so all failures in the microdiagnostics will be reported as time-out errors.

The current test may be aborted at any time by typing a control-c.

NOTE: For any of the above prompts, use the DELETE key to delete mis-typed parameters before the terminating carriage-return is typed. If you note an error in a parameter that has already been terminated with a carriage-return, type a control-c to return to the initial prompt and re-enter all parameters.

2.2.1 REUSING PARAMETERS

After the first test has been specified and completed, the following prompt will be issued:

```
Re-use parameters (Y/N) [Y] ?
```

Answering this prompt with a carriage-return or a 'Y' followed by a carriage-return, results in repeating the last test specified, using the parameters that were previously used. Answering the prompt with a 'N' followed by a carriage-return will cause the test to prompt for new parameters.

2.3 SETTING CLEARING FLAGS

N/A

2.4 PROGRESS REPORTS

Each time the K completes one full pass through the test specified, an end of pass report is displayed. A full pass is defined as:

1. The K completes the test with no errors detected.

2. The K fails its test, and the P.ioc times-out.

The end of pass message is displayed as follows:

```
End of Pass nnnnnn, xxxxxx Errors, yyyyyy Total Errors
```

The pass count 'nnnnnn' is a decimal count of the number of complete passes made. The error count (xxxxxx) indicates the number of errors detected during the current pass. The total error count (yyyyyy) indicates the number of errors detected during all passes completed so far.

SECTION 3 - ERROR INFORMATION

3.1 ERROR MESSAGE FORMAT

All error messages produced by the this test conform to the HSC5Ø Diagnostic Error message format. The first line of an error message contains general information concerning the error, and is mandatory. The second line of an error message consists of text describing the error, and is also mandatory. The third and succeeding lines of an error message are used to display additional information when required, and are optional. Listed below is a typical error message.

```
OKTS>hh:mm T#aaa E#bbb U-ØØØ
  <Text describing error>
  MA -xxxxxxxx
  EXP-yyyyyy
  ACT-zzzzzz
```

WHERE:

```
   hh       = Elapsed hours since last bootstrap
   mm       = Elapsed minutes
   aaa      = decimal number denoting test
   bbb      = decimal number denoting the error detected
   xxxxxxxx = Address of location causing the error
   yyyyyy   = Data that was expected
   zzzzzz   = Data that was actually found
```

3.1.1 K.CI PATH STATUS INFORMATION

Whenever a K.ci is enabled it will run the CI Link test as part of its microdiagnostics. The Link test performs loop-back tests on CI paths 'A' and 'B' of the K.ci. To pass the Link test, either path must work (only one failing path is not a fatal error). The microdiagnostics return information in the K Control Area which specifies which paths worked, and how many retries were required. (The test retries 64 times before declaring a failure).

The Offline K Test selector will report the CI Path status each time the K.ci is initialized. If the link test is selected (K.ci test 11), the path status will only be reported after the link test completes. (When the K.ci is enabled, it will run all of its microdiagnostics, including the link test, then if the link test was selected, the K.ci will run the link test once more).

The CI Path Status display indicates which path failed the link test, if any. If both paths fail, the microdiagnostics will fail in test 11, and no path status information will be displayed. The status display also includes the number of retries required for paths that passed the link test.

3.2 SPECIFIC ERROR MESSAGES

Errors detected by this test fall into one of three classes:

1. Control Memory errors detected by the P.ioc when testing the K Control Area.

2. Unexpected traps detected by the P.ioc (NXM and Parity).

3. Failures in a K microdiagnostic, detected by a time-out.

Control memory errors occur when the P.ioc is testing the portion of Control memory that will be used to communicate with the K. (The P.ioc does not test Data memory).

Error numbers ØØØ thru ØØ7 are all Control memory errors detected by the P.ioc. The difference among these errors being the exact step in the memory test where they are detected. The step where an error was detected can be a helpful clue to the cause of the error.

Error ØØ8 indicates that the K failed to initialize properly. Error ØØ9 indicates that the K failed the selected microdiagnostic.

Errors Ø1Ø and Ø11 are unexpected trap errors detected by the P.ioc. Error Ø1Ø signifies a parity trap occurred, and error Ø11 indicates a non-existent-memory trap. The reports for unexpected trap errors differ slightly from a data error report, since they do not display expected and actual data.

Error 12 indicates that no working control memory could be found for a K Control Area. Refer to section 3.2.13 for further information.

The following sections (3.2.x) describe the nature of the failure indicated by each error number.

3.2.1 ERROR ØØØ

Error ØØØ occurs in the Moving Inversions test (see section 4.1.1), when the P.ioc is testing the K Control Area. A memory location did not contain the expected pattern.

MA = The address of the failing location.
EXP= The data pattern expected.
ACT= The data pattern actually found.

This error can be caused by a DATA ERROR in the address specified, or it may indicate a DUAL-ADDRESSING problem (the location was incorrectly addressed and written when some other location was written.).

DUAL-ADDRESSING ERRORS - At this step in the test, a dual-addressing problem is characterized by:

a) The ACTual data contains a single additional 'one'.

b) The additional 'one' bit occurs immediately to the left of the leftmost 'one' in the EXPected data.

```
E.G.  EXP=ØØØ377, ACT=ØØØ777
      EXP=Ø77777, ACT=177777
      EXP=ØØØØØØ, ACT=ØØØØØ1
```

In the first example, the location in error was probably written with the pattern ØØØ777 when a lower numbered address was being written with the same pattern. When the location in error was subsequently checked to insure it still contained the 'previous' pattern (ØØØ377), it was found to contain the 'next' pattern (ØØØ777).

DATA ERRORS - Data errors at this step of the test fall into one of the following classes: 5.break

a) The ACTual and EXPected data differ by more than one bit.
E.G. EXP=Ø17777, ACT=Ø17477

b) The ACTual data contains less 'ones' than the EXPected data.
E.G. EXP=ØØ3777, ACT=ØØ1777

c) The bit in error is not in the bit position immediately to the left of the leftmost 'one' in the EXPected data.
E.G. EXP=ØØØ777, ACT=ØØ2777

3.2.2 ERROR ØØ1

Error ØØ1 occurs in the Moving Inversions Test (sec. 4.1,1). when the P.ioc is testing the K Control Area. A location was written with a pattern. Immediately after the write, the location was read and found to contain an incorrect pattern.

MA = The address of the failing location
EXP= The data pattern expected
ACT= The data pattern actually found

This error indicates a memory data problem. One of the following hardware failures is indicated:

a) A bit was picked up or dropped when the location was written.
b) A bit was picked up or dropped when the location was read.

If the error occurs repeatedly, but only in a single location, the memory chip containing the failing bit for that address is probably defective.

If the error occurs in many locations, but only occurs in a particular Nibble (4 bit field), one of the bus data tranceivers for that nibble is probably defective.

If the error occurs in many locations, and the bits in error are radomly spaced thru-out the word, the memory or bus timing is probably flakey.

If the error occurs in more than one location, but the addresses of the failing locations are similar, there could be crosstalk between the memory data and addressing lines. E.G. All failing addresses end with either 2 or 6.

3.2.3 ERROR ØØ2

Error ØØ2 occurs in the Moving Inversions test (section 4.1.1), when the P.ioc is testing the K Control Area. A memory location did not contain the expected pattern.

MA = The address of the location in error.
EXP= The data pattern expected.
ACT= The data pattern actually found

This error can be caused by a DATA ERROR in the address specified, or it may indicate a DUAL ADDRESSING problem (The location was incorrectly addressed and written when some other location was being written.).

DUAL ADDRESSING ERRORS - At this step in the test, a dual-addressing problem is characterized by:

a) The ACTual data contains one more 'zero' than the EXPected data.

b) The additional zero occurs in the same bit position as the leftmost 'one' in the EXPected data.

```
E.G.   EXP=ØØ3777, ACT=ØØ1777
       EXP=ØØØØ17, ACT=ØØØØØ7
       EXP=177777, ACT=Ø77777
```

In the first example, the location in error was probably written with the pattern ØØ1777 when a lower numbered address was being written with the same pattern. When the location in error was subsequently checked to insure it still contained the 'previous' pattern (ØØ3777), it was found to contain the 'next' pattern (ØØ1777).

DATA ERRORS - Data errors in this step of the Moving Inversions test fall into one of the following categories:

a) The ACTual and EXPected data differ by more than one bit.
   E.G. EXP=177777, ACT=174777

b) The ACTual data contains more 'ones' than the EXPected data.
   E.G. EXP=Ø37777, ACT=Ø77777

c) The bit in error is not in the same bit position as the leftmost 'one' in the EXPected data.
   E.G. EXP=ØØ1777, ACT=ØØ1377

3.2.4 ERROR 003

Error 003 occurs in the Moving Inversions Test (sec. 4.1.1), when the P.ioc is testing the K Control Area. A location was written with a pattern. Immediately after the write, the location was read and found to contain an incorrect pattern.

MA = The address of the failing location
EXP= The data pattern expected
ACT= The data pattern actually found

This error indicates a memory data problem. One of the following hardware failures is indicated:

a) A bit was picked up or dropped when the location was written.
b) A bit was picked up or dropped when the location was read.

If the error occurs repeatedly, but only in a single location, the memory chip containing the failing bit for that address is probably defective.

If the error occurs in many locations, but only occurs in a particular Nibble (4 bit field), one of the bus data tranceivers for that nibble is probably defective.

If the error occurs in many locations, and the bits in error are radomly spaced thru-out the word, the memory or bus timing is probably flakey.

If the error occurs in more than one location, but the addresses of the failing locations are similar, there could be crosstalk between the memory data and addressing lines. E.G. All failing addresses end with either 2 or 6.

3.2.5 ERROR ØØ4

Error ØØ4 occurs in the Moving Inversions test (see section 4.1.1), when the P.ioc is testing the K Control Area. A memory location did not contain the expected pattern.

```
MA = The address of the failing location.
EXP= The data pattern expected.
ACT= The data pattern actually found.
```

This error can be caused by a DATA ERROR in the address specified, or it may indicate a DUAL-ADDRESSING problem (the location was incorrectly addressed and written when some other location was written.).

DUAL-ADDRESSING ERRORS - At this step in the test, a dual-addressing problem is characterized by:

a) The ACTual data contains a single additional 'one'.

b) The additional 'one' bit occurs immediately to the left of the leftmost 'one' in the EXPected data.

```
E.G.   EXP=ØØØ377, ACT=ØØØ777
       EXP=Ø77777, ACT=177777
       EXP=ØØØØØØ, ACT=ØØØØØ1
```

In the first example, the location in error was probably written with the pattern ØØØ777 when a higher numbered address was being written with the same pattern. When the location in error was subsequently checked to insure it still contained the 'previous' pattern (ØØØ377), it was found to contain the 'next' pattern (ØØØ777).

DATA ERRORS - Data errors at this step of the test fall into one of the following classes:

a) The ACTual and EXPected data differ by more than one bit.
   E.G. EXP=Ø17777, ACT=Ø17477

b) The ACTual data contains less 'ones' than the EXPected data.
   E.G. EXP=ØØ3777, ACT=ØØ1777

c) The bit in error is not in the bit position immediately to the left of the leftmost 'one' in the EXPected data.
   E.G. EXP=ØØØ777, ACT=ØØ2777

3.2.6 ERROR ØØ5

Error ØØ5 occurs in the Moving Inversions Test (sec. 4.1.1), when the P.ioc is testing the K Control Area. A location was written with a pattern. Immediately after the write, the location was read and found to contain an incorrect pattern.

```
MA = The address of the failing location
EXP= The data pattern expected
ACT= The data pattern actually found
```

This error indicates a memory data problem. One of the following hardware failures is indicated:

a) A bit was picked up or dropped when the location was written.
b) A bit was picked up or dropped when the location was read.

