HONEYWELL

MULTICS ONLINE TEST AND DIAGNOSTICS REFERENCE MANUAL

SOFTWARE

MULTICS ONLINE TEST AND DIAGNOSTICS REFERENCE MANUAL

SUBJECT

Detailed Information Needed by the User to Perform Online Test, Interpret the Results, and Maintain Maximum System Availability

SPECIAL INSTRUCTIONS

This manual supersedes AU77, Revision 2, dated September 1979, and its addenda, AU77-02A, dated December 1980, and AU77-02B, dated August 1981. Refer to the Preface for "Significant Changes."

Throughout the manual, change bars in the margins indicate technical changes and asterisks denote deletions. Appendix B is new and does not contain change bars.

SOFTWARE SUPPORTED

Multics Software Release 10.2

ORDER NUMBER

AU77-03

March 1984

Honeywell

PREFACE

This manual describes the online maintenance capabilities of the Multics System. It contains detailed information concerning installation, access, tests, and termination requirements.

Initial installation of the subsystem is covered in Section 2.

Users with a basic knowledge of the operating subsystem need only concern themselves with Section 3 as regards TOLTS operation. New users will find it beneficial to familiarize themselves with the complete document.

This manual should be used in conjunction with the following Customer Service Representative documentation:

Series 6000 Test and Diagnostics Manual (58008382)

MPC Subsystem Handbook (TNG-335-004)

T&D Microfiche Documentation

Significant Changes in AU77-03

New items incorporated with this release are:

1. MOLTS -- Disk Confidence Program, Tape Compatibility Program, and Tape Shotgun Program.

2. MOLTS, POLTS, ISOLTS, and COLTS -- a number of additional extensions to the pcd request.

3. Added Appendix B -- Formatting Disks with MTR.

Revised items are:

1. load_tandd_library command.

2. MOLTS, POLTS, ISOLTS, and COLTS -- msg request.

 File No.: 1L53, 1U53 AU77-03

CONTENTS

SECTION 1

INTRODUCTION

The online test and diagnostic (T&D) function utilized in the Multics System is part of the overall maintenance program dedicated to providing maximum system availability for production use. The present online program is part of a continuing evolutionary maintenance plan intended to minimize usage of the system and its peripherals in a test environment.

Online tests running concurrently with normal system activities are a proven approach to system maintainability and availability. Testing may be initiated by either site operations or field engineering personnel. Individual access to the system or its related peripherals is controlled by the System Administrator.

This manual describes the installation requirements and executive programs for the Total OnLine Test System (TOLTS), MPC OnLine Test Subsystem (MOLTS), Peripheral OnLine Test Subsystem (POLTS), ISolated OnLine Test Subsystem (ISOLTS), and the Communications OnLine Test Subsystem (COLTS).

SECTION 2

TOLTS INSTALLATION

TOLTS provides a limited command environment and an interface for the MOLTS, POLTS, ISOLTS, and COLTS active test subsystems. Included in TOLTS is the TOLTS command processor, a limited process overseer called tolts_overseer_, plus a "quit" handler that limits a TOLTS user to interactively call and execute only test and diagnostic test pages.

All TOLTS programs that interface with the Multics system reside in the segment >system_library_tools>bound_tolts_.

All test pages and slave mode executions that are driven by the bound_tolts_ modules are located in the keyed sequential file tandd_deck_file located in >system_library_tandd.

TOLTS INSTALLATION INSTRUCTIONS

TOLTS testing is normally initiated by Honeywell field engineering personnel. The system administrator provides a project for field engineering to log into. This project, HFED, is normally registered at Multics cold startup time. If the HFED project is missing, or if additional personids are desired, register the HFED project by creating and installing the following Project Master File (PMF) entry:

```
/*  Project Master File for Online T&D  */

Projectid:          HFED;
Initproc:           tolts_overseer_;
Grace:              60;

personid:           <name>;
    .                  .
    .                  .
    .                  .
end;
```

Add the following commands to the system_start_up.ec (part 3) to give the HFED project the necessary access to the gates rcp_sys_, rcp_priv_, phcs_, and tandd_.

```
hpsa >system_library_1>rcp>sys_ re *.HFED.*
hpsa >system_library_1>rcp>priv_ re *.HFED.*
hpsa >system_library>phcs_ re *.HFED.*
hpsa >system_library_1>tandd_ re *.HFED.*
hpsa >system_control_1>opr_query_data rw *.HFED.*
hpsa >system_control_1>rcp>tandd.acs rw *.HFED.*
hpsa >system_control_1>cdt r *.HFED.*
```

In order to run COLTS on a given DATANET 355/6670 Front-End Network Processor (FNP), the following must be done:

1. Definition of COLTS channel
   The COLTS executive channel must be configured in the Channel Definition Table (CDT). The name of this channel is X.c000, where X is the name of the FNP. The entry in the Channel Master File (CMF) should be as follows:

   name: X.c000; service: slave; baud: 9600; line_type: COLTS; comment: "COLTS executive channel";

   See the description of the CMF in MAM Communications, Order No. CC75.

2. COLTS module in FNP core image
   The bindfile for the FNP must include the name "mclt" in the "order" statement. See the description of the FNP bindfile in MAM Communications for more information.

3. Access required
   The user must have at least re access on the >sc1>tandd_ gate, r access to >system_control_1>cdt, and at least rw access on all of the following access control segments in >system_control_1>rcp:

   tandd.acs
   X.c000.acs (where X is the FNP name)
   {channel_name}.acs (where {channel_name} is the name of any subchannel of the FNP that is to be tested)

SECTION 3

TOLTS EXECUTIVE

This section details the overall operating characteristics of the TOLTS executive, tolts_overseer_. It also describes TOLTS operating restrictions and instructions and includes MOLTS and POLTS test requests, in a condensed version, for those users needing only a reminder of operating procedures.

TOLTS EXECUTIVE PROGRAM FUNCTIONS

The TOLTS executive is the overall test system that coordinates and communicates with the other executive programs. It provides an interface between the subsystem executives and Multics. All tests are run under MOLTS, POLTS, or ISOLTS subsystem executives (no testing is accomplished by TOLTS itself).

Multiprogrammed Peripheral Controller Testing

MOLTS, a subexecutive of TOLTS, permits test and diagnostic programs to be run on the Microprogrammed Peripheral Controller (MPC), and peripheral devices attached to MPCs. Refer to Section 4 for specific subsystems that may be tested with MOLTS.

Peripheral Testing

POLTS provides a set of test and diagnostic peripheraal programs that test peripherals in a system environment. Refer to Section 5 for specific peripherals that may be tested with POLTS.

Processor Testing

ISOLTS provides the capability to test processors (in a multi processor and system controller configuration) completely independent and isolated from the Multics environment. Refer to Section 6 for operational details of the ISOLTS subsystem.

Front-End Network Processor Testing

COLTS provides the capability of functional testing of the FNP adapters in an online environment. Refer to Section 7 for operational details of the COLTS subsystem.

TOLTS OPERATING RESTRICTIONS

The following restrictions apply to the MOLTS and POLTS subsystems.

1. When running Isolation Test Routines (ITRs) on a Unit Record MPC, all I/O drivers with configured devices running on that MPC must be logged out. On the other hand, if Micro-coded Device Routines (MDRs) or POLTS is being run on a device connected to a Unit Record MPC, only the I/O driver configured for that device must be logged out. Refer to Section 4 for a description of ITRs and MDRs.

2. Before running POLTS or MDR Test Pages on a disk drive, the disk drive must be configured as an I/O device. This is done by the system operator and can be achieved by using the UDSK configuration card or the set_drive_usage initializer command. For an explanation of the UDSK configuration card and the set_drive_usage command, refer to the Multics Operators Handbook, Order No. AM81.

3. In order to run ITRs on a disk MPC, there must be a redundant data path to the devices connected to that MPC (dual crossbarred MPCs).

4. In order to run MDRs on a disk device, there must be a redundant MPC data path so that the system may perform I/O on the other devices connected to the MPC through which MDRs are being executed.

The following restriction applies to the ISOLTS subsystem:

At least two processors and two system controllers must be configured in order to run ISOLTS.

TOLTS OPERATING INSTRUCTIONS

A user logging in under the HFED project types:

```
login Person_id HFED
```

where Person_id is the registered name of a person on the HFED project.

After the password is given, TOLTS responds by displaying the following message:

```
***tolts executive version xxxxxx on yyyyyy at zz.zzz

***enter "polts", "molts", "isolts", "colts," "quit", or "msg"
???
```

where:

1. xxxxxx
   is the TOLTS version date expressed as yymmdd (year month day).

2. yyyyyy
   is the current date expressed as yymmdd.

3. zz.zzz
   is the time in hours and thousandths of hours.

After the three question marks (???), the user types one of the expected responses. If polts, molts, isolts , or colts is typed, control is passed to the appropriate subsystem (refer to Sections 4, 5, 6, or 7). If quit is typed, TOLTS issues the following message and then logs out:

```
***tolts executive version xxxxxx off yyyyyy at zz.zzz
```

If msg is typed, the system prepares to accept one line of user input (not to exceed 80 characters) to be routed to the operator master console. If an interaction with the operator at the master console is required, the message must be terminated with a "?" which in turn prompts the operator for a response. This facility allows the user to request resources (disk usage, tape label, disk packs, etc.) prior to invoking ISOLTS, COLTS, MOLTS, or POLTS. The following sequence of lines is associated with an example session at a users' terminal:

```
???msg
***enter one line message up to 80 char
???is tape drive tapa_04 available?
no          (this is the operator response -- see below)
```

The next sequence is associated with the system operator's console and is a response to the question asked above. In this case the operator is saying "no", the requested tape drive is not available. Following is the system operator's sequence to the above question:

```
is tape drive tapa_04 available?
Please reply: x oqr followed by message of up to 80 characters
x oqr no
```

The next sequence is associated with the user's terminal which simply instructs the operator (i.e., the comment is without "?" and no operator response is necessary):

```
???msg
***enter one line message up to 80 char
???please use tape number m1234 for next mount request
```

If a communication is required after invoking an executive, enter a "test msg" request in response to the "???" prompt and the same facility is used.

CONDENSED VERSION OF COLTS, MOLTS, AND POLTS OPERATION

The remainder of this section is devoted to those users familiar with the operation of the TOLTS subsystems who need only sketchy reminders to accomplish operational procedures.

Detailed descriptions for MOLTS, POLTS, and COLTS are contained in Sections 4, 5, and 7, respectively. There is no condensed version for ISOLTS (refer to Section 6).

Test Requests

Any time a quit signal is issued, by pressing the appropriate key on the terminal (e.g., QUIT, BRK, ATTN), TOLTS responds by displaying "???" and pauses for entry of a new request. Up to eight COLTS or POLTS request may be active at one time, but in MOLTS, only one ITR or MDR can be active at a given time. MOLTS ITRs may be interrupted, but not terminated.

MOLTS, POLTS, AND COLTS COMMON REQUESTS

lstal	list active text pages.
msg	communication request to system operator.
pcd	list current peripheral configuration.
w	wrap up operation.

MOLTS REQUESTS

mdciccdd	run MDC (where iccdd is IOM/channel/device).
modciccdd	communication request for MDC.
medciccdd	terminate MDC.
mdriccdd	run MDR.
modriccdd	communication request for MDR.
medriccdd	terminate MDR.
mmticcdd	run MTR/RMC.
momticcdd	communication request for MTR/RMC.
memticcdd	terminate MTR/RMC.
mpcicc	run ITR.
mopcicc	communication request for ITR.
mepcicc	terminate ITR.
mtciccdd	run MTC.
motciccdd	communication request for MTC.
metciccdd	terminate MTC.
mtgiccdd	run MTG.
motgiccdd	communication request for MTG.
metgiccdd	terminate MTG.

POLTS REQUESTS

piccdd	run new test.
poiccdd	communication request.
peiccdd	terminate test.

COLTS REQUESTS

cnccss	run test page (where nccss is the processor, IOM channel, and subchannel numbers).
conccss	communication request.
cenccss	terminate test.

OPTION CHARACTERS AND CONTROL MNEMONICS

MOLTS	POLTS	COLTS	OPTION	DESCRIPTION
	*		a	accumulate error messages in collection file.
*	*	*	b	bypass messages. The halt (h) option overrides.
*	*		exx	output transient error message. If numeric characters xx follow, that value is used to override the standard number of retries. If "-1" is typed, the standard is restored.
*	*	*	h	halt for input of options following errors.
*	*	*	i	inform the user of normal "end tests." The halt (h) option appends "enter options."
*	*	*	l	loop on current test (except 0).
*	*	*	n	negate the following option character. (MOLTS -- valid only if preceding b, h, i, l, p, r, t, or the z options. POLTS -- valid only if preceding b, e, h, i, l, p, r, t, x, or the z options. COLTS -- valid only if preceding a, b, e, h, i, l, p, r, or the t options.)
*	*	*	o	go to "enter options."
*	*	*	p	issue an end pass message when back jump is detected. The halt (h) option appends "enter options."
*	*	*	r	issue an end cycle message and cycle back to the first test. The halt (h) option appends "enter options."
*	*	*	s	skip to next test.
*	*	*	txxx	jump to test xxx (xxx may be one, two, or three numeric characters).
*	*		u	unload option facilitates use of currently loaded tape or disk without operator intervention.
	*		v	specifies device instead of subchannel test.
		*	x	enables the extended status portion of the error message. It is initially forced "on" automatically.
*	*		z	trace I/O setup. Provides a four line MOLTS, or five line POLTS message for each test I/O issued.

Control mnemonics (prefixed with ".")

MOLTS	POLTS	COLTS	OPTION	DESCRIPTION
*	*	*	.go	return to test page where interrupted.
	*		.man	force test page to manual mode.
*	*	*	.opt	enter options.
*	*	*	.pr2 .prt	append error messages to a print file.
	*	*	.rseq	resequence tests.

*	.seqxx	accept TDL input for manual test page.
*	.seqaxx	accept data input for manual test page.
* *	.seqt	sequence tests in special order.
*	.stnxx	initialize manual test page.
*	.strxx	set single test mode for manual test page.
* * *	.tal	error tally. Option p or r must be on.
* * *	.test e	force termination.
* * *	.test w	wrap up COLTS, MOLTS, or POLTS.
* * *	.typ	error message sent to terminal.
* * *	.wait	test page is put in a "wait" condition.

Special Test Page Control Mnemonics for HSLA test pages

*	.async	test as asynchronous.
*	.eloop	loop on subtest in error.
*	.gp	reset .async and .synch options.
*	.neloop	turn off .eloop option.
*	.nsloop	turn off .sloopxx option.
*	.sloopxx	restart test and loop on subtest xx.
*	.synch	test general purpose subchannel as TLPK synchronous subchannel.

SECTION 4

MOLTS EXECUTIVE

MOLTS provides the online capability to run tests on the MPC and its peripheral devices. MOLTS also provides the capability to reboot MPC Firmware while the system is up and running Multics.

NOTE: A restriction exists when running ITRs on disk MPCs. ITRs cannot be run on a disk MPC unless the system has another path to the disk drives connected to the MPC.

FUNCTIONAL CAPABILITIES

MOLTS is a test subsystem that operates under the TOLTS executive program. It provides a driver and a user interface with ITRs, MDRs, MTRs (Media Test and Analysis Routines), RMCs (Removal Media Certification Routines), MDC (MOLTS Disk Confidence Programs), MTC (MOLTS Tape Compatibility), and MTG (MOLTS Tape Shotgun).

Isolation Test Routines

The ITRs are a group of micro-coded diagnostic routines that overlay the MPC firmware and test the hardware logic in the MPC. If an error occurs in one of the ITRs, the MPC is halted at the location of the error. MOLTS receives a timeout and reports an error message that contains the ITR identifier. The Customer Service Representative (CSR) can determine which circuit board and group of integrated circuit chips are defective by examining the various register displays in the MPC and then using this data in conjunction with the fault dictionary for the failed test. After the error is corrected, or if all tests run correctly, MOLTS reboots the MPC firmware and returns control of the MPC to the operating system.

Micro-coded Device Routines

The MDRs are a group of highly comprehensive peripheral device diagnostic routines that are loaded and executed in the MPC under control of the MPC firmware and the central system. Since the MDRs are executed from the MPC, they have high resolution in testing the MPC/device interface and also use various diagnostic modes within the devices. If an error occurs while running MDRs, MOLTS displays an error message that contains a fault dictionary index. By using the fault dictionary index and looking at the dictionary entry, an CSR can determine what component or group of components in a peripheral device is defective. Since firmware is not destroyed by running the MDRs, MOLTS returns control of the MPC either after test completion, or after an error has occurred.

MTRs consist of seven program modules designed to systematically evaluate and enhance the data handling properties of the fixed media contained within Mass Storage Units (MSU) 500, 501, and 509. Each of the seven modules execute under control of the MOLTS subsystem. A full array of MTR module identifiers and test option commands are available to ensure user versatility in detecting media defects and minimizing operational impact.

The MTR programs are designed to provide an economical method of extending the useful life of the nonremovable mass storage media, which is generally determined by the severity and number of media defects. Media defects may be the result of inconsistent manufacturing processes or from damage to the media platter. Regardless of the cause of the defect, the net result is either a degradation of data fidelity or the loss of data. For simplicity, all media defects are termed either "hard" or "soft" failures, where soft implies that data written over a platter defect is retrievable some of the time and hard implies that data is nonretrievable regardless of the number of attempts made. Hard and soft defects result in two types of failures: count field or data field errors. The purpose of MTR is to overcome the effects of either of these two types of media problems by reorganizing the recorded data in an attempt to eliminate the operational effect of the faulty area on the platter.

The MTR programs are progressive in performing their designed function. Program mtr1, while being a "read only" program, provides limited analysis of failure symptoms. The error indications determined by the execution of mtr1 are further analyzed by successive MTRs. This process continues until the detected problem is eliminated through data reorganization or a determination is made that the store media is unsalvageable. A functional description of each MTR follows.

mtr1 - device read only testing.
Full protection of the data base is ensured by allowing execution only when write permission is not granted. Program mtr1 is comprised of the following subtests, which are also capable of detecting data field and count field media defects.

subtst 1 - read track data field test
subtst 2 - read/verify count field data test
subtst 3 - list defective and assigned alternate tracks
subtst 4 - online test information

mtr2 - media certification testing.
The primary purpose is to detect and diagnose new media defects. All defects are classified as:

1. Transient read error - defined as a soft failure (1 out of 9 read attempts fail). This type of defect is completely acceptable for continued use by the operation system.

2. Marginal track - defined as a soft failure (anywhere from 2 to 8 of 9 read attempts fail). This type of defect is detrimental to further system operation.

3. Defective track - defined as a hard failure (9 out of 9 read attempts fail).

Detection and diagnosis is directly related to the degree of media certification performed, which is directly related to the number of

certification write patterns specified for execution. The expected confidence level for detecting all unrecoverable media defects is:

Pattern 1 - 97.%
Pattern 2 - 99.0%
Pattern 3 - 99.4%
Pattern 4 - 99.6%
Pattern 5 - 99.7%
Pattern 6 - 99.8%
Pattern 7 - 99.9%

Program mtr2 is comprised of the following subtests:

subtst 1 - write/read media certification
subtst 2 - online test information

mtr3 - track and cylinder reformatting.
The purpose is to provide the capability to repair defective tracks that were diagnosed by mtr2. In addition, it provides the optional capability to assign alternate tracks to those tracks identified as defective (no alternate assigned). Program mtr3 is comprised of the following subtests:

subtst 1 - reformat 1 track (good)
subtst 2 - reformat 1 cylinder (good)
subtst 3 - reformat 1 track (defective)
subtst 4 - assign alternate tracks on logical device

mtr4 - online reformat information.
The purpose is to provide online information pertinent to reformatting. The information is output at the controlling terminal or at a dedicated printer (pr2) if the .prt option is invoked.

mtr5 - PRMFL emergency track repair.
The purpose is to provide the capability to repair diagnosed media defects on a device that contains "live system data."

A mtr6 - reformat physical device.
The purpose is to provide the CSR with the capability to record a prescribed format on a Head Device Assembly (HDA) in a field environment. This includes replacement, upgrade, and downgrade of HDAs.

B mtr7 - special format functions.
The purpose is to provide the CSR with the capability to assign alternate tracks on the physical device, and to create and write Error Logging Track (ELTs). Program mtr7 is comprised of the following subtests:

subtst 1 - assign alternate tracks on physical device
subtst 2 - write ELTs

ENHANCEMENTS

Following is a description of the most important enhancements made to MTR (Rev. 3).

Test Speed Up

Most segments are designed with test speedup features and execution time is reduced up to 70% and more. The speedup was accomplished through the application of a cylinder chain I/O. The Mass Store Controller is dedicated to execute the chain I/O or cylinder. It cannot be interrupted until the chain has completed execution.

Multiple Device Testing

Testing is provided with all of the MTR segments. A maximum of seven segments can be put into execution at the same time. As an example, if there are seven HDAs that require formatting, all seven can be formatted at the same time.

PRMFL Emergency Track Repair

Program mtr5 provides a safe way to try and repair individual tracks on an HDA that contains "live system data."

The user must become familiar with the term "locked sector." A locked sector defines the technique used by mtr5 to simulate an unrecoverable sector. The lock is provided through the format recorded on a given track. If the system tries to read a locked sector the result is a data field sync byte error (13-26 or 13-27).

The unlock is provided through any write command to a (physical) locked sector. Thus when a data restore is performed, it unlocks the sector during the write process and the system has access to valid data. In summary, the locked sector protects the system from being able to recover garbage data until a data restore is performed.

The track repair test requires one spare alternate track to exist on the pack before the test can begin. This alternate track holds the data while the bad track is being repaired. If the bad track is repairable, the data is copied from the alternate track and the alternate track is released and can be used at another time. If the track is not repairable, it is formatted bad with the alternate track as its alternate. It is possible that the bad track is already an alternate for some other bad track and the process is essentially the same.

Fast Read Scan

This is a special option provided in Test 1 (subtst 1). When the "all track" test mode is selected, the option is made available through a test request. Its primary application is to quickly scan a logical device for hard media errors. Multiple copies can be utilized but no more than one per odd/even device pair.

The Fast Scan executes in approximately 6 to 10 minutes, even under heavy system load conditions.

The ideal time to use this feature is just prior to a full system initialize. The fast scan should be executed on all fixed media disk devices connected to the system. Bad tracks should be repaired using program mtr5 (PRMFL Emergency

Track Repair). Following this the data restore should be performed on the "locked sectors." This allows cleanup of the bad tracks and at the same time have the complete data base intact.

Removable Media Certification Routines

RMCs consist of six program modules designed to systematically evaluate and enhance the data handling properties of the removal media contained within Mass Storage Units, (.DS450, .DS400, .DS310, .DS191, .DS190, and .DS181). Each of the six modules execute under control of the MOLTS subsystem. A full array of RMC module identifiers and test option commands are available to ensure user versatility in detecting media defects and minimizing operational impact.

The RMC programs are designed to provide an economical method of extending the useful life of the removable mass storage media, which is generally determined by the severity and number of media defects. Media defects may be the result of inconsistent manufacturing processes or from damage to the media platter. Regardless of the cause of the defect, the net result is either a degradation of data fidelity or the loss of data. For simplicity, all media defects are termed either "hard" or "soft" failures, where soft implies that data written over a platter defect is retrievable some of the time and hard implies that data is nonretrievable regardless of the number of attempts made. Hard and soft defects result in two types of failures: count field or data field errors. The purpose of RMC is to overcome the effects of either of these two types of media problems. Bad tracks are marked defective (no alternate assigned) and the user has the option to assign an alternate.

The RMC programs are progressive in performing their designed function. Program rmc1, while being a "read only" program, provides limited analysis of failure symptoms. The error indications determined by the execution of rmc1 are further analyzed by successive RMCs. This process continues until the detected problem is eliminated through data reorganization or a determination is made that the store media is unsalvageable. A functional description of each RMC follows.

rmc1 - device read only testing.
Full protection of the data base is ensured by allowing execution only when write permission is not granted. Program rmc1 is comprised of the following subtests, which are also capable of detecting data field and count field media defects.

subtst 1 - read track data field test
subtst 2 - read/verify count field data test
subtst 3 - list defective and assigned alternate tracks
subtst 4 - online test information

rmc2 - media certification testing.
The primary purpose is to detect and diagnose new media defects. All defects are classified as:

1. Transient read error - defined as a soft failure (1 out of 9 read attempts fail). This type of defect is completely acceptable for continued use by the operation system.

2. Marginal track - defined as a soft failure (anywhere from 2 to 8 of 9 read attempts fail). This type of defect is detrimental to further system operation.

3. Defective track - defined as a hard failure (9 out of 9 read attempts fail).

Detection and diagnosis is directly related to the degree of media certification performed, which is directly related to the number of certification write patterns specified for execution. The expected confidence level for detecting all unrecoverable media defects is:

Pattern 1 - 97.%
Pattern 2 - 99.0%
Pattern 3 - 99.4%
Pattrn 4 - 99.6%
Pattern 5 - 99.7%
Pattern 6 - 99.8%
Pattern 7 - 99.9%

Program rmc2 is comprised of the following subtests:

subtst 1 - write/read media certification
subtst 2 - online test information

rmc3 - track and cylinder reformatting.
The purpose is to provide the capability to repair defective tracks that were diagnosed by rmc2. In addition, it provides the optional capability to assign alternate tracks to those tracks identified as defective (no alternate assigned). Program rmc3 is comprised of the following subtests:

subtst 1 - reformat 1 track (good)
subtst 2 - reformat 1 cylinder (good)
subtst 3 - reformat 1 track (defective)
subtst 4 - assign alternate tracks on logical device

rmc4 - online reformat information.
The purpose is to provide online information pertinent to reformatting. The information is output at the controlling terminal or at a dedicated printer (pr2) if the .prt option is invoked.

rmc5 - PRMFL emergency track repair.
The purpose is to provide the capability to repair diagnosed media defects on a device that contains "live system data."

The track repair test requires one spare alternate track to exist on the pack before the test can begin. This alternate track holds the data while the bad track is being repaired. If the bad track is repairable, the data is copied from the alternate track and the alternate track is released and can be used at another time. If the track is not repairable, it is formatted bad with the alternate track as its alternate. It is possible that the bad track is already an alternate for some other bad track and the process is essentially the same.

This segment is not available for IFAD A.2 release.

rmc6 - reformat physical device.
The purpose is to provide the CSR with the capability to record a prescribed format on a disk pack in a field environment.

MOLTS Disk Confidence

MDC is a "removable disk device and media verification" program that consists of five separately executable worst-case pattern "write" tests, and any data "read" tests. The tests use GCOS I/O dialog to exercise the 451 removable disk device. It can be used to distinguish between device and pack failures. No attempt is made to diagnose a problem down to an optimum replaceable unit (ORU), but extensive status translation is provided for "error detection" messages. The controller "error correction mechanism" is disabled to permit an unbiased view of disk errors and permits up to three command retries on error detection.

The program's read tests do not compromise customer or system disk pack security since a nondata transfer DCW is used (i.e., a read disk method that discards data before it can be seen). Typically, write tests are run on scratch disk packs and result in data destruction of the pack.

Tests 1 and 3 (Sequential Read and Write) have a finite execution time whereas Tests 4, 5, and 6 (Random Write, Read, and Write-Read) run until terminated. The program writes and reads the entire usable disk surface, not just the T&D cylinder. In addition, all status is translated into easy-to-understand error messages.

PURPOSE

Verify removable disk media and device performance of the 451 disk devices with the following tests:

mdciccdd -OR- mdciccddt1
read data sequentially.

mdciccddt3
write worst-case data sequentially.

mdciccddt4
write worst-case data randomly.

mdciccddt5
read data randomly.

mdciccddt6
write-read worst-case data randomly.

options can be set with the test call:

modciccddoo

and terminate testing with:

medciccdd

HARDWARE FUNCTIONS TESTED

None directly. This program is used to differentiate between media and device problems. It is not intended to be an extensive T&D tool with board callouts, etc. Random address Tests utilizing Tests 4, 5, and 6 are excellent servo exercisers.

WRITE TESTS

The write tests of this program DESTROY pack data. The user MUST decide whether or not to SAVE the overlaid pack contents before starting this test.

RESTRICTIONS

1. Pack formatting -- the program assumes the test pack is formatted.

2. Prior testing required -- POLTS should run error-free on the device to be tested.

3. Units required -- a 451 removable disk device.

TEST DESCRIPTIONS

Test 1 -- read sequential

This test (mdciccdd -OR- mdciccddt1) reads the contents of the entire disk pack sequentially, skipping only the label track. This is accomplished by utilizing a nondata transfer DCW (IONTP) so no actual data is transferred for security reasons, but rather a data integrity check is made at the disk surface. The test reads data records (either worst-case or customer data) one trackful at a time. The test moves slowly and methodically from beginning to end of the disk. The program can be queried at any time to determine how far the test has progressed, or the test can be terminated early by using the .r option (see Program Options below).

Test 3 -- write sequential

This test (mdciccddt3) writes sequentially on the entire disk, skipping only the first label track. Full track records are written with each I/O connect. Each track is written and followed by a read verify to validate every track (i.e., two passes per track are executed). Refer to "Write Tests" above as regards SAVING pack data before executing this test request.