If the error occurs repeatedly, but only in a single location, the memory chip containing the failing bit for that address is probably defective.

If the error occurs in many locations, but only occurs in a particular Nibble (4 bit field), one of the bus data tranceivers for that nibble is probably defective.

If the error occurs in many locations, and the bits in error are radomly spaced thru-out the word, the memory or bus timing is probably flakey.

If the error occurs in more than one location, but the addresses of the failing locations are similar, there could be crosstalk between the memory data and addressing lines. E.G. All failing addresses end with either 2 or 6.

3.2.7 ERROR ØØ6

Error ØØ6 occurs in the Moving Inversions test (section 4.1.1), when the P.ioc is testing the K Control Area. A memory location did not contain the expected pattern.

```
MA = The address of the location in error.
EXP= The data pattern expected.
ACT= The data pattern actually found
```

This error can be caused by a DATA ERROR in the address specified, or it may indicate a DUAL ADDRESSING problem (The location was incorrectly addressed and written when some other location was being written.).

DUAL ADDRESSING ERRORS - At this step in the test, a dual-addressing problem is characterized by:

a) The ACTual data contains one more 'zero' than the EXPected data.

b) The additional zero occurs in the same bit position as the leftmost 'one' in the EXPected data.

```
E.G.   EXP=ØØ3777, ACT=ØØ1777
       EXP=ØØØØ17, ACT=ØØØØØ7
       EXP=177777, ACT=Ø77777
```

In the first example, the location in error was probably written with the pattern ØØ1777 when a higher numbered address was being written with the same pattern. When the location in error was subsequently checked to insure it still contained the 'previous' pattern (ØØ3777), it was found to contain the 'next' pattern (ØØ1777).

DATA ERRORS - Data errors in this step of the Moving Inversions test fall into one of the following categories:

a) The ACTual and EXPected data differ by more than one bit.
   E.G. EXP=177777, ACT=174777

b) The ACTual data contains more 'ones' than the EXPected data.
   E.G. EXP=Ø37777, ACT=Ø77777

c) The bit in error is not in the same bit position as the leftmost 'one' in the EXPected data.
   E.G. EXP=ØØ1777, ACT=ØØ1377

3.2.8 ERROR 007

Error 007 occurs in the Moving Inversions Test (sec. 4.1.1), when the P.ioc is testing the K Control Area. A location was written with a pattern. Immediately after the write, the location was read and found to contain an incorrect pattern.

```
MA = The address of the failing location
EXP= The data pattern expected
ACT= The data pattern actually found
```

This error indicates a memory data problem. One of the following hardware failures is indicated:

a) A bit was picked up or dropped when the location was written.
b) A bit was picked up or dropped when the location was read.

If the error occurs repeatedly, but only in a single location, the memory chip containing the failing bit for that address is probably defective.

If the error occurs in many locations, but only occurs in a particular Nibble (4 bit field), one of the bus data tranceivers for that nibble is probably defective.

If the error occurs in many locations, and the bits in error are radomly spaced thru-out the word, the memory or bus timing is probably flakey.

If the error occurs in more than one location, but the addresses of the failing locations are similar, there could be crosstalk between the memory data and addressing lines. E.G. All failing addresses end with either 2 or 6.

3.2.9 ERROR ØØ8

Error ØØ8 indicates that the selected K did not complete its Init sequence properly. When the P.ioc enables the K to perform a test, the K begins its Init sequence (which includes executing certain microdiagnostics). At the end of the K's Init sequence, the K indicates that it found the K-Control-Area by complementing a pointer word in the Control memory. If the K fails to complement this pointer word within 4.2 seconds of being enabled, an error ØØ8 is reported.

The contents of the K Status Register are displayed with the error report. Refer to the appendices at the end of this document for information on K status values.

If this error occurs, make sure that the 'requestor Number' parameter given matches the actual requestor number of the K.

3.2.10 ERROR ØØ9

ERROR ØØ9 indicates that the K failed the selected microdiagnostic test. This usually indicates a serious hardware problem in the K. The contents of the K Status Register are displayed with the error report. Refer to the appendices at the end of this document for further information on the test that failed, and the meaning of the K Status value. There is one appendix for each of the different types of K's (K.sdi, K.sti, K.ci).

3.2.11 ERROR Ø1Ø

ERROR Ø1Ø indicates the P.ioc detected a Parity trap. The 22-bit address of the location that caused the trap is displayed as the 'MA' data in the error report.

```
  MA = The address causing the parity trap.
 VPC = The Virtual PC of the memory test at the time the
       trap occurred.  Reference this address in the
       listing to locate the area of the test where
       the error occurred.
```

Since the data is lost when a parity trap occurs, no EXPected or ACTual data is displayed. After the trap is reported, the program will attempt to restart the test from the beginning.

3.2.12 ERROR Ø11

Error Ø11 indicates the P.ioc detected a Non-Existent-Memory trap. A NXM error is caused when no memory responds to a particular address. The 'MA' data in the error report indicates the address which produced the NXM trap. After reporting the trap, the program attempts to restart the test from the beginning.

```
  MA = The address causing the NXM trap.
 VPC = The virtual PC of the memory test at the time the
       trap occurred.  Reference this address in the
       listing to locate the area of the test where
       the error occurred.
```

If this error occurs at a memory address that should be in your memory configuration, the memory in question is not supplying an 'ACK' to the P.ioc when the specified address is presented on the memory bus. The most probable point of failure is the logic on the memory module that compares addresses on the memory bus to the range of addresses that the module should respond to. The comparitor itself cound be faultly, or the [C IN, C OUT], [D IN, D OUT], or [P IN, P OUT] lines on the backplane could be in error.

3.2.13 ERROR Ø12

Error 12 indicates that no working control memory could be found for a K Control Area. A K Control Area is required to communicate with a K. The control memory must be repaired before the K Test Selector can be used to test a K. Use the Offline Loader's "Test Memory" command to test the control memory.

SECTION 4 - TEST SUMMARIES

4.1 TEST SUMMARIES

4.1.1 TEST ØØØ - MOVING INVERSIONS TEST (FROM P.IOC)

Test ØØØ is the Moving Inversions (MOVI) memory test used by the P.ioc to test a K Control Area. The K Control Area is used to pass memory test parameters to the K, and to return the results of memory tests to the P.ioc. The Moving Inversions RAM test is used to detect data and addressing problems in dynamic semiconductor memories.

MOVING INVERSIONS ALGORITHM

1) Write ØØØØØØ in each location being tested.

2) Read all locations in order from lowest to highest. After reading a location and checking for a zero, rewrite the same location with a single one in the least significant bit. Then reread the location and verify that the write worked correctly.

3) Again read all locations in order from lowest to highest, checking each location to contain the data previously written, and rewriting the data found with a single additional 'one' bit, and rereading to check that the write worked properly.

4) Repeat step 3 until the test pattern consists of a word containing all 'ones' (pattern 177777).

5) Repeat step 3, but this time substitute a single extra zero each time, instead of a one.

6) Continue step 5 until the test pattern consists of a word of all zeroes. (pattern ØØØØØØ)

7) Repeat steps 1 thru 6, but this time start at the highest memory address each time and work down to the lowest. This will work each memory location from all zeroes to all ones, and back to all zeroes.

8) End of test. All memory is cleared to ØØØØØØ.

4.1.2 TEST ØØ1 thru TEST Ø11 (K MICRODIAGNOSTICS)

Test ØØ1 thru Test Ø11(8) are the K's microdiagnostic tests. Refer to one of the appendices of this document, depending on the type of K being used (K.ci, K.sdi, K.sti), and the test number that failed.

SECTION 5 - APPENDICES

5.1 K.CI MICRODIAGNOSTICS

The following list shows the test number and name of each of the K.ci microdiagnostics.

1. Test 0 - Sequencer Test.
2. Test 1 - ALU Test.
3. Test 2 - Data Bus Test.
4. Test 3 - Control Bus Test.
5. Test 4 - PROM Parity Test.
6. Test 5 - Memory Test (Not available via K Test Selector).
7. Test 6 - RAM test.
8. Test 7 - PLI Interface Test.
9. Test 10 - Packet Buffer Test.
10. Test 11 - Link Test.

5.2 K.SDI MICRODIAGNOSTICS

The following list shows the test number and name of each of the K.sdi microdiagnostics.

1. Test 0 - Sequencer Test.
2. Test 1 - ALU Test.
3. Test 2 - Data Bus Test.
4. Test 3 - Control Bus Test.
5. Test 4 - PROM Parity Test.
6. Test 5 - Memory Test (Not available via K Test Selector).
7. Test 6 - RAM test.
8. Test 7 - SERDES/RSGEN Test.
9. Test 10 - Partial SDI Interface Test.

5.3 K.STI MICRODIAGNOSTICS

The following list shows the test number and name of each of the K.sti microdiagnostics.

1. Test 0 - Sequencer Test.
2. Test 1 - ALU Test.
3. Test 2 - Data Bus Test.
4. Test 3 - Control Bus Test.
5. Test 4 - PROM Parity Test.
6. Test 5 - Memory Test (Not available via K Test Selector).
7. Test 6 - RAM test.
8. Test 7 - SERDES Test.
9. Test 10 - Partial STI Interface Test.

5.4 K STATUS BYTE INTERPRETATION

The meaning of K Status is defined as follows:

1. Bit 7 of the status byte is the parity bit. The overall parity (count of 1's) of the status byte should be odd. If the parity of bits 0 thru 6 is even, then bit 7 should be set to make the overall parity odd. If the parity of bits 0 thru 6 is odd, then bit 7 should be clear. If the parity of the K S atus byte is even, then the meaning of the K status can not be determined.
2. Bit 6 of the status byte is set to indicate that the status value represents a microdiagnostic failure. Bit 6 is clear when the status value represents a K Type Code. The K Type Code identifies the type of K. See section 5.5.
3. Bits 0 thru 5 have one of two meanings:
    1. If bit 6 is set, then bits 0 thru 5 represent the test number of a microdiagnostic which failed.
    2. If bit 6 is clear, then bits 0 thru 5 represent the K Type Code. See section 5.5.

5.5 K TYPE CODES

The following K Type Codes are currently defined:

1. Code ØØ1 represents a K.CI (Host Port). The K status byte will be ØØ1(8).

2. Code ØØ2 represents a K.SDI (Disk Data Channel). The K status byte will be ØØ2(8).

3. Code ØØ3 represents a K.STI (Tape Data Channel). The K status byte will be 2Ø3(8). (The status byte includes the parity bit to make the overall parity odd).

Other type codes will be defined as new K's are developed.