Nine possible worst-case data patterns are available although the "all sixes" default pattern is the most common. The test starts each record at sector zero of each track and runs from beginning to end of the disk. The program can be queried at any time to determine how far the test has progressed, or the test can be terminated early by using the .r option.

Test 4 -- write random

This test (mdciccddt4) writes worst-case patterns randomly throughout the disk, one record at a time. There is no way to determine where the test may start or end write-verifying each track-size record. Typically, each record bridges adjacent track boundaries. No time limit is set for this test, that is, it MUST be terminated by invoking either the .r option or the "test mediccdd" request.

Test 5 -- read random

This test (mdciccddt5) is similar to Test 4 except a read is performed after each seek instead of a write. The records read may be either worst-case patterns or customer data. No data is transferred to the mainframe for security reasons (i.e., the data is merely checked for validity at the disk pack surface). The test MUST be terminated by invoking either the .r option or the "test mediccdd" request.

Test 6 -- write-read random

This test (mdciccddt6) is a combination of Tests 5 and 6. In addition to random seeks, it performs random reads and writes. It is terminated by the .r option or the "test mediccdd" request.

PROGRAM OPTIONS

This program is a MOLTS test page, and as such, executes all of the standard MOLTS options. In addition, it uses two unique options described below.

.r (report option, invoked during either a read or write)
the report option provides the ability to see how far the program has progressed on the disk and how many errors and retries have occurred.

NOTE: Solid errors are those persisting after three program retries. All sequential tests (t4, t5, and t6) must be terminated using this option.

The following user/program dialog is presented as an explanatory definition of the .r option:

```
<QUIT>      issued by pressing appropriate key
??? test modciccdd.r
**t(mdciccdd  ) tx enter options -
**t(mdciccdd  ) dev xxx   test data = xxxxxxxxxxxx
current disk confidence test
current addr under test (cyl/hd/sect) = cxxx/hxxx/sxx
unrecov errs = xx   no. of retries = xx
do you wish to end program?
```

If EOM (end of message or return) is the response to the above question, then the program continues writing or reading to the disk. If "yes" is the response, the program terminates with:

```
**t(mdciccdd ) normal term 1
```

.1 through .8 (worst-case data pattern option, write only)
the normal worst-case write data pattern for this program is "all sixes"

(default). It is possible, during initial call of the program, to set up for writing any of the following eight data patterns:

OPTION	DATA PATTERN
.1	777752227777
.2	225070452250
.3	777725257045
.4	277666666666
.5	646264625252
.6	525265532665
.7	532665532665
.8	532600000000
(default)	666666666666

A typical input request to invoke this option would be:

```
???test mdciccddo      (o for option)
**t(mdciccdd  )to enter options - .4      (may be .1 through .8)
```

NORMAL TERMINATION

The following message is provided for normal termination of sequential write and read tests.

```
**t(mdciccdd  ) dev type = xxx  test data = xxxxxxxxxxx
rmvbl disk seq (rd or wr) confidence test complete
unrecov errors = xx    no. of retries = xx
```

Random write and read tests must be user-terminated. This is accomplished by executing a .r option and results in the following message:

```
**t(mdciccdd  )  dev type = 451    t5
rmvbl disk random (rd or wr) conf test
current addr under test (cyl/hd/sect) = cxx/hxx/sxx
unrecov errs = xx         no of retries = xx
do you wish to end prog?  yes or eom to continue.
```

Termination of the test (prior to normal termination) is accomplished with the test call "medciccdd".

ABNORMAL TERMINATION

In addition to the standard forced termination conditions described in "Test Descriptions" (above), and "Test Wrapup Request" (described later in this section), there are conditions that cause the program to terminate prematurely. When one of the following conditions occur, the program provides a self-explanatory error message.

- Other users on device
- Device under test nonremovable
- Device not attached
- Single bit errors (no status returned, power off, etc.)
- IOM errors

MEDIA CERTIFICATION

This program should be run on at least two drives. If both drives report the same error at the same address on the same pack, label this area as defective and rerun the test. Keep in mind only those error messages indicated as a third retry are solid. The one and two retry errors are system recoverable, but result in reduced performance.

DEVICE CERTIFICATION

If a pack runs error-free using this program on one drive, but picks up random errors on a second drive, it is likely that the second drive has hardware problems.

SEQUENTIAL DATA VOLUME OVERVIEW

One record of worst-case data words (2560 words on 40 sections of 64 words each) is written or read on each track of this 800 cylinder pack, starting at location 240 (cylinder 00, head 04, sector 00) and ending at location 2271540 (T&D cyl 814, head 18, sector 39). A total of 15481 tracks are tested. Four label tracks at the beginning are not tested.

RANDOM DATA VOLUME OVERVIEW

Random data records are written and read as described under the sequential data volume overview except they may begin anywhere within a track including the start of each track. Random records can span track and cylinder boundaries (sequential records do not).

MDC MESSAGES

MDC error messages are described later in this section under the "MOLTS Messages" header.

MOLTS Tape Compatibility

MTC writes and reads predictable random data and record size information (4-1K words) to and from a magnetic tape device (only one device can be run per test page). When the read pass takes place, the program regenerates the data previously written to the tape device so that a "data compare" can be made (i.e., compare the write data information with that read back from the tape device to verify compatibility). A message is output at the completion of each pass, either read or write, asking what type of pass to accomplish next. Error messages are provided immediately, as they occur.

Multiple devices can be tested by recalling the test page. The test page can be invoked from either a local console or a remote terminal.

PURPOSE

To perform tape compatibility and device quality testing of any Magnetic Tape Subsystem using MPC Controllers. The specific devices tested are MTU410, MTU500, MTU600, MTU610, PENA 30/32, and MTU MPT. The test is invoked via the test call:

mtciccdd

options are set with the test call:

motciccddoo

and termination of the test page (prior to normal termination) is accomplished with the test call:

metciccdd

TEST DESCRIPTION

Refer to "MOLTS Operating Instructions" (described later in this section) for examples of (1) New Test Request (mtciccdd), (2) Test Communication Request (motciccddoo), and (3) Test End Request (metciccdd).

Invoking "test mtciccdd" loads the program. The system then responds with several prompts, each prompt requesting intervention (i.e., the entry of parameters). The parameter sequence can be bypassed by answering "yes" to the "bypass parameters" prompt, in which case default values are provided for testing. In the event of a "no" response to the bypass prompt, the system responds with seven prompts, the first being:

no. of records [s,eom,xxxxx,?]

where "s" or end-of-message implies use of the default value (1500 records) for loading. To run other than the standard value, the value (xxxxx) of records desired is entered (e.g., "2500" results in reading or writing 2500 records).

The second prompt is:

enter density [2,5,8,16,62,?]

The response to this prompt sets the density to be used for the test page. For example, a response of "5" implies a density of 556 bpi. The default value is the highest density available for the device being tested.

The third prompt is:

h200/2000 read-write binary commands [y,n,eom,?]

where a response of "y" sets the test to run using h2000 mode commands, and "n" or "eom" sets the test to run using standard binary commands. (Default is no h200/2000 commands.) This prompt is not provided when testing is being accomplished on an MTC500.

The fourth prompt is:

read, write, eom, or help [r,w,eom,?]

where a response of "r" sets the test for a read pass, "w" sets the test for a write pass, and "eom" terminates the test page. (There is no Default value.) This prompt is always provided.

The fifth prompt is:

enter error retry count [x,s,eom,?]

where "x" is a response of "0" through "8", which sets the retry count to 0-8 (i.e., the number of retries performed on any data alert condition), "s" sets the retry to "64", and "eom" sets the retry count to 0 (zero retries). (Default is zero retries.)

The sixth prompt is:

standard compatibility or mpc code conversion [s or m]

where a response of "s" implies standard binary commands, and "m" indicates ASCII and EBCDIC commands. When using code conversion, the test page translates (to and from) the ASCII/EBCDIC codes. (Default is standard mode.)

The seventh prompt is:

mpc options.. ascii, ebcdic, ascii-ebcdic [y,n,?]

The response to this prompt can be shown as a combination of "y"s and "n"s. For example, the response "yny" sets the test to run using ASCII commands alone (y), no EBCDIC commands alone (n), and mixed ASCII-EBCDIC commands (y). Any combination of ASCII-EBCDIC commands can be run. (There is no default.) This prompt is not provided unless the test is being run utilizing code conversion (see "sixth prompt" above).

When a "?" is entered in response to any of the prompts, the system responds with a help message.

PROGRAM OPTIONS

Due to the nature of the program, it is recommended that only the following options be used for this test.

b
: bypasses error message output.

o
: goes to "enter options" following processing of the complete option string containing the "o".

.go
: returns to the test page where interrupted.

.pr2
: outputs forthcoming test page error messages to a file for printing.

.typ
: outputs forthcoming test page error messages to the terminal.

.wait
: puts the test page in a wait condition.

Refer to "Standard Test Page Option" below for a complete description of the options.

MTC MESSAGES

Four different types of MTC error messages are associated with this test page, all of which are described later in this section under the "MOLTS Messages" header.

If the .pr2 or .prt option is not being used, the error messages are output to the terminal. One message can vary between 2 to 10 lines of information; therefore, it is recommended that the .pr2 or .prt option be used to route the error messages to an accounting file or a printer. A warning message is provided by the system when the .pr2 or .prt option is not used, to notify of a possible tie-up of the terminal in the case of multiple error messages.

MOLTS Tape Shotgun

MTG is a magnetic tape device program designed to stress-test a device at its extreme limits (only one device can be run per test page). The test verifies whether or not the handler can read, write, start/stop, and position tape correctly. MTG performs most commands in a random sequence, writing and reading data in a random pattern and length (5-1K words). It also tests the S2000 interchange facility, allowing for selection of initial character position and terminating character position. Error messages are provided immediately, as they occur.

Multiple devices can be tested by recalling the test page. The test page can be invoked from a terminal.

PURPOSE

To perform tape compatibility and device quality testing of any Magnetic Tape Subsystem using MPC Controllers. The specific devices tested are MTU410, MTU500, MTU600, MTU610, PENA 30/32, and MTU MPT. The test is invoked via the test call:

```
mtgiccdd
```

options are set with the test call:

```
motgiccddoo
```

and termination of the test page (prior to normal termination) is accomplished with the test call:

```
metgiccdd
```

TEST DESCRIPTION

Refer to "MOLTS Operating Instructions" (described later in this section) for examples of (1) New Test Request (mtgiccdd), (2) Test Communication Request (motgiccddoo), and (3) Test End Request (metgiccdd).

Invoking "test mtgiccdd" loads the program. The system then responds with several prompts, each prompt requesting intervention (i.e., the entry of parameters). The parameter sequence can be bypassed by answering "yes" to the "bypass parameters" prompt, in which case default values are provided for testing. In the event of a "no" response to the bypass prompt, the system responds with three prompts, the first being:

enable h2000 mode, icp, tcp [y,n,?]

The response to this prompt can be shown as a combination of "y"s and "n"s. For example, the response "yny" sets the test to run using h2000 mode commands (y), without using initial character position (n), and using terminate character position (y). Any combination of the above can be selected. The default value for this prompt is "yyy". This prompt is bypassed when running the test on an MTC500 controller and the default in this case is "nnn".

The second prompt is:

density [2,5,8,16,all,?]

The response to this prompt sets the density to be used for the test page. For example, a response of "5" implies a density of 556 bpi. If "all" is entered, all available densities for the selected handler are used in a random sequence. Default value is all.

The third prompt is:

run time in minutes

The response to this prompt sets the amount of time the test is to run. For example, a response of "90" implies the test is set to run for 90 minutes. The maximum time for the program to run is 300 minutes. Default value is 60 minutes.

When a "?" is entered in response to any of the prompts, the system responds with a help message.

PROGRAM OPTIONS

Due to the nature of the program, it is recommended that only the following options be used for this test.

b
: bypasses error message output.

h
: halts following error messages.

o
: goes to "enter options" following processing of the complete option string containing the "o".

.go
: returns to the test page where interrupted.

.pr2
: outputs forthcoming test page error messages to a file for printing.

.prt
: outputs forthcoming test page error messages to an allocated printer.

.typ
: outputs forthcoming test page error messages to the terminal.

.wait
: puts the test page in a wait condition.

MTG MESSAGES

Five different types of MTG error messages are associated with this test page, all of which are described later in this section under the "MOLTS Messages" header.

If the .pr2 or .prt option is not being used, the error messages are output to the terminal. One message can vary between 2 to 10 lines of information; therefore, it is recommended that the .pr2 or .prt option be used to route the error messages to an accounting file or a printer. A warning message is provided by the system when the .pr2 or .prt option is not used, to notify of a possible tie-up of the terminal in the case of multiple-error messages.

Hardware Tested

Currently, MOLTS is capable of performing micro-coded tests on the following MPCs and MPC peripheral devices:

CONTROLLER DESCRIPTION (ITRs)	TEST PROGRAM LABEL
Microprogrammed Peripheral Controller	itrd
DEVICE DESCRIPTION (MDRs)	
MASS STORAGE:	
DSS190 Model A Mass Store	mdrn
DSS190 Model B Mass Store	mdrs/mdc
DSS191 Mass Store	mdrs/mdc
MSU0400 Mass Store	mdrs/mdc
MSU0450/0451 Mass Store	mdrs/mdc
MSU0500/501	mdrs/mdc
MAGNETIC TAPE STORAGE:	
MTS500 7/9 Track Tape Handler	mdrs/mtc/mtg
MTU0400/MTU0500 7/9 Track Tape Handler	mdrs/mtc/mtg
MTU0410 7/9 Track Tape Handler	mdrs/mtc/mtg
MTU0600 7/9 Track Tape Handler	mdrs/mtc/mtg
MTU0610/630 7/9 Track Tape Handler	mdrs/mtc/mtg
CARD READER:	
CRZ301	mdra
CARD PUNCH:	
CPZ301	mdrc
PRINTER:	
PRT303	mdre
PRU1200/PRU1600 Device Adapter	mdrf

MOLTS also runs MTRs on the MSU500/501/509 and RMCs on the DS450/400/310/191/181 Mass Storage Devices.

Documentation for the test programs is located in the T&D Microfiche Documentation Box.

MOLTS OPERATING INSTRUCTIONS

MOLTS is entered by answering the TOLTS query:

```
*** enter "polts", "molts", " isolts", "colts", "quit", or "msg"
???
```

with the keyword "molts". After a slight pause for the core image loading of the MOLTS slave executive, an input prompt/request for input (???) is again displayed. When the prompt is responded to with a valid test request, MOLTS responds with a greeting message.

```
??? test x...
*** molts executive versions wwwwww xxxxxx on yyyyyy at zz.zzz
```

where

1. test x...
   can be any of the requests listed in the following paragraphs. The maximum character length of any request is 11 characters (including options).

2. wwwwww
   is the Multics interface module version date expressed as yymmdd.

3. xxxxxx
   is the slave MOLTS version date expressed as yymmdd.

4. yyyyyy
   is the current date expressed as yymmdd.

5. zz.zzz
   is the time in hours and thousandths of hours.

Any time a quit signal is issued, by pressing the appropriate key on the terminal (e.g., QUIT, BRK, ATTN), MOLTS responds by displaying "???" and pauses for entry of a new msg (refer to "Tolts Operating Instructions" in Section 2) or test request. Only one ITR or MDR can be active at any given time.

Test pcd Request

The test pcd request can be entered in three forms: (1) "pcd" displays the current peripheral configuration, (2) "pcd xxx" where xxx identifies a specific subsystem device type display (e.g., pun, dsk, tap), and (3) "pcd xxxx" where xxxx identifies a particular device display (e.g., punb, dska, tapa). Examples of each follow.

```
??? test pcd

molts configuration:

iom a a nsa iom on scu port 0 is online
iom b a nsa iom on scu port 1 is online
dska 451 16 units;  starting with device no. 1
     020xx          primary channel of 4 logical channels on mpc card mspa
     026xx          secondary channel of 4 logical channels on mpc card mspa
     124xx          secondary channel of 4 logical channels on mpc card mspb
     122xx          secondary channel of 4 logical channels on mpc card mspb
dskb 451 16 units;  starting with device no. 17
     120xx          primary channel of 4 logical channels on mpc card mspb
```

```
     126xx          secondary channel of 4 logical channels on mpc card mspb
     024xx          secondary channel of 4 logical channels on mpc card mspa
     022xx          secondary channel of 4 logical channels on mpc card mspa
dskc 501 16 units;  starting with device no. 1
     028xx          primary channel of 4 logical channels on mpc card mspc
     130xx          secondary channel of 4 logical channels on mpc card mspd
dskd 501 16 units;  starting with device no. 17
     128xx          primary channel of 4 logical channels on mpc card mspd
     030xx          secondary channel of 4 logical channels on mpc card mspc
dske 451  8 units;  starting with device no. 1
     132xx          primary channel of 4 logical channels on mpc card mspe
     134xx          secondary channel of 4 logical channels on mpc card mspe
dskf 501 16 units;  starting with device no. 1
     036xx          primary channel of 4 logical channels on mpc card mspf
     138xx          secondary channel of 4 logical channels on mpc card mspg
dskg 501 16 units;  starting with device no. 17
     136xx          primary channel of 4 logical channels on mpc card mspg
     038xx          secondary channel of 4 logical channels on mpc card mspf
tapa 630  2 units;  starting with device no. 1
     610  6 units;  starting with device no. 3
     630  2 units;  starting with device no. 9
     016xx          primary channel of 2 logical channels on mpc card mtpa
prtd 00801 model 1600 prt with 160 columns and a 600 print belt
     008xx          primary channel of 5 logical channels on mpc card urpa
prta 00901 model 1200 prt with 136 columns and a 600 print belt
     009xx          secondary channel of 5 logical channels on mpc card urpa
prtb 01001 model 1600 prt with 136 columns and a 600 print belt
     010xx          secondary channel of 5 logical channels on mpc card urpa
rdra 01101 model 500 rdr
     011xx          secondary channel of 5 logical channels on mpc card urpa
puna 01201 model 300 pun
     012xx          secondary channel of 5 logical channels on mpc card urpa
prtc 10801 model 1201 prt with 136 columns and a 600 print belt
     108xx          primary channel of 2 logical channels on mpc card urpb
prte 10901 model 1600 prt with 136 columns and a 600 print belt
     109xx          secondary channel of 2 logical channels on mpc card urpb
punb 11001 model 300 pun
     110xx          primary channel of 2 logical channels on mpc card urpc
rdrb 11101 model 500 rdr
     111xx          secondary channel of 2 logical channels on mpc card urpc
hcha 03000 model 1 hch
     030xx          special purpose channel
hchb 03100 model 1 hch
     031xx          special purpose channel
```

where

1. prta
   is the Multics device name.

2. 00901
   is the IOM/channel/device in decimal (iccdd).

3. 1200
   is the model (in this case -- 1200 lpm).

4. 136
   is the number of print columns.

5. 600
   is the print belt part number (or cartridge number).

```
??? test pcd pun

pun configuration:
```

```
puna 01201 model 300 pun
     012xx          secondary channel of 5 logical channels on mpc card urpa
punb 11001 model 300 pun
     110xx          primary channel of 2 logical channels on mpc card urpc

??? test pcd punb

pun configuration:

punb 11001 model 300 pun
     110xx          primary channel of 2 logical channels on mpc card urpc

??? test pcd hch

hch configuration:

hcha 03000 model 1 hch
     030xx          special purpose channel
hchb 03100 model 1 hch
     031xx          special purpose channel
```

Test List Request

To list the active test pages in MOLTS, enter one of the following requests:

```
test mlstal
```

-or-

```
test lstal
```

Note: Only one MDR or ITR test can be active at any given time.

Sample output:

```
??? test lstal
molts lstal:
in execution:
**0(mdr01603) t//04
```

Test Wrapup Request

To wrap-up (stop) all MOLTS testing, enter one of the following requests:

```
test mw
```

-or-

```
test w
```

Sample output (MDR driver):

```
??? test w
**0(mdr01602) forced term 1
molts wrapping up
***molts executive wrap-up code  td2  p.t.  9246
  ic 002771  ir 100000  ba 000000  er 000  ar 000000632402  qr 777777770077
    tr 00070202
  x0 000300  x1 202001  x2 000116  x3 000036  x4 402000  x5 002254  x6 024255
    x7 000152
***molts executive wrap-up code  td2  p.t.  9246
```

If the .pr2 option was in force the register banner information plus an octal dump of the MOLTS segment would be appended to the print file to be dprinted later.

Sample output (ITR driver):

```
??? test w
**0(mpc016) normal term 1
**0(mpc016) normal term 2
***molts executive wrap-up code  td2  p.t.  3797
  ic 002771  ir 100000  ba 000000  er 000  ar 000000632402  qr 777777770077
    tr 00070066
  x0 000300  x1 202001  x2 000116  x3 000036  x4 000004  x5 000000  x6 000000
    x7 000152
**0(mpc016): molts loading mtp601 firmware m601 rev. h1
***molts executive wrap-up code  td2  p.t.  3797
```

Notice that firmware is reloaded before MOLTS termination is complete.

New Test Request

To start execution of a new test, enter one of the following requests:

```
test mdciccddooo (to run an MDC)
test mdriccddooo (to run an MDR)
test mtciccddooo (to run an MTC)
test mtgiccddooo (to run an MTG)
```

-or-

```
test mpciccooooo (to run an ITR)
```

-or-

```
test mmticcddooo (to run an MTR or RMC)
```

Note: Only one MDR or ITR test can be active at any given time.

where

1. iccdd
   is the IOM/channel/device.

2. o...
   is the new option character set.

MDR EXAMPLE

In some cases the test to be run cannot be determined by processing the configuration deck. In these cases an additional input indicating the optional device type is required. (In the following examples mt700 and mt500 are used.)

```
??? test mdr01603
***molts executive versions 790524 790518 on 790523 at 12.69
**0(mdr01603) enter hardware device type? ("?" = help) mt700
**0(mdr01603) short wait, allocation queued
**0(mdr01603) start mdr16a-mdrdvr, ttldat 790327, phy./log. id t//04
**0(mdr01603)
                    mth current device under test
port   opi  dvc    record    current   dvc    mth   7/9  load  write
```

```
 #          add     cap       density  speed  model  trk  comp  enable
 02     y    3     16,8,       1600     200   700     9    yes   yes   !!!!
**0(mdr01603)
what densities to be run ?
enter(   /8/16   /eom=all)
**0(mdr01603)
canister mounted (y/n) n
**0(mdr01603)
do you want to run the data
security erase test (y/n) y
**0(mdr01603) normal term 1
***molts executive version 790518 off 790523 at 12.79 p.t. 158456


??? test mdr11707u
***molts executive versions 790524 790518 on 790523 at 13.23
**0(mdr11707) enter hardware device type? ("?" = help) mt500
**0(mdr11707) start mdr14a-mdrdvr, ttldat 790327, phy./log. id t//04
**0(mdr11707)
no device 07 available for test
do you want port override option.
enter (y or n) y
enter port # (use two digits only 00,09,12) 02
**0(mdr11707) normal term 1
***molts executive version 790518 off 790523 at 13.30 p.t. 68145
```

ITR EXAMPLE

```
??? test mpc016
***molts executive versions 790524 790518 on 790523 at 12.80
**0(mpc016) start tdc16a-itrdvr, ttldat 790509, phy./log. id t//04
**0(mpc016) normal term 1
**0(mpc016): molts loading mtp601 firmware m601 rev. h1
***molts executive version 790518 off 790523 at 12.81 p.t. 13493
```

Also refer to the example included under "MOLTS ITR Error Message" later in this section.

MTR EXAMPLE

To run MTR on MSU500/501/509s, the complete HDA (odd/even devices) must be assigned to I/O. A particular MTR test can be called by the test request:

```
test mmticcddtx
```

where x is the desired MTR number (1, 2, 3, 4, 5, 6, or 7). (For additional information refer to "Media Test and Analysis Routines" described above.)

RMC EXAMPLE

Similar to MTR above with the exception that "x" represents only 1-6. (For additional information refer to "Removable Media Certification" described above.)

MTR/RMC OPTIONS

The following description defines the application of the MTR or RMC test ".options". To request control (after MTR or RMC is invoked with the test mmticcdd request), type:

```
test momticcdd.x
```

where

1. iccdd
   is the IOM/channel/device.

2. .x
   is the option. Options can be selected from the following:

```
.e   display transient error cylinder/head address.
.f   turn off all active ".options".
.i   interrupt test and output summary reports.
.r   report current cylinder/head address.
.s   append summary reports if .t option is active.
.t   report test progress every 100 cylinders.
```

MDC EXAMPLE

```
??? test mdc12020i
***molts executive versions 831012 830727 on 831031 at 21.61
**0(mdc12020) short wait, allocation queued
**0(mdc12020) start tdmdc1-mdcdvr, ttldat 830328, phy./log. id t//04
**0(mdc12020)
  ** rmvbl disk seq read confidence test  **
**0(mdc12020)
 dev type = 451   t1
read    fail address---0212100      c093/h01/s00
status------424000000000      342400722402
*maj/sub  =  02/40         retried 0 times
attention -
device offline -
**************************************************
**0(mdc12020)
dev attention. manual intervention required.
program will wait 1 minute and then continue.
??? test medc12020
**0(mdc12020) forced term 1
test e request received
***molts executive version 830727 off 831031 at 21.73 p.t. 196331
```

MTC EXAMPLE

```
??? test mtc01605
***molts executive versions 830926 830727 on 831010 at 20.59
**0(mtc01605) short wait, allocation queued
**0(mtc01605) start tdmtc1-mtcdvr, ttldat 830310, phy./log. id t//04
**0(mtc01605)
*bypass parameters [y,n or eom,?] y
**0(mtc01605)
*read, write, eoj, or help [r,w,eom,?] w
**0(mtc01605)
t c o m p   e r r o r   r e p o r t
20:38   mtu 610     9trk  200ips  density 6250  den avail   1600, 6250
wtb    maj/sub  03/10   lateral parity       iom stat 431000000000 033124000000
device extended status in hex:      failing record no.    69      retries    0
00ec3b6f05a30800442b13130a40102002c408c008098a000000
**0(mtc01605)
t c o m p   e r r o r   r e p o r t
20:38   mtu 610     9trk  200ips  density 6250  den avail   1600, 6250
wtb    maj/sub  03/10   lateral parity       iom stat 431000000000 032743000000
device extended status in hex:      failing record no.   509      retries    0
40ed2b6f05e30800442b1313284110230004000008 1bc0000400
cmd code error          cmd even parity
**0(mtc01605)
t c o m p   e r r o r   r e p o r t
```

```
20:39   mtu 610     9trk  200ips  density 6250  den avail   1600, 6250
wtb    maj/sub  03/10   lateral parity       iom stat 431000000000 032612000000
device extended status in hex:      failing record no.  1325      retries    0
00ec3b6f05a30800442b13133a4010208004889c080982000000
**0(mtc01605)
 write pass totals
***number of errors this pass        3
***number of records this pass    1500
**0(mtc01605)
*read, write, eoj, or help [r,w,eom,?] r
**0(mtc01605)
t c o m p   e r r o r   r e p o r t
20:40   mtu 610     9trk  200ips  density 6250  den avail   1600, 6250
rtb    maj/sub  03/10   lateral parity       iom stat 431000000000 035226040000
device extended status in hex:      failing record no.   211      retries    0
00ac2b6f05c30800442b13133860000702c402d008098a000000
**0(mtc01605)
t c o m p   e r r o r   r e p o r t
20:41   mtu 610     9trk  200ips  density 6250  den avail   1600, 6250
rtb    maj/sub  03/10   lateral parity       iom stat 431000000000 034604040000
device extended status in hex:      failing record no.  1253      retries    0
00ac2b6f05c30800442b1313e86000060a040ad0080982000000
**0(mtc01605)
t c o m p   e r r o r   r e p o r t
20:41   mtu 610     9trk  200ips  density 6250  den avail   1600, 6250
rtb    maj/sub  03/10   lateral parity       iom stat 431000000000 034542040743
device extended status in hex:      failing record no.  1329      retries    0
00ac2b6f05c30800442b1313d86000060b4408f308098a080000
**0(mtc01605)
  read pass totals
***number of errors this pass        3
***number of records this pass    1500
**0(mtc01605)
*read, write, eoj, or help [r,w,eom,?]
**0(mtc01605) normal term 1
***molts executive version 830727 off 831010 at 20.70 p.t. 210361


MTG EXAMPLE

??? test mtg01605i
***molts executive versions 830926 830727 on 831010 at 20.48
**0(mtg01605) short wait, allocation queued
**0(mtg01605) start tdmtg1-mtgdvr, ttldat 830315, phy./log. id t//04
**0(mtg01605)
bypass parameters [y,n,?] y
**0(mtg01605)
s h o t g u n   e r r o r   r e p o r t
20:32   mtu 610     9trk  200ips  density 1600  den avail   1600, 6250
wtb    maj/sub  02/04   device in standby    iom stat 420402000000 122500700000
device extended status in hex
02ec1b6f05230800442b131358019000000000000000000000000
standby
**0(mtg01605)
***please fix device***
**0(mtg01605)
s h o t g u n   e r r o r   r e p o r t
20:33   mtu 610     9trk  200ips  density 1600  den avail   1600, 6250
wtb    maj/sub  03/10   lateral parity       iom stat 431000000000 022030000000
device extended status in hex
00ec1b6f05230800442b0b0b186090062044ff00080992000000
??? test metg01605
**0(mtg01605) forced term 1
test e request received
***molts executive version 830727 off 831010 at 20.57 p.t. 99922
```

Test Communications Request

To change the option character set presently in force or to enter a single control mnemonic for a request presently in execution, enter one of the following requests:

```
test modciccddoo (for an MDC)
test modriccddoo (for an MDR)
test motciccddoo (for an MTC)
test motgiccddoo (for an MTG)
```

-or-

```
test momticcddoo (for an MTR or RMC)
```

-or-

```
test mopciccoooo (for an ITR)
```

where o... is the new option character set (the "o" option interrupts a test page).