HSC50 Off-Line K/P Memory Test (TEST MEM BY K)

USER DOCUMENTATION

PRODUCT CODE:	XX-XXXXX-XX
PRODUCT NAME:	HSC50 Offline K/P Memory Test
PRODUCT DATE:	12-APR-1983
MAINTAINER:	Diagnostic Engineering - Colorado (CX)
AUTHOR:	Mike Hare

TABLE OF CONTENTS

SECTION 1 - GENERAL INFORMATION

1.1 PROGRAM ABSTRACT

The Offline K/P Memory Test allows the HSC50 Control and Data memories to be tested from a 'K' (K.sdi, K.ci, K.sti). The Offline K/P Memory Test executes from the P.ioc, and uses the HSC50 'K Control Area' to instruct one of the sub-system 'K's to test either the Control or Data memories. The operator selects the 'K' to be used, and the starting and ending addresses of the section of memory to be tested. The test algorithm used by the 'K' is designed to stress the memories in an effort to detect transient errors caused by bus and memory timing problems. Errors are reported on the console terminal as they occur.

1.2 SYSTEM REQUIREMENTS

Hardware Required:

a) P.ioc (processor) module with HSC50 Boot ROMS.
b) At least one M.std (memory) module.
c) TU58 controller with at least one working drive.
d) Terminal connected to P.ioc's console interface.
e) At least one working 'K' (K.sdi, K.sti, or K.ci).
f) Working Control Memory for a K Control Area.

Software Required:

a) HSC50 Offline Diagnostic Tape

1.3 RELATED DOCUMENTS

a) HSC50 Offline Diagnostic Loader User Document
b) HSC50 Offline P.ioc Test User Document
c) HSC50 ROM Bootstrap User Document
d) HSC50 Offline K Test Selector User Document
e) HSC50 K Microdiagnostic User Document
f) HSC50 Offline Bus Interaction Test User Document

1.4 HARDWARE ASSUMED TO BE WORKING

In the process of loading the Offline Diagnostic Loader, several diagnostics are run. The ROM Bootstrap tests the basic F-11 instruction set, tests a partition in Program Memory, and tests the TU58 used for the boot. Then the bootstrap loads the Offline P.ioc Test, which completes the F-11 tests and tests the remainder of the P.ioc module. After these tests, the Offline Diagnostic Loader is loaded from the TU58 to memory, and control is passed to the Loader. When the operator instructs the Loader to 'TEST MEMORY BY K', the Offline K/P Memory test is loaded and started. Due to the sequence of tests that precede the memory test, the memory test assumes that the P.ioc module and the TU58 are tested and working.

SECTION 2 - OPERATING INSTRUCTIONS

2.1 START-UP PROCEDURE

Follow the steps below to start the memory test:

1) Insert the HSC5Ø OFFLINE TESTS cassette into the TU58's Unit Ø drive. (left hand drive)

2) Power-on the HSC5Ø, or depress and release the INIT button on the HSC5Ø front panel.

3) The TU58 drive-motion LED should light within 1Ø seconds, indicating that the Bootstrap is loading the OFFLINE DIAGNOSTIC LOADER to Program Memory.

4) After about 6Ø seconds of loading, the OFFLINE DIAGNOSTIC LOADER will indicate that it has loaded properly by displaying the following:

```
        HSC5Ø OFL Diagnostic Loader, Version Vnn-nn
        Radix=Octal,Data Length=Word,Reloc=ØØØØØØØØ
  ODL>
```

5) Type TEST MEMORY BY K in response to the Loader prompt (ODL>). The TU58 drive-motion LED will light as the memory test is loaded.

6) The memory test will indicate it has been loaded properly by displaying the following:

```
        HSC5Ø OFL K/P Memory Test, Version Vnn-nn
```

7) The memory test now prompts for parameters. See section 2.2 below.

2.2 TEST PARAMETER ENTRY

The Offline K/P Memory Test first prompts:

```
K requestor # (1 thru 7) [] ?
```

Answer this question with single digit (1 thru 7), that specifies the requestor # of the K to be used. Terminate the response by typing a carriage-return. After the requestor # is supplied, a K-Control-Area is located in the Control memory and tested. This area is required for communicating with the K that will perform tests of Data and Control memory. The test then prompts:

```
Control (0) or Data (1) memory [0] ?
```

Type a 0 to test Control Memory, or type a 1 to test Data Memory. After typing the desired digit, type a carriage-return to terminate your response. (Typing just a carriage-return results in selecting the Control memory test.) The memory test next prompts for the first address to test:

```
First (min=XXXXXXXX) [min] ?
```

Enter the first address to be tested. Addresses are 8 octal digits in length. The 'min' address displayed is the lowest address that may be entered for the memory chosen. After typing the address, terminate your response with a carriage-return. (Typing just a carriage-return will cause the first address to default to the 'min' address.)

NOTES:

1) Control memory addresses - Because K's test control memory in 4-byte units, the lowest 2 bits of the starting address are ignored (treated as binary zeroes). E.G. If address 16000203 is entered as the first address, the K will start testing at address 16000200.

2) Data memory addresses - Because K's test data memory in 64-byte units, the lower 6 bits of the starting address are ignored (treated as binary zeroes). E.G. If address 14012376 is entered as the first address, the K will start testing at address 14012300.

The test now prompts for the last address to test:

```
     Last (max=XXXXXXXX) [] ?
```

Enter the last address to be tested. The 'max' address displayed is the highest address that is still within the memory chosen. If your system does not have a fully populated memory, the last address that may be tested will be less than the 'max' address displayed. If you choose a last address that exceeds the amount of memory in your system, the memory test will display a 'Non-Existent-Memory' (NXM) error when the test reaches the first address beyond the end of your memory. (Use the Offline Loader's SIZE command to determine the actual last address in a given HSC50.)

NOTES:

1) Control memory addresses - Because K's test control memory in 4-byte units, the lower 2 bits of the ending address are ignored (treated as binary ones). E.G. If address 16023400 is specified as the last address, the K will test up to, and including, address 16023403.

2) Data memory addresses - Because K's test data memory in 64-byte units, the lower 6 bits of the ending address are ignored (treated as binary ones). E.G. If address 14005400 is specified as the last address, the K will test up to, and including, address 14005477.

Finally, the memory test prompts:

```
     # of passes to perform (D) [1] ?
```

Enter a decimal number between 1 and 2,147,483,647 (omit commas), which specifies the number of times the memory test should be repeated. (Entering a 0, or just a carriage-return, will result in 1 pass being performed.) The memory test will now begin. The memory test can be aborted at any time by typing a control-c.

NOTE: For any of the above prompts, use the DELETE key to delete mis-typed parameters before the terminating carriage-return is typed. If you note an error in a parameter that has already been terminated with a carriage-return, type a control-c to return to the initial prompt and re-enter all parameters.

2.2.1 REUSING PARAMETERS

After the first memory test has been specified and completed, the following prompt will be issued:

```
Re-use parameters (Y/N) [Y] ?
```

Answering this prompt with a carriage-return, or a 'Y' followed by a carriage-return, results in repeating the last test specified, using the parameters that were previously used. Answering the prompt with a 'N' followed by a carriage-return will cause the test to prompt for new parameters.

2.3 SETTING CLEARING FLAGS

N/A

2.4 PROGRESS REPORTS

Each time the K completes one full pass through the memory specified, an end of pass report is displayed. A full pass is defined as:

1. A complete test of the memory specified with no errors detected.

2. Testing the memory specified until an error occurs

The end of pass message is displayed as follows:

```
End of Pass nnnnnn, xxxxxx Errors, yyyyyy Total Errors
```

The pass count 'nnnnnn' is a decimal count of the number of complete passes made. The Error count (xxxxxx) indicates the number of errors detected on the current pass. The total error count (yyyyyy) indicates the number of errors detected during the passes completed so far.

2.5 DISABLING P.IOC PARITY ERRORS

When a parity error occurs, it is desirable to know whether the error was produced by the loss or gain of a data bit, or by the loss or gain of a parity bit. When a parity trap occurs in the P.ioc, the data that was read is discarded by the PDP-11. However, a feature of the P.ioc allows parity traps to be disabled. Using this feature, a user can determine if a parity error is being caused by a data or parity bit as follows:

1. After a (P.ioc detected) parity trap is reported, type a control-c to terminate the memory test.

2. Type another control-c to return to the OFL Diagnostic Loader. The Loader will prompt: "ODL>".

3. Type "Ex 1777ØØ42" followed by a carriage-return.

4. The contents of the P.ioc "Switch Control and Status Register" (SWCSR) will be displayed as follows: "(I) 1777ØØ42 nnnnnn".

5. Type "De * nnnn4n" followed by a carriage-return. The 'nnnn4n' represents the previous contents of the register, including a 'one' in bit 5.

6. Return to the memory test by typing "Start" followed by a carriage-return.

7. Rerun the memory test with the original parameters.

8. If the location that previously produced a parity trap now produces a data error, the original parity trap was caused by a data bit problem. The error report will indicate the failing bit via the EXPected and ACTual data displayed.

9. If the location that previously produced a parity trap does not fail again when the memory test is rerun, the original parity trap was caused by an error in one of the parity bits (high or low byte) for that word.

10. Type a control-c to return to the Loader, and re-enable parity errors by typing "De 1777ØØ42 nnnnØn" followed by a carriage-return. The 'nnnnØn' represents original contents of the P.ioc's SWCSR, before parity traps were disabled. (Refer to step 5 above).

SECTION 3 - ERROR INFORMATION

3.1 ERROR MESSAGE FORMAT

All error messages produced by the memory test conform to the HSC5Ø Diagnostic Error message format. The first line of an error message contains general information concerning the error, and is mandatory. The second line of an error message consists of text describing the error, and is also mandatory. The third and succeeding lines of an error message are used to display additional information when required, and are optional. Listed below is a typical error message.

```
OKPM>hh:mm T#aaa E#bbb U-ØØØ
  6<Text describing error>
  MA -xxxxxxxx
  EXP-yyyyyy
  ACT-zzzzzz
  <K-Error-Summary-Info>
```

WHERE:

```
   hh       = Elapsed hours since last bootstrap
   mm       = Elapsed minutes
   aaa      = decimal number denoting test
   bbb      = decimal number denoting the error detected
   xxxxxxxx = Address of location causing the error
   yyyyyy   = Data that was expected
   zzzzzz   = Data that was actually found
   <K-Error-Summary-Info>   =   See next section.
```

3.1.1 K ERROR SUMMARY INFORMATION

When the K reports a memory test failure to the P.ioc, the following information is supplied:

1. The address of the failing memory location.

2. The data expected and data actually found.

3. Error summary information.

The error summary information is supplied as a 3-bit field, which includes the following:

1. A bit indicating a Parity error occurred while reading the location.

2. A bit indicating a NXM error occurred while accessing the location.

3. A bit indicating Control Bus (CBUS) error occurred while accessing the location.

When a memory error report is issued for an error detected by the K, the last line of the error report will include a list of the error summary bits that were set (if any).

A Control Bus (CBUS) Error indicates that the K asserted an illegal combination of the three "CCYCLE" lines when accessing control memory. Since these lines were previously tested from the P.ioc (in the OFL P.IOC TEST), a Control Bus Error is most likely to be caused by a problem with the K's drivers that assert the "CCYCLE" lines.

3.2 SPECIFIC ERROR MESSAGES

Error messages produced by this test can be caused by a memory error detected either by the P.ioc, or by the K being used to test memory. Errors detected by the P.ioc occur when the P.ioc is testing the portion of Control memory that will be used to communicate with the K. (The P.ioc does not test Data memory). To determine whether the P.ioc or the K detected an error, examine the second line of the error message. The text either begins with a "(P)" or a "(K)". If the text begins with a "(P)", the P.ioc detected the error. If the text begins with a "(K)", the K detected the error.

Error numbers 000 thru 007 are all Control memory errors detected by the P.ioc. The difference among these errors being the exact step in the memory test where they are detected. The step where an error was detected can be a helpful clue to the cause of the error.

Error 008 indicates that the K failed to initialize properly.

Error 009 indicates a Control or Data memory error detected by the K. In addition to the normal error information, the last line of the error report will contain a K Error Summary (see previous section).

Errors 010 and 011 are unexpected trap errors detected by the P.ioc. Error 010 signifies a parity trap occurred, and error 011 indicates a non-existent-memory trap. The reports for unexpected trap errors differ slightly from a data error report, since they do not display expected and actual data.

Error 12 indicates that no working control memory could be found for a K Control Area. Refer to section 3.2.13 for more information.

The following sections (3.2.x) describe the nature of the failure indicated by each error number.

3.2.1 ERROR ØØØ

Error ØØØ occurs in the Moving Inversions test (see section 4.1.1), when the P.ioc is testing the K Control Area. A memory location did not contain the expected pattern.

MA = The address of the failing location.
EXP= The data pattern expected.
ACT= The data pattern actually found.

This error can be caused by a DATA ERROR in the address specified, or it may indicate a DUAL-ADDRESSING problem (the location was incorrectly addressed and written when some other location was written.).

DUAL-ADDRESSING ERRORS - At this step in the test, a dual-addressing problem is characterized by:

a) The ACTual data contains a single additional 'one'.

b) The additional 'one' bit occurs immediately to the left of the leftmost 'one' in the EXPected data.

```
E.G.  EXP=ØØØ377, ACT=ØØØ777
      EXP=Ø77777, ACT=177777
      EXP=ØØØØØØ, ACT=ØØØØØ1
```

In the first example, the location in error was probably written with the pattern ØØØ777 when a lower numbered address was being written with the same pattern. When the location in error was subsequently checked to insure it still contained the 'previous' pattern (ØØØ377), it was found to contain the 'next' pattern (ØØØ777).

DATA ERRORS - Data errors at this step of the test fall into one of the following classes:

a) The ACTual and EXPected data differ by more than one bit.
E.G. EXP=Ø17777, ACT=Ø17477

b) The ACTual data contains less 'ones' than the EXPected data.
E.G. EXP=ØØ3777, ACT=ØØ1777

c) The bit in error is not in the bit position immediately to the left of the leftmost 'one' in the EXPected data.
E.G. EXP=ØØØ777, ACT=ØØ2777

3.2.2 ERROR ØØ1

Error ØØ1 occurs in the Moving Inversions Test (sec. 4.1,1). when the P.ioc is testing the K Control Area. A location was written with a pattern. Immediately after the write, the location was read and found to contain an incorrect pattern.

MA = The address of the failing location
EXP= The data pattern expected
ACT= The data pattern actually found

This error indicates a memory data problem. One of the following hardware failures is indicated:

a) A bit was picked up or dropped when the location was written.
b) A bit was picked up or dropped when the location was read.

If the error occurs repeatedly, but only in a single location, the memory chip containing the failing bit for that address is probably defective.

If the error occurs in many locations, but only occurs in a particular Nibble (4 bit field), one of the bus data tranceivers for that nibble is probably defective.

If the error occurs in many locations, and the bits in error are radomly spaced thru-out the word, the memory or bus timing is probably flakey.

If the error occurs in more than one location, but the addresses of the failing locations are similar, there could be crosstalk between the memory data and addressing lines. E.G. All failing addresses end with either 2 or 6.

3.2.3 ERROR 002

Error 002 occurs in the Moving Inversions test (section 4.1.1), when the P.ioc is testing the K Control Area. A memory location did not contain the expected pattern.

```
MA = The address of the location in error.
EXP= The data pattern expected.
ACT= The data pattern actually found
```

This error can be caused by a DATA ERROR in the address specified, or it may indicate a DUAL ADDRESSING problem (The location was incorrectly addressed and written when some other location was being written.).

DUAL ADDRESSING ERRORS - At this step in the test, a dual-addressing problem is characterized by:

a) The ACTual data contains one more 'zero' than the EXPected data.

b) The additional zero occurs in the same bit position as the leftmost 'one' in the EXPected data.

```
E.G.  EXP=003777, ACT=001777
      EXP=000017, ACT=000007
      EXP=177777, ACT=077777
```

In the first example, the location in error was probably written with the pattern 001777 when a lower numbered address was being written with the same pattern. When the location in error was subsequently checked to insure it still contained the 'previous' pattern (003777), it was found to contain the 'next' pattern (001777).

DATA ERRORS - Data errors in this step of the Moving Inversions test fall into one of the following categories:

a) The ACTual and EXPected data differ by more than one bit.
   E.G. EXP=177777, ACT=174777

b) The ACTual data contains more 'ones' than the EXPected data.
   E.G. EXP=037777, ACT=077777

c) The bit in error is not in the same bit position as the leftmost 'one' in the EXPected data.
   E.G. EXP=001777, ACT=001377

3.2.4 ERROR ØØ3

Error ØØ3 occurs in the Moving Inversions Test (sec. 4.1.1), when the P.ioc is testing the K Control Area. A location was written with a pattern. Immediately after the write, the location was read and found to contain an incorrect pattern.

MA = The address of the failing location
EXP= The data pattern expected
ACT= The data pattern actually found

This error indicates a memory data problem. One of the following hardware failures is indicated:

a) A bit was picked up or dropped when the location was written.
b) A bit was picked up or dropped when the location was read.

If the error occurs repeatedly, but only in a single location, the memory chip containing the failing bit for that address is probably defective.

If the error occurs in many locations, but only occurs in a particular Nibble (4 bit field), one of the bus data tranceivers for that nibble is probably defective.

If the error occurs in many locations, and the bits in error are radomly spaced thru-out the word, the memory or bus timing is probably flakey.

If the error occurs in more than one location, but the addresses of the failing locations are similar, there could be crosstalk between the memory data and addressing lines. E.G. All failing addresses end with either 2 or 6.

3.2.5 ERROR 004

Error 004 occurs in the Moving Inversions test (see section 4.1.1), when the P.ioc is testing the K Control Area. A memory location did not contain the expected pattern.

```
MA = The address of the failing location.
EXP= The data pattern expected.
ACT= The data pattern actually found.
```

This error can be caused by a DATA ERROR in the address specified, or it may indicate a DUAL-ADDRESSING problem (the location was incorrectly addressed and written when some other location was written.).

DUAL-ADDRESSING ERRORS - At this step in the test, a dual-addressing problem is characterized by:

a) The ACTual data contains a single additional 'one'.

b) The additional 'one' bit occurs immediately to the left of the leftmost 'one' in the EXPected data.

```
E.G.  EXP=000377, ACT=000777
      EXP=077777, ACT=177777
      EXP=000000, ACT=000001
```

In the first example, the location in error was probably written with the pattern 000777 when a higher numbered address was being written with the same pattern. When the location in error was subsequently checked to insure it still contained the 'previous' pattern (000377), it was found to contain the 'next' pattern (000777).

DATA ERRORS - Data errors at this step of the test fall into one of the following classes:

a) The ACTual and EXPected data differ by more than one bit.
   E.G. EXP=017777, ACT=017477

b) The ACTual data contains less 'ones' than the EXPected data.
   E.G. EXP=003777, ACT=001777

c) The bit in error is not in the bit position immediately to the left of the leftmost 'one' in the EXPected data.
   E.G. EXP=000777, ACT=002777

3.2.6 ERROR 005

Error 005 occurs in the Moving Inversions Test (sec. 4.1.1), when the P.ioc is testing the K Control Area. A location was written with a pattern. Immediately after the write, the location was read and found to contain an incorrect pattern.

```
MA = The address of the failing location
EXP= The data pattern expected
ACT= The data pattern actually found
```

This error indicates a memory data problem. One of the following hardware failures is indicated:

a) A bit was picked up or dropped when the location was written.
b) A bit was picked up or dropped when the location was read.

If the error occurs repeatedly, but only in a single location, the memory chip containing the failing bit for that address is probably defective.

If the error occurs in many locations, but only occurs in a particular Nibble (4 bit field), one of the bus data tranceivers for that nibble is probably defective.

If the error occurs in many locations, and the bits in error are radomly spaced thru-out the word, the memory or bus timing is probably flakey.

If the error occurs in more than one location, but the addresses of the failing locations are similar, there could be crosstalk between the memory data and addressing lines. E.G. All failing addresses end with either 2 or 6.

3.2.7 ERROR 006

Error 006 occurs in the Moving Inversions test (section 4.1.1), when the P.ioc is testing the K Control Area. A memory location did not contain the expected pattern.

```
MA = The address of the location in error.
EXP= The data pattern expected.
ACT= The data pattern actually found
```

This error can be caused by a DATA ERROR in the address specified, or it may indicate a DUAL ADDRESSING problem (The location was incorrectly addressed and written when some other location was being written.).

DUAL ADDRESSING ERRORS - At this step in the test, a dual-addressing problem is characterized by:

a) The ACTual data contains one more 'zero' than the EXPected data.

b) The additional zero occurs in the same bit position as the leftmost 'one' in the EXPected data.

```
E.G.  EXP=003777, ACT=001777
      EXP=000017, ACT=000007
      EXP=177777, ACT=077777
```

In the first example, the location in error was probably written with the pattern 001777 when a higher numbered address was being written with the same pattern. When the location in error was subsequently checked to insure it still contained the 'previous' pattern (003777), it was found to contain the 'next' pattern (001777).

DATA ERRORS - Data errors in this step of the Moving Inversions test fall into one of the following categories:

a) The ACTual and EXPected data differ by more than one bit.
   E.G. EXP=177777, ACT=174777

b) The ACTual data contains more 'ones' than the EXPected data.
   E.G. EXP=037777, ACT=077777

c) The bit in error is not in the same bit position as the leftmost 'one' in the EXPected data.
   E.G. EXP=001777, ACT=001377

3.2.8 ERROR ØØ7

Error ØØ7 occurs in the Moving Inversions Test (sec. 4.1.1), when the P.ioc is testing the K Control Area. A location was written with a pattern. Immediately after the write, the location was read and found to contain an incorrect pattern.

MA = The address of the failing location
EXP= The data pattern expected
ACT= The data pattern actually found

This error indicates a memory data problem. One of the following hardware failures is indicated:

a) A bit was picked up or dropped when the location was written.
b) A bit was picked up or dropped when the location was read.

If the error occurs repeatedly, but only in a single location, the memory chip containing the failing bit for that address is probably defective.

If the error occurs in many locations, but only occurs in a particular Nibble (4 bit field), one of the bus data tranceivers for that nibble is probably defective.

If the error occurs in many locations, and the bits in error are radomly spaced thru-out the word, the memory or bus timing is probably flakey.

If the error occurs in more than one location, but the addresses of the failing locations are similar, there could be crosstalk between the memory data and addressing lines. E.G. All failing addresses end with either 2 or 6.

3.2.9 ERROR 008

Error 008 indicates that the selected K did not complete its Init sequence properly. When the P.ioc enables the K to perform the memory test, the K begins its Init sequence (which includes executing certain microdiagnostics). At the end of the K's Init sequence, the K indicates that it found the K-Control-Area by complementing a pointer word in the Control memory. If the K fails to complement this pointer word within 50 milli-seconds (4.2 seconds for K.ci) of being enabled, an error 008 is reported.