Sample output:

```
??? test modr016o
***(molts executive) c(modr016o): invalid input-
invalid device number
??? test modr01607o
***(molts executive) c(modr01607o): invalid input-
test entry not found
??? test modr01603o
**0(mdr01603) t1 enter options: .pr2
**0(mdr01603) t1 enter options: z
**0(mdr01603) normal term 1
molts dump file >udd>HCSRD>CSRD>!BBBJJMqlPWpMjc.molt.dump has
  been queued for printing
***molts executive version 790518 off 790523 at 12.88 p.t. 72730
```

Refer to "Standard Test Page Options" later in this section for a description of the z option and the .pr2 control mnemonic used in the above example. In addition, refer to the .tal control mnemonic example for a sample use of the mopciccoooo request.

Test End Request

To terminate execution of a MOLTS test, enter one of the following requests:

```
test medciccdd (for an MDC)
test medriccdd (for an MDR)
test metciccdd (for an MTC)
test metgiccdd (for an MTG)
```

-or-

```
test memticcdd (for an MTR or RMC)
```

-or-

```
test mepcicc (for an ITR)
```

Sample output:

```
??? test medr01602
**0(mdr01602) forced term 1
test e request received
***molts executive version 790518 off 790525 at 15.89 p.t. 8802
```

Standard Test Page Options

The following error and control options can be entered in response to an enter options message (test communication request) or designated in the option string of a new test request.

Although these options are implemented in that they are processed and the appropriate execution flags are set in the MOLTS executive, whether or not they are effective depends upon the particular test running.

OPTION CHARACTERS AND DESCRIPTION

b bypass error message output. Bypass overrides a pass or cycle message unless halt is set. Halt forces these messages to be displayed (overriding the bypass option).

exx output transient error message. If numeric characters xx follow, that value is used to override the test page standard value for the number of retries to be made. If a "-1" is typed, then the test standard is restored. This control of retry count is independent of the turn "on" or "off" capabilities of the e or ne option.

h halt for input of options following error messages, end test messages, end pass messages, and end cycle messages.

i inform the user of each normal "end test." If halt (h) is also specified, then an "enter options" is appended to the end test message. The end test message is overridden if the next segment is being called, an end pass message is being output, or if a cycle ends.

Sample output:

```
t0,h,p,r  enter options: i
**0(mdr01602),h,i,p,r end t0,next t1  enter options:
```

l loop on current test (cannot loop on test 0).

n negate the following option character (valid only if preceding b, e, h, i, l, p, r, t, or the z options).

o go to "enter options" following processing of the complete option string containing the "o."

p issue an end pass message any time a back jump is detected by the next test sequencing. If halt (h) is also specified, then an "enter options" is appended to the end pass message. The error tallies for the current pass are reset when an "end pass" is reported or when "p" is being turned on.

r issue an end cycle message any time a normal test page termination occurs and cycle back to the first test in the current sequence. If halt (h) is also specified, then an "enter options" is appended to the end cycle message. The error tallies for both the current pass and current cycle are reset when "end cycle" is reported. The error tallies for the cycle are reset whenever "r" is being turned on.

Sample output:

```
??? test mdr01602r
***molts executive versions 790524 790518 on 790525 at 15.98
**0(mdr01602) enter hardware device type? ("?" = help) mt700
**0(mdr01602) short wait, allocation queued
**0(mdr01602) start mdr16a-mdrdvr, ttldat 790327, phy./log. id t//04
**0(mdr01602)
                mth current device under test
port   opi  dvc    record    current   dvc    mth   7/9  load  write
 #          add     cap      density speed  model  trk  comp  enable
 01     y    2     16,8,      1600     200   700     9    no    yes   !!!!
**0(mdr01602)
what densities to be run ?
enter(   /8/16   /eom=all)
**0(mdr01602)
canister mounted (y/n) n
**0(mdr01602)
do you want to run the data
security erase test (y/n) n
**0(mdr01602) end cycle 1
```

At this point the MDR driver recycles to the start of MDR sequence.

s unconditionally skip to the next test.

txxx if turned on (no preceding negate), then unconditionally jump to the first occurrence of the test in the current sequence. The test number xxx must follow and must be nonzero, and can consist of one, two, or three numeric characters. For segmented test pages, a value outside of the current segment causes a jump to the corresponding segment without further processing of the current option string. If turned off (ntxxx), then the test number must be in the current segment and sequence, and not the forced termination test number.

u unload option. Facilitates the use of a tape or disk drive without requiring operator intervention. The media currently mounted on the tape or disk drive is used without operator authentication or mounting required. This option is especially useful when running MDRs on a tape or disk drive that does not load properly. This option is only available with the initial option string.

Sample output:

```
??? test mdr01603u
***molts executive versions 790524 790518 on 790523 at 13.61
**0(mdr01603) enter hardware device type? ("?" = help) mt700
**0(mdr01603) start mdr16a-mdrdvr, ttldat 790327, phy./log. id t//04
**0(mdr01603)
                 mth current device under test
port   opi  dvc    record    current   dvc    mth   7/9  load  write
 #          add     cap      density  speed  model  trk  comp  enable
 02     y    3     16,8,      800      200   700     9    yes   yes   !!!!
**0(mdr01603)
what densities to be run ?
enter(   /8/16   /eom=all) 8
**0(mdr01603)
canister mounted (y/n) n
**0(mdr01603)
do you want to run the data
security erase test (y/n) y
**0(mdr01603) normal term 1
***molts executive version 790518 off 790523 at 13.76 p.t. 102015
```

z trace I/O setup. A message is output for each test I/O issued.

Sample output:

```
??? test mdr11707z
***molts executive versions 790524 790518 on 790523 at 12.55
**0(mdr11707) short wait, allocation queued
```

```
**0(mdr11707) enter hardware device type? ("?" = help) mt500
**0(mdr11707) start mdr14a-mdrdvr, ttldat 790327, phy./log. id t//04

*** i/o trace ***
idcw: - 000000704001
dcw list:
000245000001

*** i/o trace ***
idcw: - 120000724002
dcw list:
000245000001 160000704000 000246000073

*** i/o trace ***
idcw: - 310000700200
dcw list:
000245000001

*** i/o trace ***
idcw: - 200000704000
dcw list:
000245000001
***molts executive version 790518 off 790523 at 12.55 p.t. 3724
```

Control Mnemonics (prefixed with ".")

NOTE: Only one control mnemonic (.option) can be input for any single "enter options message" and it is processed only if found at the beginning of the option string.

.go return to the test page where interrupted. If the skip (s), jump (txxx), resequence (.rseq), or sequence (.seqt) option is specified, the next test sequencing done.

.opt an enter options message is output.

.pr2
.prt forthcoming test page error messages are appended to a uniquely named stream file, which is printed on test termination, test wrap-up, or acceptance of the .typ control mnemonic. (Refer to the "Sample output" in the .typ mnemonic description below.)

.tal a message with a tally of errors is output. Option p or r must be set. The message includes the information for the current pass or cycle, or both (depending on the state of p and r). The error tallies are reset for the pass or cycle, or both when reported.

Sample output:

```
??? test mopc117o
**0(mpc117) t1,r enter options: .tal
**0(mpc117) end cycle 2
t1,r enter options:

??? test modr01602o
**0(mdr01602) t1,p,r  enter options: .tal
**0(mdr01602) end pass  0
                    and cycle 1
t1,p,r  enter options:
```

.test e the test page is forcibly terminated.

Sample output:

```
t0 enter options: .test e
**0(mpc016) forced term 1
```

```
          .test e request received
          **0(mpc016): molts loading mtp601 firmware m601 rev. h1
          ***molts executive version 790518 off 790523 at 12.90 p.t. 4095
```

.test w MOLTS is wrapped up.

Sample output:

```
          **0(mdr01603) t0 enter options: .test w
          **0(mdr01603) forced term 1
          molt wrapping up
          ***molts executive wrap-up code  td3  p.t.  13430
            ic 010201  ir 100000  ba 000000  er 000  ar 000000632403
              qr 662020202020 tr 00070747
            x0 004302  x1 007110  x2 022104  x3 022000  x4 020430  x5 000000
              x6 000000 x7 776377
          ***molts executive wrap-up code  td3  p.t.  13430
```

.typ forthcoming test page error messages are output on the terminal.

Sample output:

```
          t1,d  enter options: .pr2
          **0(mdr01602) t1,d  enter options: z
          ??? test modr01602o
          **0(mdr01602) t1,z  enter options: .typ
          molts dump file >udd>HCSRD>CSRD>!BBBJJNDGkWLgQN.molt.dump has
            been queued for printing
          **0(mdr01602) t1,z  enter options:
```

.wait the test page is put in a wait condition.

MOLTS MESSAGES

MOLTS Informative Messages

MOLTS LOG ON MESSAGE

A message is displayed on the terminal when MOLTS is called initially at command level. (Refer to the example included under "MOLTS Operating Instructions" earlier in this section.)

MOLTS LOG OUT MESSAGE

The following message is displayed on the terminal when MOLTS comes to an orderly (not forced or error caused) conclusion of testing:

***molts executive version xxxxxx off yyyyyy at zz.zzz p.t. aaaaa

where p.t. aaaaa is the processor time used in milliseconds.

MOLTS WRAPUP MESSAGE

A message is displayed on the terminal when MOLTS terminates due to a test?w or test?mw request, or a ".test w" option. (Refer to previously described requests and options for examples of the message.)

MOLTS ABORT MESSAGE

A message is displayed on the terminal when an internal error is detected by MOLTS, or MOLTS is given a fatal error code from some Multics routine call. Because these are system errors that must be analyzed by a system programmer, they are not explained.

WAIT FOR DEVICE MESSAGE

The following message is displayed on the terminal when intervention is required by system operations personnel (e.g., the computer operator is notified to mount a scratch tape on handler dd):

```
**0(iccdd) short wait, allocation queued
```

DEVICE BUSY MESSAGE

When a test request is issued to a device that is already in use by another Multics process, a message similar to the following is displayed. The allocation is completed when the requested device is released by the other process.

```
**0(mdr01804):  device busy, allocation queued
```

START TEST EXECUTION MESSAGE

A message is displayed on the terminal when a page starts execution. (Refer to MDR and ITR Examples included under "New Test Request" earlier in this section.)

TEST TERMINATION MESSAGE

A message is displayed on the terminal when execution of a test page terminates. (Refer to MDR and ITR Examples included under "New Test Request.")

END CYCLE MESSAGE

A message is displayed on the terminal when a normal test page termination occurs and the r option is on. (Refer to the example included with the r option under "Standard Test Page Options" earlier in this section.)

END PASS MESSAGE

A message is displayed on the terminal when the next test sequencing detects a back jump in the test sequence and the p option is on. (Refer to the example included with the .tal control mnemonic under "Standard Test Page Options".)

END TEST MESSAGE

A message is displayed on the terminal when a test ends normally and the inform user (i) option is on. (Refer to the example included with the i option under "Standard Test Page Options.")

RELOAD FIRMWARE MESSAGE

Reloading operational firmware in the MPC being tested terminates ITRs. The following message is displayed on the terminal and the system console to document the firmware revision in use:

```
**0(mpc117):  molts has loaded mtc500 firmware m500 rev.  t1
```

MULTIPLE CATALOG MESSAGE

Periodically because of differences in the revision status of MPCs and noncompatibility of ITRs and firmware for these MPCs, an IFAD tape is released to the field that has two (or more) ITR/firmware files for the same MPC type. The IFAD type loader, load_tandd_library, builds two separate catalog records for these ITR/firmware files. Since MOLTS and the Multics online driver have no pre-knowledge of which of these catalogs are to be used, this decision is left to the user. The following is an example of the multiple catalog message with user's response:

```
***enter "polts", "molts", "isolts", "colts", "quit", or "msg1"


??? molts
??? test mpc019
***molts executive versions 791220 791113 on 791220 at 14.99
**0(mpc019) start tdc16a-itrdrv, ttldat 790926, phy./log. id t//04
**0(mpc019): Multiple catalog files (2) for mtp610 itr catalog.
Indicate which one is to be used by answering yes to
one of the following catalog entrys:
mtp610 itr catalog, firmware rev c2, file # 4, - no
mtp610 itr catalog, firmware rev b1, file # 5, - yes
**0(mpc019) normal term 1
**0(mpc019): molts has loaded mtp610 firmware m610 rev. b1
***molts executive version 79113 off 791220 at 15.03 p.t. 46136
```

MTR User Communication Messages

LOG ON MESSAGE

```
**0(mmt02833)
mtr5 is at your service for PRMFL emergency track repair
```

1. User must specify seek address (sa) for data failure.
2. Primary application is to repair "1" track with unrecoverable data (data restore required).
3. Secondary application is to repair marginal track(s) detected in search range (no data restore required).

INPUT DATA REQUEST MESSAGE

```
**0(mmt02833)
enter "sa" (12 char.'s) for bad track - xxxxx0001370
```

NO BAD TRACK DETECTED MESSAGE

```
**0(mmt02833)
***** mtr5 - search for bad track results *****
all tracks in search range are error free
specified seek addr. = 0001370
```

NOTE: mtr5 search range = 0000000 - 0002757
Suspect error was not confirmed. If error is marginal, initiate a retry. Otherwise, verify device No., seek address (sa) and/or label has been properly initialized (flagged tracks withdrawn).

MARGINAL TRACK REPAIR MESSAGE

```
**0(mmt02834)
***** mtr5 - skip generator report *****
reclaimed - repaired data field
418/00
**0(mmt02834)
***** mtr5 - marginal track repair report *****
following info defines the final state of track repair for marginal
tracks (no data restore req)
std addr    alt addr
********    ********

418/00    = rls'd
```

UNRECOVERABLE DATA REPAIR MESSAGE

```
**0(mmt02815)
***** mtr5 - skip generator report *****
defective - 3 skips on track
101/18
**0(mmt02815)
***** mtr5 - track repair summary report *****
final condition of track with unrecoverable data
std addr    alt addt
********    ********

101/18    = 840/07
**0(mmt02815)
***** mtr5 - locked sector summary report *****
following list of sector addr.'s will remain locked until a data restore has
been performed.

sector       consecutive
address      sector count
*******      ************

2044070      (01)
```

LOG ON MESSAGE

```
**0(mmt02833)
rmc5 is at your service for PRMFL emergency track repair
```

1. User must specify "sa" for data failure.
2. Primary application is to repair "1" track with unrecoverable data (data restore req.).
3. Secondary application is to repair marginal track(s) detected in search range (no data restore req.).

INPUT DATA REQUEST MESSAGE

```
**0(mmt02833)
enter "sa" (12 char.'s) for bad track - xxxxx0001370
```

NO BAD TRACK DETECTED MESSAGE

```
**0(mmt02833)
***** rmc5 - search for bad track results *****
all tracks in search range are error free
specified seek addr. = 0001370
```

NOTE: rmc5 search range = 0000000 - 0002757
Suspect error was not confirmed. If error is marginal, initiate a retry. Otherwise, verify device No., seek address (sa) and/or label has been properly initialized (flagged tracks withdrawn).

MARGINAL TRACK REPAIR MESSAGE

```
**0(mmt02834)
***** rmc5 - skip generator report *****
reclaimed - repaired data field
418/00
**0(mmt02834)
***** rmc5 - marginal track repair report *****
following info defines the final state of track repair for marginal
tracks (no data restore req)
std addr   alt addr
********   ********

418/00   = rls'd
```

UNRECOVERABLE DATA REPAIR MESSAGE

```
**0(mmt02815)
***** rmc5 - skip generator report *****
defective - 3 skips on track
101/18
**0(mmt02815)
***** rmc5 - track repair summary report *****
final condition of track with unrecoverable data
```

```
std addr    alt addt
********    ********

101/18    = 840/07
**0(mmt02815)
***** rmc5 - locked sector summary report *****
following list of sector addr.'s will remain
locked until a data restore has been performed.

sector       consecutive
address      sector count
*******      ************

2044070      (01)
```

MOLTS MDR Error Message

If a channel or IOM status error is detected while running MDRs, a standard message is displayed in the form:

```
**0(mdr11707)
*** mdr driver for mth-500 rev a ***
t14  test-001 iom-1 ch.-17 dev-07
op code-31
subtest - 20
2023 is the fault dictionary entry
**********************************
```

This is an example of an MDR error message that has a corresponding fault dictionary entry. The program may or may not continue execution at this point depending on the particular MDR involved. Refer to the CSR/T&D documentation as regards individual MDR operation.

The following is an example of an MDR that encountered a system status error (power off) that is outside of the realm of MDR testing (terminating, therefore, the MDR).

```
**0(mdr11707)
mpc status poff trying to execute mdr t14b01
status word was 600000000000
**0(mdr11707) normal term 1
```

MOLTS ITR Error Message

Whenever an error is detected running an ITR, either a status error or a timeout (waiting for a special interrupt after sending the ITR to the MPC), an error message similar to the following is displayed.

```
??? test mpc117
***molts executive versions 790524 790518 on 790523 at 12.97
**0(mpc117) start tdc14a-itrdvr, ttldat 790509, phy./log. id t//04
**0(mpc117)
the test overlay named *csltb1* for mtc500
has encountered an error please investigate
**0(mpc117)
you have not made a complete error free pass of
all the itrs. type 'wait' to leave channel down, 'r' to recycle,
eom to restore firmware.
(request repeats in 1 minute if 'wait' is used)? wait
**0(mpc117)
you have not made a complete error free pass of
all the itrs. type 'wait' to leave channel down, 'r' to recycle,
eom to restore firmware.
```

```
(request repeats in 1 minute if 'wait' is used)? r
**0(mpc117) end cycle 1
**0(mpc117) end cycle 2
```

where *csltb1* (in line 5 of this example) is actually the test overlay named cslt rev b1.

MOLTS MTR Error Message

The following message is displayed on the terminal when a transmission parity error is encountered while running MTR.

```
**0(mmt02831)
*** major/sub status error ***
data alert -
parity -
major and sub status -           = 03-02
physical addr. at time of failure = 022/39
```

where

```
major status 03      -- data alert
substatus    02      -- transmission parity alert
physical address 022 -- cylinder
                  39 -- head
```

For additional error codes, refer to the CSR MPC Subsystem Handbook.

MOLTS Input Error Message

The following message is displayed on the terminal when an input following "test" contains an error:

```
***molts executive (erroneous data) invalid input
(reason)
```

where (reason) is one of the following:

channel not assignable
: The channel number input is not configured on the system.

device busy, release and retry
: The device is in use by the operating system; release the device from the system and retry.

device is invalid for this type subsystem
: MOLTS expects an input device name other than what it received. Prior to processing the device name, MOLTS would have issued a list of valid choices if the operator responded with either a question mark (?) or an unexpected name.

invalid channel number
: The channel number input was nondecimal.

invalid iom number
: The IOM number input was nonoctal or greater than the largest IOM configured on the system.

must be bootable psia channel
: This message results when a request is made to test an MPC and the channel number entered is a PSIA channel that cannot be booted.

must be logical channel 0 on psia
A test request was received to run MDR testing through an MPC over some channel other than logical channel zero of the MPC. Logical channel zero is the lowest number IOM channel on a PSIA.

no such test page exists
A Test request was made for a channel for which a test does not exist.

unknown request for this executive
The input does not match any of the valid inputs to MOLTS.

MOLTS Option Error Messages

The following message is displayed when options input to the test are processed and some error is detected in those options. A request to enter new options is always appended to this error message.

```
**0(iccddc) illegal option:
(reason)
(current options) enter options:
```

where (reason) is any of the following:

illegal control mnemonic (.option) encountered
A request was received for an option that was not recognized by the option processor.

only a "1" is allowed following "e"
A character other than a "1" was used to try to restore transient error recovery retry count to test page standard.

pass or recycle must be set to output error tallies
A .tal option was input but neither p nor r was set ON.

'x' is an illegal option character
'x' is some character other than b, e, h, i, l, n, o, p, r, s, t, or u. An illegal option character has been detected by the option processor.

'.' must be the first option character
A control mnemonic was found embedded in an option string.

Sample output:

```
**0(mdr01602) t1,d  enter options: c
**0(mdr01602) illegal option:
'c' is an illegal option character
t1,d  enter options:
```

MOLTS MDC Error Messages

The different types of error messages that the MDC program provides are described below.

INITIALIZATION CORRECTABLE ERROR MESSAGES

During initialization, the program does a quick scan of the first status return looking for any gross errors that may be correctable. A brief warning message is output suggesting corrective action be taken, and a one-minute countdown begins. This provides a reprieve to correct the problem within a one-minute

window without aborting the program. The two possible situations and accompanying messages are:

1. No power on device.

```
**0(mdc01004 )
dev in standby.  you have 1 minute to make dev ready.
```

2. Main power switch is OFF, or maintenance panel offline switch is OFFLINE.

```
**t(mdc00805  )
dev offline.  dev on/off sw is off or dev is nonexistent.  1 minute
to correct.
```

UNCORRECTABLE ERROR MESSAGES

There is a remote possibility that certain uncorrectable errors might occur, either during initialization or actual disk exercising. An appropriate error message is output and the program terminated. The following are a few typical examples:

1. Single bit errors.

```
**t(mdc01001  ) dev type = 181      test data = 666666666666
wt/ver fail address---0612699      c741/h10/s36
*** sngl bit stat err ***
status word 1 = 451006000000
lost interrupt
```

2. IOM errors.

```
**t(mdc00813  ) dev type = 190a
read fail address---0569575      c749/h08/s15
*** iom status error ***
chan/central status = 70
parity error, io bus, data to channel
```

TEST EXECUTION ERROR MESSAGES

These messages describe how well the disk is working. Once initialization and parameter setup is completed by test 01, test 03 begins disk exercising. The disk drive is accessed with either a seek-write or a seek-read, and the status is checked after each IO. Any abnormal status results in an error message. Each bad disk IO is retried up to three times, or until a good status results, whichever occurs first. In either case, an error message results giving all pertinent information, including the number of retries. This program overrides MPC retry. Under actual operating conditions, a disk MPC may retry a faulty IO up to a dozen times. However, a solid error is reasonably certain after three retries on a disk. One or two retries may be considered questionable, but it is within tolerance. In special cases of a maj/sub status of 02/10 (attention/dev fault), the extended status with English translation is tacked on the message.

NOTE: dev attn (02) is considered serious, and no retry is attempted if encountered.

Following are several typical error messages:

```
**t(mdc00807  ) dev type = 451 t3 test data = 666666666666
wt/ver fail address---0612701      c806/h03/s21
status------432600000000      xxxxxxxxxxxx
maj/sub = 13/26                retried 1 times
```

```
mpc device data alert -
sync byte error -

**t(mdc00803  ) dev type = 451   t1
read  fail address---0484029     c636/h16/s29
status------436000000000      xxxxxxxxxxxx
*maj/sub = 03/60        retried 2 times
data alert -
cyclic check & compare alert -

**t(mdc00804  ) dev type = 451 t1
read   fail address---0398223               c523/h18/s23
status------532600000000      xxxxxxxxxxxx
*maj/sub = 13/26        retried 3 times
mpc device data alert -
sync byte error -
```

Following is the extended status type message for all attn/dev faults.

```
**t(mdc00807  )  dev type =451  t3     test  data = 666666666666
wt/ver fail address---0000010        c000/h00/s10
status------421000000000      xxxxxxxxxxxx
*maj/sub = 02/10
attention -
device fault -
*extnd status = ed1b80200000
      mask    = fffffffffffff
dev reserved
dev seized
dev in standby
dli fault
dev protected
dev in diag mode
invalid cmd seq
state violation
trnsf timing err
data parity err
loss of write current
```

If on fault retry, the returned status differs from a previous attempt, all status is displayed. For example:

```
**t(mdc00804  ) dev type = 451  t1
read  fail address---0398223
status------532600000000      xxxxxxxxxxxx
*maj/sub = 13/26        retried 3 times
mpc device data alert
sync byte error
previous io at this seek addr yielded the following status...
1st io = 13/26
1st rtry = 03/60
2nd rtry = 03/60
```

MOLTS MTC Error Messages

The four types of error messages that the MTC program provides are described below.

IOM error -- this error outputs any time an IOM error condition exists. Bits 18 to 23 of IOM status word 1 are checked.

Power off/offline error -- this error outputs if the device being tested is offline or powered off. Bit one of IOM status word 1 is checked.

Maj/sub status error -- bits 2 to 11 of IOM status word 1 are checked. The message contains octal and English decode of maj/sub status bits. Octal value of IOM status words 1 and 2. Hex value of the 26 extended status bytes and the English translation of any extended status fault bits set. The failing record number is also output.

Compare error -- a data compare is done on all read commands. Should an error be found, this message is output.

MOLTS MTG Error Messages

The five types of error messages that the MTG program provides are described below.

IOM error -- this error outputs any time an IOM error condition exists. Bits 18 to 23 of IOM status word 1 are checked.

Power off/offline error -- this error outputs if the device being tested is offline or powered off. Bit 1 of IOM status word 1 is checked.

Maj/sub status error -- bits 2 to 11 of IOM status word 1 are checked. The message contains octal and English decode of maj/sub status bits. Octal value of IOM status words 1 and 2. Hex value of the 26 extended status bytes and the English translation of any extended status fault bits set.

Compare error -- a data compare is done on all read commands. Should an error be found, this message is output. Contained in the message is the record length and the location, in the read buffer, of the first word in error. "Data was" and "data should be" are also output.

Position error -- a tape position check is done after all forward or backspace commands. Should an error exist, this message is output. A check is made of the tally residue in status word 1 and compared to the tally in the test page, and contains "residue was" and "residue should be" information.

SECTION 5

POLTS EXECUTIVE

POLTS provides the online capability of functional testing and online troubleshooting of malfunctioning peripheral devices. This is accomplished without unduly interfering with the overall system operation.

A secondary function is that during slack periods of user operation, or peripheral availability, equipment testing can be accomplished. This allows the normally scheduled preventive maintenance time to be utilized more effectively in the actual maintenance of equipment rather than the running of tests.

FUNCTIONAL CAPABILITIES

POLTS is a test subsystem that operates under the TOLTS executive program. It provides a language interpreter and I/O driver for the functional peripheral tests (called test pages) that are written in Test and Diagnostic Language (TDL).

Hardware Tested

Currently, POLTS is capable of performing online tests on the following peripheral devices:

DEVICE DESCRIPTION	DEVICE TYPE	TEST PAGE IDENTIFICATION	
		PAGE NAME	FILE HEADER
MTS400 7-Track Magnetic Tape	400	asa7cb	td11ca
MTS400 9-Track Magnetic Tape	400	asa9a,asa9b	td12ca,td12cb
MTS500 7-Track Magnetic Tape	500	t57a	td13ca
MTS500 9-Track Magnetic Tape	500	t59a,t59b,t59c	td14ca,td14cb,td14cc
MTP601 7-Track	410/600/500	t67a	td15ca,b
MTP601 9-Track	410/600/500	t69a	td16ca,b,c,d
DSS191	DSU191	d191	td62ca,b
DSS451	DSU451	d451	td65ca,b,c
DSS500	DSU500	d451	td65ca,b,c
CR20 Card Reader	20c	cr20cb	td20ca
CR21 Card Reader	201	cr21	td21ca
CRU301 Card Reader	301	cr34	td34ca
CPZ201 Card Reader	200/201	cp23cc	td23ca
CPU300 Card Punch	300/301	cp32cc	td32ca
PRT202 Printer	201/202	print	td26ca
PRT300/301 Printer	300	p300ca,p300	td25ca,td25cb
PRT303 Printer	303	p303ca,p303	td35ca,td35cb
PRU1200/1600 Printer (136 lines)	401/402	p401/2	td24ca,td24cb
PRU1200/1600 Printer (160 lines)	401/402	p401/2	te24ca,te24cb
System Console (Selectric)	IBM	consol	td30ca
Entry Model Console (EMC)	SCC	scc	td31ca

Documentation for these test pages is located in the T&D Microfiche Documentation Box.

WARNING: Site maintainers should be aware that if the Crash on Console Recovery Failure option has been selected, either by presence of the CCRF parameter in the config_file or the usage of the "set_system_console -crash" command, and the bootload console is deconfigured via the set_system_console command, Multics will crash with the message:

```
ocdcm_ (reconfigure): Bootload console deconfigured
   with CCRF set.
```

This command is documented in the MAM--System, Order No. AK50.