The contents of the 'K Status Register' are displayed with the error report. Refer to the 'K Microdiagnostic User Document' (or the appendices of the 'Offline K Test Selector User Document) for further information on K status values.

If this error occurs, make sure that the 'Requestor Number' parameter given matches the actual requestor number of the K.

3.2.10 ERROR 009

ERROR 009 indicates a Control or Data memory error detected by the K.

```
MA = The 22-bit address of the failing location.
EXP= The data pattern expected by the K
ACT= The data pattern found by the K
```

In addition to the address and the expected/actual data, the K returns an error summary, which is displayed as the last line of the error report. The error summary information indicates whether the error was caused by a Parity error, a Non-Existent-Memory (NXM) error, or a Control Bus (CBUS) error. If the error was not caused by any of the above, the error summary line does not appear in the error report. Refer to section 3.1.1 for further information on the error summary.

3.2.11 ERROR Ø1Ø

ERROR Ø1Ø indicates the P.ioc detected a Parity trap. The 22-bit address of the location that caused the trap is displayed as the 'MA' data in the error report.

```
  MA = The address causing the parity trap.
 VPC = The Virtual PC of the memory test at the time the
       trap occurred.  Reference this address in the
       listing to locate the area of the test where
       the error occurred.
```

Since the data is lost when a parity trap occurs, no EXPected or ACTual data can be displayed. To further localize the problem, it is desirable to disable parity errors and rerun the test. (See section 2.5) If the original failure was in a data-bit position, the memory test will detect and report the error, displaying the EXPected and ACTual data. This will help in tracing the error to a pariticular address and/or bit position. If no further errors are detected after disabling Parity errors, the original failure was in one of the Parity-bits for the address displayed in the parity trap report.

3.2.12 ERROR Ø11

Error Ø11 indicates the P.ioc detected a Non-Existent-Memory trap. A NXM error is caused when no memory responds to a particular address. The 'MA' data in the error report indicates the address which produced the NXM trap. After the trap is reported, the program will attempt to restart the test from the beginning.

```
  MA = The address causing the NXM trap.
 VPC = The virtual PC of the memory test at the time the
       trap occurred.  Reference this address in the
       listing to locate the area of the test where
       the error occurred.
```

If this error occurs at a memory address that should be in your memory configuration, the memory in question is not supplying an 'ACK' to the P.ioc when the specified address is presented on the memory bus. The most probable point of failure is the logic on the memory module that compares addresses on the memory bus to the range of addresses to which the module should respond. The comparitor itself cound be faultly, or the [C IN, C OUT], [D IN, D OUT], or [P IN, P OUT] lines on the backplane could be in error.

3.2.13 ERROR Ø12

Error 12 indicates that no working control memory could be found for a K Control Area. A K Control Area is required to communicate with a K. The control memory must be repaired before the K/P Memory Test can be used. Use the Offline Loaders' "TEST MEMORY" command to test the control memory.

SECTION 4 - TEST SUMMARIES

4.1 TEST SUMMARIES

4.1.1 TEST 000 - MOVING INVERSIONS TEST (FROM P.IOC)

Test 000 is the Moving Inversions (MOVI) memory test used by the P.ioc to test a K Control Area. The K Control Area is used to pass memory test parameters to the K, and to return the results of memory tests to the P.ioc. The Moving Inversions RAM test is used to detect data and addressing problems in dynamic semiconductor memories.

MOVING INVERSIONS ALGORITHM

1) Write 000000 in each location being tested.

2) Read all locations in order from lowest to highest. After reading a location and checking for a zero, rewrite the same location with a single one in the least significant bit. Then reread the location and verify that the write worked correctly.

3) Again read all locations in order from lowest to highest, checking each location to contain the data previously written, and rewriting the data found with a single additional 'one' bit, and rereading to check that the write worked properly.

4) Repeat step 3 until the test pattern consists of a word containing all 'ones' (pattern 177777).

5) Repeat step 3, but this time substitute a single extra zero each time, instead of a one.

6) Continue step 5 until the test pattern consists of a word of all zeroes. (pattern 000000)

7) Repeat steps 1 thru 6, but this time start at the highest memory address each time and work down to the lowest. This will work each memory location from all zeroes to all ones, and back to all zeroes.

8) End of test. All memory is cleared to 000000.

4.1.2 TEST ØØ1 - MOVING INVERSIONS TEST (FROM K)

Test ØØ1 is the Moving Inversions test implemented in the K microcode. The algorithm is identical to that described in section 4.1.1, except that steps 5 and 6 are omitted for speed purposes. When the K detects an error, the remainder of the test is aborted, and the information concerning the error is returned to the P.ioc via the K Control Area. The P.ioc is responsible for displaying the error report.

HSC50 Off-Line Memory Test (TEST MEM)

USER DOCUMENTATION

PRODUCT CODE:	XX-XXXXX-XX
PRODUCT NAME:	HSC5Ø Offline Memory Test
PRODUCT DATE:	12-APR-1983
MAINTAINER:	Diagnostic Engineering - Colorado (CX)
AUTHOR:	Mike Hare

TABLE OF CONTENTS

SECTION 1 - GENERAL INFORMATION

1.1 PROGRAM ABSTRACT

The Offline Memory test is designed to exercise the HSC50 memories. The operator may select CONTROL, DATA, or Program memory to be tested. Two memory testing algorithms are used: the MOVING INVERSIONS algorithm, and the WALKING 1's algorithm. The algorithms are designed to stress the memories in an effort to detect transient errors caused by bus and memory timing problems. Errors are reported on the console terminal as they occur. After reporting a DATA error, testing continues where it left off. After reporting a PARITY or NXM error, testing is restarted from the beginning.

1.2 SYSTEM REQUIREMENTS

Hardware Required:
- a) P.ioc (processor) module with HSC50 Boot ROMS.
- b) At least one M.std (memory) module.
- c) TU58 controller with at least one working drive.
- d) Terminal connected to P.ioc's console interface.

Software Required:
- a) HSC50 Offline Diagnostic Tape

1.3 RELATED DOCUMENTS

- a) HSC50 Offline Diagnostic Loader User Document
- b) HSC50 Offline P.ioc Test User Document
- c) HSC50 ROM Bootstrap User Document
- d) HSC50 Offline K/P Memory Test User Document
- e) HSC50 Offline System Sizer User Document

1.4 HARDWARE ASSUMED TO BE WORKING

In the process of loading the Offline Memory Test, several diagnostics are run. The ROM Bootstrap tests the basic PDP-11 instruction set, tests a partition in Program Memory, and tests the TU58 used for the boot. At this point the bootstrap loads, and begins to execute, the Offline P.ioc test. The Offline P.ioc diagnostic completes the PDP-11 testing begun by the Bootstrap, and tests the remainder of the P.ioc module. After these tests, the Offline Diagnostic Loader is loaded from the TU58 to memory, and control is passed to the Loader. When the operator instructs the Loader to 'TEST MEMORY', the Offline Memory test is loaded and started. Due to the sequence of tests that precede the memory test, the memory test assumes that the P.ioc module and the TU58 are tested and working.

SECTION 2 - OPERATING INSTRUCTIONS

2.1 START-UP PROCEDURE

Follow the steps below to start the memory test:

1) Insert the HSC5Ø OFFLINE TESTS cassette into the TU58's Unit Ø drive. (left hand drive)

2) Power-on the HSC5Ø, or depress and release the INIT button on the HSC5Ø front panel.

3) The TU58 drive-motion LED should light within 1Ø seconds, indicating that the Bootstrap is loading the OFFLINE DIANGOSTIC LOADER to Program Memory.

4) After about 6Ø seconds of loading, the OFFLINE DIAGNOSTIC LOADER will indicate that it has loaded properly by displaying the following:

```
     HSC5Ø OFL Diagnostic Loader, Version Vnn-nn
     Radix=Octal,Data Length=Word,Reloc=ØØØØØØØØ
 ODL>
```

5) Type TEST MEMORY in response to the Loader prompt (ODL>). The TU58 drive-motion LED will light as the memory test is loaded.

6) The memory test will indicate it has been loaded properly by displaying the following:

```
     HSC5Ø OFL Memory Test, Version Vnn-nn
```

7) The memory test now prompts for parameters. See section 2.2 below.

2.2 TEST PARAMETER ENTRY

The Offline Memory Test first prompts:

```
Control(0),Data(1),or Program(2) Memory [0] ?
```

Type a 0 to test Control Memory, type a 1 to test Data Memory, or type a 2 to test Program Memory. After typing the desired digit, type a carriage-return to terminate your response. (Typing just a carriage-return will cause the Control memory test to be selected.) The memory test next prompts for the first address to test:

```
First (min=XXXXXXXX) [min] ?
```

Enter the first address to be tested. Addresses are 8 digits in length. The 'min' address displayed is the lowest address that may be entered for the memory chosen. After typing the address, terminate your response with a carriage-return. (Typing just a carriage-return will cause the first address to default to the 'min' value.) The test now prompts for the last address to test:

```
Last (max=XXXXXXXX) [] ?
```

Type the last address to be tested. The 'max' address displayed is the highest address that is still within the memory chosen. If your system does not have a fully populated memory, the last address that may be tested will be less than the 'max' address displayed. If you choose a last address that exceeds the amount of memory in your system, the memory test will display a 'Non-Existent-Memory' (NXM) error when the test reaches the first address beyond the end of your memory. Use the Offline Loader's SIZE command to determine the actual last address in a given HSC50. Now the test prompts:

```
# of passes to perform (D) [1] ?
```

Enter a decimal number between 1 and 2,147,483,647 (omit commas), which specifies the number of times the memory test should be repeated. (Entering a 0, or just a carriage-return, will result in 1 pass). The memory test will now begin. The test can be aborted at any time by typing a control-c.

NOTE: For any of the above prompts, use the DELETE key to delete mis-typed parameters before the terminating carriage-return is typed. If you note an error in a parameter that has already been terminated with a carriage-return, type a control-c to return to the initial prompt and re-enter all parameters.

2.2.1 RE-USING PARAMETERS

After the first memory test has been specified and completed, the following prompt will be issued:

```
Re-use parameters (Y/N) [Y] ?
```

Answering this prompt with just a carriage-return, or a 'Y' followed by a carriage-return, results in repeating the last test specified, using the previous parameters. Answering the prompt with a 'N' followed by a carriage-return will cause the test to prompt for new parameters.

2.3 SETTING CLEARING FLAGS

N/A

2.4 PROGRESS REPORTS

A complete pass thru the memory test consists of one pass through the Quick Verify test, one pass through the Moving Inversions test, and one pass through the Walking Ones test. After each complete pass, an end of pass message is displayed as follows:

```
End of Pass nnnnnn, xxxxxx Errors, yyyyyy Total Errors
```

The pass count 'nnnnnn' is a decimal count of the number of complete passes made. The Error count xxxxxx indicates the number of errors detected on the current pass. The total error count (yyyyyy) indicates the number of errors detected on all passes of the test completed so far.