POLTS OPERATING INSTRUCTIONS

POLTS is entered by answering the TOLTS query:

```
***enter "polts", "molts", "isolts", "colts", "quit", or msg"
???
```

with the keyword "polts". After a slight pause for the core image loading of the POLTS slave executive, an input prompt/request for input (???) is displayed. When the prompt is responded to with a valid test request, POLTS responds with a greeting message.

```
??? test x...
***polts executive versions wwwwww xxxxxx on yyyyyy at zz.zzz
```

where

1. test x...
   can be any of the requests listed in the following paragraphs. The maximum character length of any request is 11 characters (including options).

2. wwwwww
   is the Multics interface module version date expressed as yymmdd.

3. xxxxxx
   is the slave POLTS version date expressed as yymmdd.

4. yyyyyy
   is the current date expressed as yymmdd.

5. zz.zzz
   is the time in hours and thousandths of hours.

Any time a quit signal is issued, by pressing the appropriate key on the terminal (e.g., QUIT, BRK, ATTN), POLTS responds by displaying "???" and pauses for entry of a new msg (refer to "Tolts Operating Instructions" in Section 2) or test request. Up to eight POLTS requests can be active at any given time.

Test pcd Request

The test pcd request can be entered in three forms:

1. "pcd" displays the current peripheral configuration

2. "pcd xxx" where xxx identifies a specific subsystem device type display (e.g., iom, dsk, opc)

3. "pcd xxxx" where xxxx identifies a particular device display (e.g., ioma, dske, opcc).

Examples of each follow.

```
??? test pcd

polts configuration:

iom a a nsa iom on scu port 0 is online
iom b a nsa iom on scu port 1 is online
dska 451 16 units;  starting with device no. 1
     020xx          primary channel of 4 logical channels on mpc card mspa
     026xx          secondary channel of 4 logical channels on mpc card mspa
     124xx          secondary channel of 4 logical channels on mpc card mspb
     122xx          secondary channel of 4 logical channels on mpc card mspb
dskb 451 16 units;  starting with device no. 17
     120xx          primary channel of 4 logical channels on mpc card mspb
     126xx          secondary channel of 4 logical channels on mpc card mspb
     024xx          secondary channel of 4 logical channels on mpc card mspa
     022xx          secondary channel of 4 logical channels on mpc card mspa
dskc 501 16 units;  starting with device no. 1
     028xx          primary channel of 4 logical channels on mpc card mspc
     130xx          secondary channel of 4 logical channels on mpc card mspd
dskd 501 16 units;  starting with device no. 17
     128xx          primary channel of 4 logical channels on mpc card mspd
     030xx          secondary channel of 4 logical channels on mpc card mspc
dske 451  8 units;  starting with device no. 1
     132xx          primary channel of 4 logical channels on mpc card mspe
     134xx          secondary channel of 4 logical channels on mpc card mspe
dskf 501 16 units;  starting with device no. 1
     036xx          primary channel of 4 logical channels on mpc card mspf
     138xx          secondary channel of 4 logical channels on mpc card mspg
dskg 501 16 units;  starting with device no. 17
     136xx          primary channel of 4 logical channels on mpc card mspg
     038xx          secondary channel of 4 logical channels on mpc card mspf
tapa 630  2 units;  starting with device no. 1
     610  6 units;  starting with device no. 3
     630  2 units;  starting with device no. 9
     016xx          primary channel of 2 logical channels on mpc card mtpa
prtd 00801 model 1600 prt with 160 columns and a 600 print belt
     008xx          primary channel of 5 logical channels on mpc card urpa
prta 00901 model 1200 prt with 136 columns and a 600 print belt
     009xx          secondary channel of 5 logical channels on mpc card urpa
prtb 01001 model 1600 prt with 136 columns and a 600 print belt
     010xx          secondary channel of 5 logical channels on mpc card urpa
rdra 01101 model 500 rdr
     011xx          secondary channel of 5 logical channels on mpc card urpa
puna 01201 model 300 pun
     012xx          secondary channel of 5 logical channels on mpc card urpa
prtc 10801 model 1201 prt with 136 columns and a 600 print belt
     108xx          primary channel of 2 logical channels on mpc card urpb
prte 10901 model 1600 prt with 136 columns and a 600 print belt
     109xx          secondary channel of 2 logical channels on mpc card urpb
punb 11001 model 300 pun
     110xx          primary channel of 2 logical channels on mpc card urpc
rdrb 11101 model 500 rdr
     111xx          secondary channel of 2 logical channels on mpc card urpc
opc  01401 model 6601 LCC is the system console
     014xx           special purpose chan
opca 11201 model 6601 opc is the alternate system console
     112xx           special purpose chan
opcb 11301 model 6601 opc is available for test
```

```
     113xx          special purpose chan
opcc 11601 model 6601 opc is available for test
     116xx          special purpose chan
```

where:

1. prta
   is the Multics device name.

2. 00901
   is the IOM/channel/device in decimal (iccdd).

3. 1200
   is the model (in this case -- 1200 lpm).

4. 136
   is the number of print columns.

5. 600
   is the print belt part number (or cartridge number).

```
??? test pcd iom

iom configuration:

iom a a nsa iom on scu port 0 is online
iom b a nsa iom on scu port 1 is online

??? test pcd ioma

iom configuration:

iom a a nsa iom on scu port 0 is online

??? test pcd dsk

dsk configuration:

dska 451 16 units;  starting with device no. 1
     020xx          primary channel of 4 logical channels on mpc card mspa
     026xx          secondary channel of 4 logical channels on mpc card mspa
     124xx          secondary channel of 4 logical channels on mpc card mspb
     122xx          secondary channel of 4 logical channels on mpc card mspb
dskb 451 16 units;  starting with device no. 17
     120xx          primary channel of 4 logical channels on mpc card mspb
     126xx          secondary channel of 4 logical channels on mpc card mspb
     024xx          secondary channel of 4 logical channels on mpc card mspa
     022xx          secondary channel of 4 logical channels on mpc card mspa
dskc 501 16 units;  starting with device no. 1
     028xx          primary channel of 4 logical channels on mpc card mspc
     130xx          secondary channel of 4 logical channels on mpc card mspd
dskd 501 16 units;  starting with device no. 17
     128xx          primary channel of 4 logical channels on mpc card mspd
     030xx          secondary channel of 4 logical channels on mpc card mspc
dske 451  8 units;  starting with device no. 1
     132xx          primary channel of 4 logical channels on mpc card mspe
     134xx          secondary channel of 4 logical channels on mpc card mspe
dskf 501 16 units;  starting with device no. 1
     036xx          primary channel of 4 logical channels on mpc card mspf
     138xx          secondary channel of 4 logical channels on mpc card mspg
dskg 501 16 units;  starting with device no. 17
     136xx          primary channel of 4 logical channels on mpc card mspg
     038xx          secondary channel of 4 logical channels on mpc card mspf

??? test pcd dske
```

```
dsk configuration:

dske 451  8 units;  starting with device no. 1
     132xx          primary channel of 4 logical channels on mpc card mspe
     134xx          secondary channel of 4 logical channels on mpc card mspe

??? test pcd opc

opc configuration:

opc  01401 model 6601 LCC is the system console
     014xx           special purpose chan
opca 11201 model 6601 opc is the alternate system console
     112xx           special purpose chan
opcb 11301 model 6601 opc is available for test
     113xx           special purpose chan
opcc 11601 model 6601 opc is available for test
     116xx           special purpose chan

??? test pcd opcc

opc configuration:

opcc 11601 model 6601 opc
     116xx           special purpose chan
```

Test List Request

To list all of the active test pages in POLTS, enter one of the following requests:

```
test plstal
```

-or-

```
test lstal
```

Sample output:

```
??? test lstal
polts lstal:
in execution:
**0(01601c) t//04
```

If a waiting allocation message is present, it indicates that the device cannot be assigned to POLTS at this time because it is presently in use by some other process.

Test Wrapup Request

To wrapup (stop) all POLTS testing, enter one of the following requests:

```
test pw
```

-or-

```
test w
```

Sample output:

```
??? test w
***polts executive wrapup code  td2  p.t.  4130
```

```
  ic 002007  ir 100000  ba 000000  er 000  ar 000000632402  qr 777777770077
    tr 00070617
  x0 000300  x1 202001  x2 000116  x3 000036  x4 000004  x5 000000  x6 000000
    x7 000152
***polts executive wrapup code  td2  p.t.  4130
```

If the .pr2 option was in force the register banner information plus an octal dump of the POLTS segment would be appended to the print file to be dprinted later.

New Test Request

To start execution of a new test, enter the following request:

```
test piccddooooo
```

where

1. iccdd
   is the IOM/channel/device.

2. ooooo
   is the new option character set.

Sample output (using various test page options):

```
??? test p01601ih
***polts executive versions 790524 790518 on 790529 at 14.41
**a(01601c) short wait, allocation queued
**0(01601c) start td16ca - t69a, ttldat 781108, phy./log. id t//04
**0(01601c),h,i end t1,next t2  enter options: e05
**0(01601c),e,h,i end t2,next t3  enter options: l
**0(01601c),e,h,i,l end t3,next t3  enter options:
**0(01601c),e,h,i,l end t3,next t3  enter options: nl
**0(01601c),e,h,i end t3,next t4  enter options: pr
**0(01601c),e,h,i,p,r end t4,next t5  enter options: nh
**0(01601c),e,i,p,r end t5,next t6
     .
     .
     .
**0(01601c) start td16cd - t69d, ttldat 781108, phy./log. id t//04
**0(01601c),e,i,p,r end t57,next t59
**0(01601c) end cycle 1: 3 satus and 0 data errors
**0(01601c) start td16ca - t69a, ttldat 781108, phy./log. id t//04
**0(01601c),e,i,p,r end t1,next t2
**0(01601c),e,i,p,r end t2,next t3
??? test po01601ni
??? test po01601o
**0(01601c) t20,e,p,r  enter options: .tal
**0(01601c) for pass  0: 0 status and 0 data errors
            and cycle 1: 0 status and 0 data errors
t20,e,p,r  enter options: .go
**0(01601c) start td16cb - t69b, ttldat 790126, phy./log. id t//04
**0(01601c) start td16cc - t69c, ttldat 781108, phy./log. id t//04
```

Test Communications Request

To change the option character set presently in force or to enter a single control mnemonic for a request presently in execution, enter the following request:

```
test poiccddoooo
```

where o... is the new option character set (the "o" option interrupts a test page).

Sample output:

```
??? test po01601o
**0(01601c) t19 enter options: .pr2
**0(01601c) t19 enter options: z
??? test po01601o
**0(01601c) t19,z enter options: .typ
polts dump file >udd>HFED>FED!BBBJJJNQDkkXqDh.polt.dump has
  been queued for printing
```

Test End Request

To terminate execution of a test presently in execution, enter the following request:

```
test peiccdd
```

Sample output:

```
??? test pe01601
**0(01601c) forced term 2: 3 status and 0 data errors
                           transient errors: 0 read and 1 write
                           channel time: 52808399
test e request received
***polts executive version 790518 off 790529 at 14.99 p.t. 956552
```

Standard Test Page Options

The following error and control options can be entered in response to an enter options message (test communications request) or designated in the option string of a new test request.

OPTION CHARACTERS AND DESCRIPTION

b bypass error message output. Bypass overrides a pass or cycle message unless halt is set. Halt forces these messages to be displayed (overriding the bypass option).

exx output transient error message. If numeric characters xx follow, that value is used to override the test page standard value for the number of retries to be made. If a "-1" is typed, then the test standard is restored. This control of retry count is independent of the turn "on" or "off" capabilities of the e or ne option.

h halt for input of options following error messages, end test messages, end pass messages, and end cycle messages.

i inform the user of each normal "end test." If halt (h) is also specified, then an "enter options" is appended to the end test message. The end test message is overridden if the next segment is being called, an end pass message is being output, or if a cycle ends.

l loop on current test (cannot loop on test 0).

n negate the following option character (valid only if preceding b, e, h, i, l, p, r, t, x, or the z options).

o go to "enter options" following processing of the complete option string containing the "o."

p issue an end pass message any time a back jump is detected by the next test sequencing. If halt (h) is also specified, then an "enter options" is appended to the end pass message. The error tallies for the current pass are reset when an "end pass" is reported or when "p" is being turned on.

r issue an end cycle message any time a normal test page termination occurs and cycle back to the first test in the current sequence. If halt (h) is also specified, then an "enter options" is appended to the end cycle message. The error tallies for both the current pass and current cycle are reset when "end cycle" is reported. The error tallies for the cycle are reset whenever "r" is being turned on.

s unconditionally skip to the next test.

txxx if turned on (no preceding negate), then unconditionally jump to the first occurrence of the test in the current sequence. The test number xxx must follow and must be nonzero, and can consist of one, two, or three numeric characters. For segmented test pages, a value outside of the current segment causes a jump to the corresponding segment without further processing of the current option string. If turned off (ntxxx), then the test number must be in the current segment and sequence, and not the forced termination test number.

u unload option. Facilitates the use of a tape or disk drive without requiring operator intervention. The media currently mounted on the tape or disk drive is used without operator authentication or mounting required. This option is only available with the initial option string.

x enable the extended status portion of the standard error message for MPC test pages. This option is initially forced "on" automatically. The user should utilize the nx option characters if the extended status message is not desired. The message can subsequently be enabled by utilizing the x option character.

z trace I/O setup. A message is output for each test I/O issued.

Sample output:

```
??? test po01601o
**0(01601c) t17,i  enter options: niz

*** i/o trace ***
idcw: - 150100700000
dcw list:
000245000016

*** i/o trace ***
idcw: - 540100700201
dcw list:
000245000001
     .
     .
     .
     .
```

Control Mnemonics (prefixed with ".")

NOTE: Only one control mnemonic (.option) can be input for any single "enter options message" and it is processed only if found at the beginning of the option string.

.go return to the test page where interrupted. If the skip (s), jump (txxx), ".rseq", or ".seqt" option is specified, the next test sequencing

is done. If any TDL line has been changed with .seq or .seqa, the current test is restarted.

.man force the test page to manual mode. This allows the use of .seqxx, .seqaxx, and .stnxx; however, .stnxx or .strxx are not required to start execution after manual input. To modify a test or subroutine (e.g., restrict the address range to be tested) and then run all tests, use the .opt option to get out of input mode (line xx.) and then invoke the txxx option to enter the desired test.