2.5 DISABLING P.IOC PARITY TRAPS

When a parity error occurs, it is desirable to know whether the error was produced by the loss or gain of a data bit, or by the loss or gain of a parity bit. When a parity trap occurs in the P.ioc, the data that was read is discarded by the PDP-11. However, a feature of the P.ioc allows parity traps to be disabled. Using this feature, a user can determine if a parity error is being caused by a data or parity bit as follows:

1. After a (P.ioc detected) parity trap is reported, type a control-c to terminate the memory test.
2. Type another control-c to return to the OFL Diagnostic Loader. The loader will prompt: "ODL>".
3. Type "Ex 17770042" followed by a carriage-return.
4. The contents of the P.ioc "Switch Control and Status Register" (SWCSR) will be displayed as follows: "(I) 17770042 nnnnnn".
5. Type "Le * nnnn4n" followed by a carriage-return. The "nnnn4n" represents the previous contents of the register, including a 'one' in bit 5. P.ioc parity traps are now disabled.
6. Return to the memory test by typing "Start", followed by a carriage-return.
7. Re-run memory test with original parameters.
8. If the location that previously produced a parity error now produces a data error, the original parity trap was caused by a data bit problem. The error report will indicate the failing bit via the EXPected and ACTual data displayed.
9. If the location that previously produced a parity trap does not fail when the memory test is re-run, the original parity trap was caused by an error in one of the parity bits (high or low byte) for that word.
10. Type a control-c to return to the Loader, and re-enable parity errors by typing "De 17770042 nnnn0n", followed by a carriage-return. The "nnnn0n" represents the original contents of the P.ioc's SWCSR, before parity traps were disabled.

SECTION 3 - ERROR INFORMATION

3.1 ERROR MESSAGE FORMAT

All error messages produced by the memory test conform to the HSC5Ø Diagnostic Error message format. The first line of an error message contains general information concerning the error, and is mandatory. The second line of an error message consists of text describing the error, and is also mandatory. The third and succeeding lines of an error message are used to display additional information when required, and are optional. Listed below is a typical error message.

```
OMEM>hh:mm T#aaa E#bbb U-ØØØ
  <Text describing error>
  MA -xxxxxxxx
  EXP-yyyyyy
  ACT-zzzzzz
```

WHERE:

```
   hh       = elapsed hours since last bootstrap
   mm       = elapsed minutes
   aaa      = decimal number denoting test
   bbb      = decimal number denoting the error detected
   xxxxxxxx = Address of location causing the error
   yyyyyy   = Data that was expected
   zzzzzz   = Data that was actually found
```

Parity trap and NXM trap errors do not include Expected and Actual data.

3.2 SPECIFIC ERROR MESSAGES

Error messages produced by the memory test can be classed as either data errors or unexpected traps. Error numbers ØØØ thru Ø1Ø are all memory data errors, the only difference among these errors being the exact step in the testing algorithm where they are detected. The step at which a data error occurs can be an important clue to the cause of the error. Errors ØØØ thru ØØ7 are declared in the Moving Inversions algorithm, while errors ØØ8 thru Ø1Ø are declared in the Walking Ones algorithm.

Errors Ø11 and Ø12 are unexpected trap errors. Error Ø11 signifies a parity trap occurred, and error Ø12 indicates a non-existent-memory trap. The reports for unexpected trap errors differ slightly from a data error report, since they do not display expected and actual data.

The following sections (3.2.x) describe the nature of the failure indicated by each error number.

3.2.1 ERROR 000

Error 000 occurs in the Moving Inversions test (see section 4.1.1). A memory location did not contain the expected pattern.

MA = The address of the failing location.
EXP= The data pattern expected.
ACT= The data pattern actually found.

This error can be caused by a DATA ERROR in the address specified, or it may indicate a DUAL-ADDRESSING problem (the location was incorrectly addressed and written when some other location was written.).

DUAL-ADDRESSING ERRORS - At this step in the test, a dual-addressing problem is characterized by:

a) The ACTual data contains a single additional 'one'.

b) The additional 'one' bit occurs immediately to the left of the leftmost 'one' in the EXPected data.

```
E.G.  EXP=000377, ACT=000777
      EXP=077777, ACT=177777
      EXP=000000, ACT=000001
```

In the first example, the location in error was probably written with the pattern 000777 when a lower numbered address was being written with the same pattern. When the location in error was subsequently checked to insure it still contained the 'previous' pattern (000377), it was found to contain the 'next' pattern (000777).

DATA ERRORS - Data errors at this step of the test fall into one of the following classes:

a) The ACTual and EXPected data differ by more than one bit.
   E.G. EXP=017777, ACT=017477

b) The ACTual data contains less 'ones' than the EXPected data.
   E.G. EXP=003777, ACT=001777

c) The bit in error is not in the bit position immediately to the left of the leftmost 'one' in the EXPected data.
   E.G. EXP=000777, ACT=002777

3.2.2 ERROR ØØ1

Error ØØ1 occurs in the Moving Inversions Test (sec. 4.1.1). A location was written with a pattern. Immediately after the write, the location was read and found to contain an incorrect pattern.

```
MA = The address of the failing location
EXP= The data pattern expected
ACT= The data pattern actually found
```

This error indicates a memory data problem. One of the following hardware failures is indicated:

a) A bit was picked up or dropped when the location was written.
b) A bit was picked up or dropped when the location was read.

If the error occurs repeatedly, but only in a single location, the memory chip containing the failing bit for that address is probably defective.

If the error occurs in many locations, but only occurs in a particular Nibble (4 bit field), one of the bus data tranceivers for that nibble is probably defective.

If the error occurs in many locations, and the bits in error are radomly spaced thru-out the word, the memory or bus timing is probably flakey.

If the error occurs in more than one location, but the addresses of the failing locations are similar, there could be crosstalk between the memory data and addressing lines. E.G. All failing addresses end with either 2 or 6.

3.2.3 ERROR 002

Error 002 occurs in the Moving Inversions test (section 4.1.1). A memory location did not contain the expected pattern.

MA = The address of the location in error.
EXP= The data pattern expected.
ACT= The data pattern actually round

This error can be caused by a DATA ERROR in the address specified, or it may indicate a DUAL ADDRESSING problem (The location was incorrectly addressed and written when some other location was being written.).

DUAL ADDRESSING ERRORS - At this step in the test, a dual-addressing problem is characterized by:

a) The ACTual data contains one more 'zero' than the EXPected data.

b) The additional zero occurs in the same bit position as the leftmost 'one' in the EXPected data.

```
E.G.  EXP=003777, ACT=001777
      EXP=000017, ACT=000007
      EXP=177777, ACT=077777
```

In the first example, the location in error was probably written with the pattern 001777 when a lower numbered address was being written with the same pattern. When the location in error was subsequently checked to insure it still contained the 'previous' pattern (003777), it was found to contain the 'next' pattern (001777).

DATA ERRORS - Data errors in this step of the Moving Inversions test fall into one of the following categories:

a) The ACTual and EXPected data differ by more than one bit.
E.G. EXP=177777, ACT=174777

b) The ACTual data contains more 'ones' than the EXPected data.
E.G. EXP=037777, ACT=077777

c) The bit in error is not in the same bit position as the leftmost 'one' in the EXPected data.
E.G. EXP=001777, ACT=001377

3.2.4 ERROR ØØ3

Error ØØ3 occurs in the Moving Inversions Test (sec. 4.1.1). A location was written with a pattern. Immediately after the write, the location was read and found to contain an incorrect pattern.

MA = The address of the failing location
EXP= The data pattern expected
ACT= The data pattern actually found

This error indicates a memory data problem. One of the following hardware failures is indicated:

a) A bit was picked up or dropped when the location was written.
b) A bit was picked up or dropped when the location was read.

If the error occurs repeatedly, but only in a single location, the memory chip containing the failing bit for that address is probably defective.

If the error occurs in many locations, but only occurs in a particular Nibble (4 bit field), one of the bus data tranceivers for that nibble is probably defective.

If the error occurs in many locations, and the bits in error are radomly spaced thru-out the word, the memory or bus timing is probably flakey.

If the error occurs in more than one location, but the addresses of the failing locations are similar, there could be crosstalk between the memory data and addressing lines. E.G. All failing addresses end with either 2 or 6.

3.2.5 ERROR 004

Error 004 occurs in the Moving Inversions test (see section 4.1.1). A memory location did not contain the expected pattern.

```
MA = The address of the failing location.
EXP= The data pattern expected.
ACT= The data pattern actually found.
```

This error can be caused by a DATA ERROR in the address specified, or it may indicate a DUAL-ADDRESSING problem (the location was incorrectly addressed and written when some other location was written.).

DUAL-ADDRESSING ERRORS - At this step in the test, a dual-addressing problem is characterized by:

a) The ACTual data contains a single additional 'one'.

b) The additional 'one' bit occurs immediately to the left of the leftmost 'one' in the EXPected data.

```
E.G.   EXP=000377, ACT=000777
       EXP=077777, ACT=177777
       EXP=000000, ACT=000001
```

In the first example, the location in error was probably written with the pattern 000777 when a higher numbered address was being written with the same pattern. When the location in error was subsequently checked to insure it still contained the 'previous' pattern (000377), it was found to contain the 'next' pattern (000777).

DATA ERRORS - Data errors at this step of the test fall into one of the following classes:

a) The ACTual and EXPected data differ by more than one bit.
   E.G. EXP=017777, ACT=017477

b) The ACTual data contains less 'ones' than the EXPected data.
   E.G. EXP=003777, ACT=001777

c) The bit in error is not in the bit position immediately to the left of the leftmost 'one' in the EXPected data.
   E.G. EXP=000777, ACT=002777

3.2.6 ERROR ØØ5

Error ØØ5 occurs in the Moving Inversions Test (sec. 4.1.1). A location was written with a pattern. Immediately after the write, the location was read and found to contain an incorrect pattern.

MA = The address of the failing location
EXP= The data pattern expected
ACT= The data pattern actually found

This error indicates a memory data problem. One of the following hardware failures is indicated:

a) A bit was picked up or dropped when the location was written.
b) A bit was picked up or dropped when the location was read.

If the error occurs repeatedly, but only in a single location, the memory chip containing the failing bit for that address is probably defective.

If the error occurs in many locations, but only occurs in a particular Nibble (4 bit field), one of the bus data tranceivers for that nibble is probably defective.

If the error occurs in many locations, and the bits in error are radomly spaced thru-out the word, the memory or bus timing is probably flakey.

If the error occurs in more than one location, but the addresses of the failing locations are similar, there could be crosstalk between the memory data and addressing lines. E.G. All failing addresses end with either 2 or 6.

3.2.7 ERROR 006

Error 006 occurs in the Moving Inversions test (section 4.1.1). A memory location did not contain the expected pattern.

```
MA = The address of the location in error.
EXP= The data pattern expected.
ACT= The data pattern actually found
```

This error can be caused by a DATA ERROR in the address specified, or it may indicate a DUAL ADDRESSING problem (The location was incorrectly addressed and written when some other location was being written.).

DUAL ADDRESSING ERRORS - At this step in the test, a dual-addressing problem is characterized by:

a) The ACTual data contains one more 'zero' than the EXPected data.

b) The additional zero occurs in the same bit position as the leftmost 'one' in the EXPected data.

```
E.G.   EXP=003777, ACT=001777
       EXP=000017, ACT=000007
       EXP=177777, ACT=077777
```

In the first example, the location in error was probably written with the pattern 001777 when a higher numbered address was being written with the same pattern. When the location in error was subsequently checked to insure it still contained the 'previous' pattern (003777), it was found to contain the 'next' pattern (001777).

DATA ERRORS - Data errors in this step of the Moving Inversions test fall into one of the following categories:

a) The ACTual and EXPected data differ by more than one bit.
   E.G. EXP=177777, ACT=174777

b) The ACTual data contains more 'ones' than the EXPected data.
   E.G. EXP=037777, ACT=077777

c) The bit in error is not in the same bit position as the leftmost 'one' in the EXPected data.
   E.G. EXP=001777, ACT=001377

3.2.8 ERROR ØØ7

Error ØØ7 occurs in the Moving Inversions Test (sec. 4.1.1). A location was written with a pattern. Immediately after the write, the location was read and found to contain an incorrect pattern.

```
MA = The address of the failing location
EXP= The data pattern expected
ACT= The data pattern actually found
```

This error indicates a memory data problem. One of the following hardware failures is indicated:

a) A bit was picked up or dropped when the location was written.
b) A bit was picked up or dropped when the location was read.

If the error occurs repeatedly, but only in a single location, the memory chip containing the failing bit for that address is probably defective.

If the error occurs in many locations, but only occurs in a particular Nibble (4 bit field), one of the bus data tranceivers for that nibble is probably defective.

If the error occurs in many locations, and the bits in error are radomly spaced thru-out the word, the memory or bus timing is probably flakey.

If the error occurs in more than one location, but the addresses of the failing locations are similar, there could be crosstalk between the memory data and addressing lines. E.G. All failing addresses end with either 2 or 6.

3.2.9 ERROR ØØ8

ERROR ØØ8 occurs in the Walking Ones test (sec. 4.1.2). All locations in the memory under test were written with the pattern ØØØØØØ. Then all locations were read to check that they all contained ØØØØØØ. When the location specified in the error report was read, it did not contain ØØØØØØ.

```
MA = The address of the failing location
EXP= The data pattern expected (ØØØØØØ)
ACT= The data pattern actually found
```

Since all locations were cleared to ØØØØØØ before this error was detected, a dual addressing problem is unlikely. Most likely a bit was picked up when the word was written or read.

If the error occurs repeatedly, but only in one location, the memory chip containing the bit in error for that address is probably marginal.

If the error occurs in many locations, but always occurs in a particular Nibble (4-bit field), one of the bus data tranceivers for that nibble is probably marginal.

If errors occur in many locations, and the bits in error are randomly spaced thru-out the words, the memory or bus timing is probably marginal.

3.2.1Ø ERROR ØØ9

Error ØØ9 occurs in the Walking Ones Test (sec. 4.1.2). One location in the memory under test was written with the pattern 177777, and all the other locations should contain the pattern ØØØØØØ. While reading to check that all other locations are clear, a location was found to contain something other that ØØØØØ.

```
MA = The address of the failing location
EXP= The data pattern expected (ØØØØØØ)
ACT= The data pattern actually found
```

This error could signify either a DATA ERROR or a DUAL ADDRESSING ERROR (The location was incorrectly addressed and written when some other location was being written.).

DUAL ADDRESSING ERROR - At this step of the test a Dual Addressing failure is possible if the ACTual data is 177777. During this part of the test, one location in the memory was written to 177777. When this write was performed, the failing location may also have been addressed and written with the same data. Then when the test was checking that all other locations were in clear, it found the second location with the pattern 177777. If this is a true dual addressing problem, the error will be repeated on each pass of the test.

DATA ERRORS - At this step of the test, a data error is probable if the ACTual data is NOT 177777. Some clues to the possible causes of a data error are listed below.

If the error occurs repeatedly, but only in a particular bit in a single location, the memory chip that contains the failing bit for that location is defective.

HSC50 Off-Line Memory Refresh Test (TEST REFRESH)

USER DOCUMENTATION

PRODUCT CODE:	XX-XXXXX-XX
PRODUCT NAME:	HSC50 Offline Memory Refresh Test
PRODUCT DATE:	21-JAN-1983
MAINTAINER:	Diagnostic Engineering - Colorado (CX)
AUTHOR:	Mike Hare

TABLE OF CONTENTS

SECTION 1 - GENERAL INFORMATION

1.1 PROGRAM ABSTRACT

The Offline Memory Refresh test is designed to find memory problems related to refresh. Patterns are written to memory, and checked after waiting one minute. Three separate patterns are used to test each memory bit (including parity bits) in both the 'one' and 'zero' states. All three HSC50 memories are tested (Program, Control, and Data), although only the Program and Control memories require refreshing. Tests of Data memory are included because some static ram failures resemble refresh problems.

The refresh test can find problems in the memories that will not be detected by the normal memory tests. The refresh test is not intended to be run on memories which fail the normal memory tests.

Each pass of the refresh test requires 3 minutes.

1.2 SYSTEM REQUIREMENTS

Hardware Required:

1. P.ioc (processor) module with HSC50 Boot ROMS.
2. At least one M.std (memory) module, which passes the Offline Memory test and/or the Offline K/P memory test.
3. TU58 controller with at least one working drive.
4. Terminal connected to P.ioc's console interface.

Software Required:

1. HSC50 Offline Diagnostic Tape

1.3 RELATED DOCUMENTS

1. HSC5Ø Offline Diagnostic Loader User Document
2. HSC5Ø Offline P.ioc Test User Document
3. HSC5Ø Offline Memory Test User Document
4. HSC5Ø Offline K/P Memory Test User Document
5. HSC5Ø ROM Bootstrap User Document
6. HSC5Ø Offline Bus Interaction Test User Document

1.4 HARDWARE ASSUMED TO BE WORKING

This test assumes that the HSC5Ø memories pass both the Offline Memory Test and the Offline K/P Memory Test. It is assumed that the memories are working except for the refresh circuitry.

SECTION 2 - OPERATING INSTRUCTIONS

2.1 START-UP PROCEDURE

Follow the steps below to start the refresh test:

1) Insert the HSC5Ø OFFLINE TESTS cassette into the TU58's Unit Ø drive. (left hand drive)

2) Power-on the HSC5Ø, then depress and release the INIT button on the HSC5Ø front panel.

3) The TU58 drive-motion LED should light within 1Ø seconds, indicating that the Bootstrap is loading the OFFLINE DIAGNOSTIC LOADER to Program Memory.

4) After about 45 seconds of loading, the OFFLINE DIAGNOSTIC LOADER will indicate that it has loaded properly by displaying the following:

```
     HSC5Ø OFL Diagnostic Loader, Version Vnn-nn
     Radix=Octal,Data Length=Word,Reloc=ØØØØØØØØ
 ODL>
```

5) Type TEST REFRESH in response to the Loader prompt (ODL>). The TU58 drive-motion LED will light as the memory refresh test is loaded.

6) The refresh test will indicate it has been loaded properly by displaying the following:

```
     HSC5Ø OFL Memory Refresh Test, Version Vnn-nn
```

7) The refresh test now prompts for parameters. See section 2.2 below.

2.2 TEST PARAMETER ENTRY

The Offline Memory Refresh Test first prompts:

```
# of passes to perform (D) [1] ?
```

Enter a decimal number between 1 and 2,147,483,647 (omit commas), which specifies the number of times the refresh test should be repeated. (Entering a Ø, or just a carriage-return, will result in 1 pass being performed.) The test will now begin. The test can be aborted at any time by typing a control-c. Each pass of the test requires 3 minutes to complete.

NOTE: For any of the above prompts, use the DELETE key to delete mis-typed parameters before the terminating carriage-return is typed. If you note an error in a parameter that has already been terminated with a carriage-return, type a control-c to return to the initial prompt and re-enter all parameters.

2.2.1 REUSING PARAMETERS

After the refresh test completes, the following prompt will be issued:

```
Re-use parameters (Y/N) [Y] ?
```

Answering this prompt with a <carriage-return>, or a 'Y' followed by a <carriage-return>, results in repeating the test using the parameters that were previously used. Answering the prompt with a 'N' followed by a <carriage-return> will cause the test to prompt for new parameters.

2.3 SETTING CLEARING FLAGS

N/A

2.4 PROGRESS REPORTS

Each time the refresh test completes one full pass, an end of pass report is displayed. Each pass of the test requires 3 minutes to complete. The end of pass message is displayed as follows:

```
End of Pass nnnnnn, xxxxxx Errors, yyyyyy Total Errors
```

The pass count 'nnnnnn' is a decimal count of the number of complete passes made. The Error count (xxxxxx) indicates the number of errors detected on the current pass. The total error count (yyyyyy) indicates the number of errors detected during the passes completed so far.

SECTION 4 - PROGRAM SUMMARY

4.1 SIZING K's

The first function of the program is to determine the number of K's present in the HSC5Ø, and the status of each K that is present. In order to collect K status, the program must build a K Control Area in the control memory. Beginning at the first address of Control memory, the program will allocate a section of memory for a K Control Area. This piece of Control memory is tested. If the test succeeds without error, the program will begin to collect K status. If the test fails, the program will allocate another control area, which begins immediately beyond the first location that failed. The memory test is repeated on the new Control Area. If the test of the new Control Area passes, the program begins to collect K status. If the new Control Area also fails the memory test, the program keeps trying to find a working piece of memory for a Control Area until the end of Control memory is reached. If no working Control memory is found, the program reports that no Control Area can be allocated, and skips collecting K status.

After a Control Area is allocated, the program enables each K in turn, and allows the K to perform its microdiagnostics. The K in requestor position #1 (K.ci) is allowed 4.2 seconds to complete its microdiagnostics. The K's in all other requestor postions (2 thru 7) are allowed only 5Ø milliseconds to complete their microdiagnostics. When the K indicates that it completed its diagnostics, or when the time limit expires, the K's status is collected and stored in a table.

3.2.3 ERROR Ø3

ERROR Ø3 indicates the P.ioc detected a Parity error. The 22-bit address of the location that caused the trap is displayed as the 'MA' data in the error report.

```
  MA = The address causing the parity trap.
 VPC = The Virtual PC of the memory test at the time the
       trap occurred.  Reference this address in the
       listing to locate the area of the test where
       the error occurred.
```

Since the data is lost when a parity trap occurs, no EXPected or ACTual data can be displayed. The parity error occurred within the program itself, not within the memory that was being tested. After the trap is reported, the program will attempt to restart the test from the beginning.

3.2.4 ERROR Ø4

Error Ø4 indicates the P.ioc detected a Non-Existent-Memory trap. A NXM error is caused when no memory responds to a particular address. The 'MA' data in the error report indicates the address which produced the NXM trap. After the trap is reported, the program will attempt to restart the test from the beginning.

```
  MA = The address causing the NXM trap.
 VPC = The virtual PC of the memory test at the time the
       trap occurred.  Reference this address in the
       listing to locate the area of the test where
       the error occurred.
```

If this error occurs at a memory address that should be in your memory configuration, the memory in question is not supplying an 'ACK' to the P.ioc when the specified address is presented on the memory bus. The most probable point of failure is the logic on the memory module that compares addresses on the memory bus to the range of addresses to which the module should respond. The comparitor itself cound be faultly, or the [C IN, C OUT], [D IN, D OUT], or [P IN, P OUT] lines on the backplane could be in error.

SECTION 4 - TEST SUMMARIES

4.1 TEST SUMMARIES

4.1.1 TEST Ø1 - PATTERN 177777

Test Ø1 consists of filling the memories with the pattern 177777. This sets all data bits, and also sets the upper and lower byte parity bits. The entire Control and Data memories are filled with pattern. All of Program memory not occupied by the Refresh test and the Offline Loader is also filled with the pattern. After filling the memories, the program delays for one minute, then each memory location is read and checked for the pattern. Any errors detected are reported on the terminal.

4.1.2 TEST Ø2 - PATTERN ØØØØØØ

Test Ø2 consists of filling the memories with the pattern ØØØØØØ. This clears all data bits, and sets the upper and lower byte parity bits. The entire Control and Data memories are filled with pattern. All of Program memory not occupied by the Refresh test and the Offline Loader is also filled with the pattern. After filling the memories, the program delays for one minute, then each memory location is read and checked for the pattern. Any errors detected are reported on the terminal.

4.1.3 TEST Ø3 - PATTERN 1ØØØØ1

Test Ø3 consists of filling the memories with the pattern 1ØØØØ1. This sets data bits Ø and 15 , and clears data bits 1 thru 14. Both parity bits will also be cleared. The entire Control and Data memories are filled with pattern. All of Program memory not occupied by the Refresh test and the Offline Loader is also filled with the pattern. After filling the memories, the program delays for one minute, then each memory location is read and checked for the pattern. Any errors detected are reported on the terminal.

HSC50 Off-Line System Sizer (OLSIZE)

USER DOCUMENTATION

PRODUCT CODE:	XX-XXXXX-XX
PRODUCT NAME:	HSC50 Offline System Sizer
PRODUCT DATE:	12-APR-1983
MAINTAINER:	Diagnostic Engineering - Colorado (CX)
AUTHOR:	Mike Hare

TABLE OF CONTENTS

SECTION 1 - GENERAL INFORMATION

1.1 PROGRAM ABSTRACT

The HSC50 Offline System Sizer performs two functions:

1. Displays the sizes of the HSC50's Program, Control, and Data memories.

2. Displays the status of the K's in the HSC50.

The memory sizing information is useful to determine how much of each type of memory a given HSC50 contains. The Sizer displays the starting and ending addresses of each type of memory, and the number of Kwords in each memory. The size information can be used with the Offline Memory tests when specifying the memory test's starting and ending addresses.

If a K passes its Initialization microdiagnostics, the K status information will identify the type of K in a particular requestor position. If a K fails its microdiagnostics, the status information identifies the test that failed. Requestor positions that do not contain a K are identified as empty requestors.

The Offline Sizer does not require any parameters from the user.

1.2 SYSTEM REQUIREMENTS

Hardware Required:

1. P.ioc (processor) module with HSC50 Boot ROMS.

2. At least one M.std (memory) module.

3. TU58 controller with at least one working drive.

4. Terminal connected to the P.ioc's console interface.

Software Required:

1. HSC50 Offline Diagnostic Tape (TU58 media).

1.3 RELATED DOCUMENTS

1. HSC5Ø Offline P.ioc Test User Document
2. HSC5Ø Offline K Test Selector User Document
3. HSC5Ø Offline K/P Memory Test User Document
4. HSC5Ø Offline Memory Test User Document
5. HSC5Ø Offline Bus Interaction Test User Document
6. HSC5Ø ROM Bootstrap User Document

1.4 HARDWARE ASSUMED TO BE WORKING

This program is invoked via the Offline Diagnostic Loader. In the process of loading the Offline Diagnostic Loader, several diagnostics are performed. The ROM bootstrap tests the basic F-11 instruction set, tests a portion of the Program memory, and tests the TU58 used for the bootstrap. Then the bootstrap loads the Offline P.ioc Test, which completes the F-11 tests, and also tests the remainder of the P.ioc module. After the P.ioc test completes, the Offline Loader is loaded from the TU58 and initiated. At this point the P.ioc and the TU58 have been tested, and are assumed to be working. In addition, the first 9 Kwords of Program memory have also been tested, and are assumed to be working. The first word of the Control memory has been tested during the Offline P.ioc test, but the state of the remainder of Control memory is unknown. The Data memory has not been tested.

SECTION 2 - OPERATING INSTRUCTIONS

2.1 START-UP PROCEDURE

The Offline Sizer program is invoked by the Offline Loader's SIZE command. In response to the SIZE command, the Offline Loader will load the Sizer from the TU58 into the Program memory, and then will initiate the Sizer. Upon completion of the Sizer, control is returned to the Offline Loader.

2.1.1 IN CASE OF TROUBLE

If the Offline Loader reports a TU58 error in response to the SIZE command, the tape media may have degraded, making the Sizer file unreadable. Try using the back-up copy of the Offline Diagnostic Tape. If the back-up tape works properly, copy the back-up tape onto a new cassette as soon as possible, and discard the failing cassette.

2.2 PARAMETER ENTRY

The Sizer does not require any user-supplied parameters.

2.3 SETTING/CLEARING FLAGS

The Sizer does not provide any user-modifiable flags.

2.4 PROGRESS REPORTS

The Sizer does not provide any progress reports other than the display of memory sizes and the K configuration information.

2.5 PROGRAM TERMINATION

The Sizer terminates automatically after displaying the size information. Control is returned to the Offline Loader. To repeat the sizer immediately, use the Loader's START command.

SECTION 3 - ERROR INFORMATION

3.1 ERROR MESSAGE FORMAT

The Offline Sizer is not designed to test any of the HSC5Ø hardware. The primary function of the Sizer is to determine the size of the HSC5Ø memories, and the status of each K present in the system. The only error condition explicitly reported by the Sizer is the complete absence of any Control memory.

3.2 SPECIFIC ERROR MESSAGES

3.2.1 NO WORKING CONTROL MEMORY

In order to determine the status of each K present in the HSC5Ø, some working Control memory must be available. To communicate with a K, the program builds a 'K Control Area' in Control memory. If the Sizer determines that no working control memory exists, it will print the following message:

```
  No Working Control Memory, Can't Size K's
```

The program is not able to display the status of the K's present in the subsystem. The size information for the three HSC5Ø memories will be displayed. Note that the memory size information only indicates the addresses to which the memory will respond without an Non-Existent-Memory (NXM) trap. The size information does not indicate whether the memory is able to properly store data. Thus the display may indicate that 64 Kwords of Control memory responds, even though none of it is usable as a K Control Area.

SECTION 3 - ERROR INFORMATION

3.1 ERROR MESSAGE FORMAT

All error messages produced by the refresh test conform to the HSC5Ø Diagnostic Error message format. The first line of an error message contains general information concerning the error, and is mandatory. The second line of an error message consists of text describing the error, and is also mandatory. The third and succeeding lines of an error message are used to display additional information when required, and are optional. Listed below is a typical error message.

```
ORFT>hh:mm T#aaa E#bbb U-ØØØ
  <Text describing error>
  MA -xxxxxxxx
  EXP-yyyyyy
  ACT-zzzzzz
```

WHERE:

hh	= Elapsed hours since last bootstrap
mm	= Elapsed minutes
aaa	= decimal number denoting test
bbb	= decimal number denoting the error detected
xxxxxxxx	= Address of location causing the error
yyyyyy	= Data that was expected
zzzzzz	= Data that was actually found

3.2 SPECIFIC ERROR MESSAGES

The following sections (3.2.x) describe the nature of the failure indicated by each error number.

3.2.1 ERROR Ø1

Error Ø1 indicates the test detected a parity error when reading the pattern from the indicated location. The expected and actual data are included in the error report. This error indicates that a data bit or parity bit was not refreshed (assuming the memory in question passes the Offline Memory Test). If the expected and actual data are both the same, one of the parity bits was not refreshed.

3.2.2 ERROR Ø2

Error Ø2 indicates the test detected a data compare error when reading the pattern from the indicated location. The expected and actual data are displayed in the error report. Note that a parity error did not occur, so more than one bit must have failed to refresh.

4.2 MEMORY SIZING

The next function of the program is to size the three HSC50 memories. The memories are sized by writing to every address within the architectural address range of each memory, and noting which addresses respond. (I.E. Do not produce a Non-Existent-Address (NXM) trap). The architectural address ranges of the HSC50 memories are as follows:

1. Program Memory - Addresses 00000000 - 13777777.

2. Data Memory - Addresses 14000000 - 15777777.

3. Control Memory - Addresses 16000000 - 16777777.

Note that addresses 17000000 thru 17757777 are undefined in the HSC50, and addresses 17760000 thru 17777777 are the PDP-11 I/O page.

While sizing the memories, the program builds a bit map of the 1 Kword chunks of each memory that respond. This bit map is used by the display routine.

4.3 SIZE DISPLAY

The size information is displayed on the local terminal. First the status of each HSC50 requestor position is displayed. Then the beginning and ending address of each chunk of responding memory is displayed. A sample display is as follows:

```
  Req.  Status Value  Meaning
  ----  ------------  -------
   001       001       K.ci
   002       203       K.sti
   003       002       K.sdi
   004       377       Empty Requestor
   005       304       Failed Test 004
   006       004       (Unknown type)
   007       210       (Bad parity)

  RESPONDING      ADDRESS          DECIMAL
  MEMORY           RANGE            WORDS     KWORDS
  ------           -----            -----     ------
 Program   00000000 - 00777777    0131072    00128
 Data      14000000 - 14077777    0016384    00016
 Data      14200000 - 14277777    0016384    00016
 Control   16000000 - 16377777    0065536    00064
```

4.3.1 INTERPRETING K SIZE INFORMATION

The program displays the status returned by each of the HSC50's 7 requestor positions. The requestor position may contain a working K or a K that failed one of its diagnostics, or the requestor may be empty. The program interprets each requestor's status and displays the meaning. The following types of status can be displayed:

1. K.CI - The requestor contains a K.ci which passed its microdiagnostics. Note that only a K.ci can occupy requestor position #1.

2. K.STI - The requestor contains a K.sti (Tape Data Channel) which passed its microdiagnostics.

3. K.SDI - The requestor contains a K.sdi (Disk Data Channel) which passed its microdiagnostics.

4. EMPTY REQUESTOR - The requestor position is empty (does not contain a K).

5. FAILED TEST nnn - The requestor contains a K which failed test nnn of its microdiagnostics. The type of K in the requestor postion is hence unknown, since the K will only supply its type if the microdiagnostics pass. The only time that the type of K can be inferred is in the case of requestor #1, which can only be a K.ci.

6. UNKNOWN TYPE - The status value of the requestor had good parity, but represents an unknown type of K. This condition probably indicates a failing K which is not returning proper status.

7. BAD PARITY - The status value can not be interpreted because the parity of the status byte is even. This condition indicates a failing K which cannot return proper status.

4.3.2 INTERPRETING MEMORY SIZE INFORMATION

The meaning of the memory size information is as follows:

1. RESPONDING MEMORY - The architectural name of the chunk of memory that responded (did not produce a NXM trap).

2. ADDRESS RANGE - The first and last address of the chunk which responded.

3. DECIMAL WORDS - The number of words in the chunk of memory which responded, in decimal.

4. KWORDS - The number of Kwords which corresponds to the number of decimal words within the address range.

Note that in the sample display (section 4.3) there are two separate chunks of Data Memory. This may indicate improper placement of the address jumpers on the memory module. The 'hole' in the Data memory from address 14100000 to address 14177777 will not prevent operation of the HSC50 Control Program, however if the memory jumpers are placed incorrectly some of the Data memory present on the module may be inaccessable, and thus wasted. The presence of many small 'holes' in any of the memories probably indicates some failure on the P.ioc or memory module. If the position of the 'holes' changes from one SIZE command to the next, a failure is definitely indicated.