Sample output:

```
??? test p01601o
***polts executive versions 790331 790412 on 790504 at 09.24
**a(01601c) short wait, allocation queued
**0(01601c) start td16ca - t69a, ttldat 781108, phy./log. id t//04
**0(01601c) t0 enter options: .man
**0(016011) t0 enter options: .seq20
**0(016011)
line 20. chg,11024,d777777777777,wtb,bks,rtb,lp20.f04.500
**0(016011)
line 21. .str20
**0(016011) invalid tdl instruction, line 20, field 01, 11024,:
last word of dcw would be outside of the write area
t1 enter options: .seq20
**0(016011)
line 20. chg,1310,d777777777777,wtb,bks,rtb,lp20.f04.500,o
**0(016011)
line 21. .str20
t1 enter options: lr
??? test po01601o
**0(016011) t1,1,r  enter options: .tal
**0(016011) for cycle 0: 0 status and 0 data errors
t1,1,r  enter options: .ovci
**0(01601c) start td16ca - t69a, ttldat 781108, phy./log. id t//04
**0(01601c),i end t1,next t2
     .
     .
     .
**0(01601c),i end t19,next t20
??? test pe01601
**0(01601c) forced term 1: 0 status and 0 data errors
                           channel time: 33422303
test e request received
***polts executive version 790412 off 790504 at 09.43 p.t. 204731
```

.ovcooooo overlay this test page with the comprehensive test page and use o... as the initial options. (Refer to the "Sample output" in the .man mnemonic description above.)

.opt an enter options message is output.

.pr2 forthcoming test page error messages are appended to a uniquely named
.prt stream file, which is printed on test termination, test wrapup, or acceptance of the .typ control mnemonic. (Refer to the "Sample output" in the .typ mnemonic description below.)

.rseq resequence tests to their initial (at page call) order.

.seqxx accept TDL input for line xx in a manual test page. A message is output:

```
**x(header)
line xx.
```

where

1. **x(header)
   is the test page header.

2. xx
   is the line to be changed.

The TDL instructions can now be entered for the line. The length of the line must not exceed 54 characters. If the input is valid, then the program issues another "line xx." message, rolling over to line 00 after 99, until a "." is detected as the first character of a line (a control mnemonic).

Any invalid input causes an appropriate illegal options message to be output. (Refer to the "Sample output" in the .man mnemonic description above.)

.seqaxx accept data input for line xx in a manual test page. A message is output:

**x(header)
line xxa where

1. **x(header)
   is the test page header.

2. xx
   is the line to be changed.

The data patterns can now be entered for the data line. The length of the line must not exceed 54 characters. If the input is valid, then the program issues another "line xx." message, rolling over to line 00 after 99, until a "." is detected as the first character of a line (a control mnemonic).

Any invalid input causes an appropriate illegal options message to be output. (Refer to the "Sample output" in the .man mnemonics description above.)

.seqt sequence tests in special order; a message is output:

**x(header) input up to zz test #'s from xxx to yyy in the sequence desired

where

1. **x(header)
   is the test page header.

2. zz
   is the number of tests allowed.

3. xxx
   is the lowest test number allowed.

4. yyy
   is the highest test number allowed.

Test numbers are input in the sequence desired and are separated by commas. A "-" in front of a test number indicates a jump either forward or backward to the first occurrence of that test number in the new sequence. For example, test numbers of 1, 2, 3, 4, -1 indicates running tests in the sequence 1, 2, 3, 4, 1, etc., with a back jump indicated at test 4 (affects the pass (p) option).

Sample output:

```
??? test p01603o
***polts executive versions 790331 790412 on 790502 at 16.12
**a(01603c) short wait, allocation queued
**0(01603c) start td16ca - t69a, ttldat 781108, phy./log. id t//04
**0(01603c) t0 enter options: .seqt 1,3,2,4,5,-1
**0(01603c)
```

```
input up to 30 test #'s from 1 to 31 in sequence desired.
1,3,4,6,2,-1
**0(01603c) t0 enter options: i
**0(01603c) t0 enter options: .go
**0(01603c),i end t1,next t3
**0(01603c),i end t3,next t4
**0(01603c),i end t4,next t6
**0(01603c),i end t6,next t2
**0(01603c),i end t2,next t1
**0(01603c),i end t1,next t3
     .
     .
     .
??? test po01603o
**0(01603c) t6,i  enter options: .rseq
**0(01603c) t6,i  enter options: .go
**0(01603c),i end t1,next t2
**0(01603c),i end t2,next t3
**0(01603c),i end t3,next t4
**0(01603c),i end t4,next t5
     .
     .
     .
     .
```

.stnxx initialize the manual test page and set single test mode starting on line xx.

.strxx set single test mode for a manual test page starting on line xx. (Refer to the "Sample output" in the .man mnemonic description above.)

.tal a message with a tally of errors is output. Option p or r must be set. The message includes the information for the current pass or cycle, or both (depending on the state of p and r). The error tallies are reset for the pass or cycle, or both when reported. (Refer to the "Sample output" in the .man mnemonic description above.)

.test e the test page is forcibly terminated.

.test w POLTS is wrapped up.

.typ forthcoming test page errors are output on the terminal.

Sample output:

```
??? test po01601o
**0(01601c) t19 enter options: .pr2
**0(01601c) t19 enter options: z
??? test po01601o
**0(01601c) t19,z  enter options: .typ
polts dump file >udd>HFED>FED>!BBBJJNQDkkXqDh.polt.dump has
  been queued for printing
```

.wait the test page is put in a wait condition.

POLTS MESSAGES

POLTS Informative Messages

POLTS LOG ON MESSAGE

A message is displayed on the terminal when POLTS is called initially at command level. (Refer to the example included under "POLTS Operating Instructions" earlier in this section.)

POLTS LOG OUT MESSAGE

A message similar to the following is displayed on the terminal when POLTS comes to an orderly (not forced or error caused) conclusion of testing:

```
***polts executive version 790518 off 790529 at 15.35 p.t. 103399
```

POLTS WRAPUP MESSAGE

A message is displayed on the terminal when POLTS terminates due to a test w or test pw request, ".test w" option or an abt TDL instruction. (Refer to previously described requests or the "." option for examples of the message.)

POLTS ABORT MESSAGE

A message is displayed on the terminal when an internal error is detected by POLTS, or POLTS is given a fatal error code from some Multics routine call. Because these are system errors that must be analyzed by a system programmer, they are not explained.

WAIT FOR DEVICE MESSAGE

The following message is displayed on the terminal when intervention is required by system operations personnel (e.g., the computer operator is notified to mount a scratch tape on handler dd):

```
**a(iccddc) short wait, allocation queued
```

DEVICE BUSY MESSAGE

When a test request is issued to a device that is already in use by another Multics process, a message similar to the following example is displayed. The allocation is completed when the requested device is released by the other process.

```
**0(p01804): device busy, allocation queued
```

START TEST EXECUTION MESSAGE

The following message is displayed on the terminal when a page starts execution:

```
**p(iccddc) start cccccc-nnnn, ttldat yymmdd, phy./log.  id t//04
```

where

1. cccccc
   is the POLTS "file header" name (such as td12ca).

2. nnnn
   is the POLTS "page" name (such as asa9a).

3. yymmdd
   is the date (year, month, and day).

TEST TERMINATION MESSAGE

A message is displayed on the terminal when execution of a test page terminates.

Sample output:

```
**0(01603c) normal term 1: 0 status and 0 data errors
                           transient errors: 11 read and 2 write
                           channel time: 52548113
***polts executive version 790412 off 790502 at 14.72 p.t. 638691
```

END CYCLE MESSAGE

A message is displayed on the terminal when a normal test page termination occurs and the r option is on.

Sample output:

```
**0(01601c) for pass 0:  0 status and 0 data errors
            and cycle 1:  0 status and 0 data errors
```

END PASS MESSAGE

A message is displayed on the terminal when the next test sequencing detects a back jump in the test sequence and the p option is on. (Refer to example included with "End Cycle Message" above.)

END TEST MESSAGE

A message is displayed on the terminal every time an individual test ends normally (e.g., end t2, next t3) and the inform user (i) option is on. (Refer to example included with "New Test Request" described earlier.)

POLTS Standard Error Message

The POLTS standard error message is displayed when an error is detected, if the bypass (b) option is not on, and if the nt modifier is not used.

Each line of the error message usually has several alternatives (either some of the fields, or the whole line) and some lines may not be present. The following information gives a detailed breakdown of the error message, line by line (line numbers are shown below for reference purposes, but are not displayed in the actual message).

```
 1   **0(01206c) 42/75b 05-rtb   00/ok 00/ok 00 t/ok ln 040/ok  123456123456
 2   tadw = 000002224520
 3   smb1 010203040506/010203040506 rrc 00/ok 166 data ers d/543210 p/------
 4   (002)012345012345 (003)012345012345 (004)012345012345 (005)012345012345
 5    s/b 010101010101  s/b 010101010101  s/b 010101010101  s/b 010101010101
 6   io#105
 7   possible transient error, retry will be made
 8   (a message supplied by a nsdd.dd instruction)
 9   device extended status in hex
10   (eep message line from tysdd instruction)
11   c,c,c,c, enter options:
```

Line 01:

Always present.

```
**0(01206c) 42/75b 05-rtb   00/ok 00/ok 00 t/ok ln 040/ok  123456123456
                            00/05 00/01    t/i  ln 040/041 pos 004/ok
                                  00/--         ln 040/--- pos 004/---
                                                rec cnt xx pos 004/005
                                                (blanks)   (blanks)
```

where

Field	0	active POLTS test page number is "0."
Field	01206c	IOM/chan/dev (the "c" signifies comprehensive).
Field	42	test number.
Field	75	line number.
Field	b	number of the device I/O on the line.
Field	05	octal op-code.
Field	rtb	mnemonic for the op-code (in this case, read tape binary).
Field	00/ok	major status (was/sb) if ok.
-or-	00/05	if bad.
Field	00/ok	substatus (was/sb) if ok, and if not, ignore substatus.
-or-	00/01	if bad, and if not, ignore substatus.
-or-	00/--	if bad ignore substatus.
Field	00	IOM/channel status.
Field	t/ok	interrupt returned (was/sb) if ok.
-or-	t/i	if bad.
Field	ln 040/ok	last DCW word count if residual DCW is good and if data transfer I/O, and not "nr," and not initiation interrupt.
-or-	ln 040/041	if bad, and not "nr."
-or-	ln 040/---	"nr" and not initiation interrupt.
-or-	rec cnt 01	record count if nondata command.
-or-	(blanks)	if initiation interrupt and data transfer I/O.
Field	123456123456	if first I/O of dual I/O was seek for mass storage.
-or-	pos 004/ok	if "pos" positioning is in effect for mag tape and positioning is good.
-or-	pos 004/---	if "pos" but position cannot be checked due to data error.
-or-	pos 004/005	if "pos" and position error.
-or-	(blanks)	if not seek or "pos" as above.

Line 02:

Disk seek address, if a disk test page and a seek type command (seek/read, seek/write).

tadw = xxxxxxxxxxxx

where x is the octal seek address.

Line 03:

Present if not IOM fault or timeout.

```
smb1 012345671234/ok           rrc 00/01    positioning error
smb1 010203040506/010203040506 rrc 00/ok 166 data ers d/543210 p/------
smb1 010203040506/------------ rrc 00/--    data chks ok 012345012345
initiation interrupt                        data not checked (init)
non-data command                            data not checked (nc)
                                            write data 123456123456
                                            write data xxxxxxxxxxxx
                                            write data character 00
```

where

Field	smb1 012345671234/ok	data xfer, not initiation interrupt, not "nm," and smb1 ok.
-or-	smb1 010102040506/------------	same, but "nm" used.
-or-	smb1 010203040506/010203040507	same, but not "nm," and checks bad.
-or-	initiation interrupt	if initiation interrupt occurred for data transfer command.
-or-	non-data command	if non-data command.
Field	rrc 00/01	residual record count (was/sb) if error, "nr," and not initiation interrupt.
-or-	rrc 00/ok	same, but checks good.
-or-	rrc 00/--	if "nr" or initiation interrupt.
Field	positioning error	if read, data checks good, not "nc", and not initiation interrupt "pos" with position error.
-or-	166 data ers d/543210 p/------	if read, data bad, not "nc," and not initiation interrupt.

where

166	number of data words in error.
d/543210 p/------	bits dropped and picked up in a character. "-" if bit is ok, otherwise # is bit number.

-or-	data chks ok 012345012345	if read, data ok, not "nc" or initiation interrupt. Reports first data word.
-or-	data not checked (init)	if read, not "nc" and initiation interrupt.
-or-	data not checked (nc)	if read and "nc."
-or-	write data 123456123456	if write, and first DCW not non-data-transfer.
-or-	write data xxxxxxxxxxxx	if write, and first DCW is non-data-transfer.
-or-	write data character 00	if write, and IOC command "sing.char." is used.

If power off, line 02 is

power off, operator intervention required

If unexpected device or MPC attention, line 02 is

device attention, operator intervention required

Line 04:

Present only if a read data error is reported in line 02.

```
(002)012345012345 (003)012345012345 (004)012345012345 (005)012345012345
(pre)012345012345 (fol)012345012345 (002)012345012345 (003)012345012345
```

If present, the (002) or (pre) reports the location (offset) in the read area of the data word in error if numeric, or that the word preceding the read transfer (pre) or following the read transfer (fol) was bad. The 012345012345 is the data "was."

Line 05:

Present only if a read data error is reported in line 2 and a line 03 "was" is displayed.

```
s/b 010101010101  s/b 010101010101  s/b 010101010101  s/b 010101010101
```

If present, the data indicates the "should be" for the word immediately above.

Line 06:

Always present.

```
io#105
```

This line indicates the absolute number of connects (I/Os) issued to the device since the test started.

Line 07:

```
unrecoverable error after 3 retries
possible transient error, retry will be made
```

This line is displayed whenever there is a transient error recovery routine for the test page (mag tapes) and an error is detected for either a read or write. An IOC error, attention or power off, or initiation interrupt overrides the transient recovery routine and this line.

Line 08:

(a message supplied by a nsdd.dd instruction)

This line is present if a "nsdd.dd" instruction applies to this I/O. The data is the data pointed to by the "nsdd.dd".

Line 09 and 10:

```
device extended status in hex
extended status unreadable

device extended status in hex
00ec014d070400000000f0f708da91032261014200004ec00
```

Each pair of hex characters represents one byte of extended status. There are 24 status bytes, numbered 0-23. Refer to individual test page documentation for a breakdown of all 24 bytes.

```
00   represents a status byte of binary 00000000
ec   represents a status byte of binary 11101100
01   represents a status byte of binary 00000001
4d   represents a status byte of binary 01001101

(eep message line from tysdd instruction)
status was 123456654321
```

These lines are present if the Extended Error Processing (EEP) option "x" is on. If there was no error on fetching the extended status, the second of the three alternatives for line 08 and 09 are displayed. If extended status is available, the second alternative is displayed. If there was an EEP error, then the third alternative for the three is displayed. For this case the status given is the IOM status word #1 in octal format.

Line 11:

```
c,c,c,c, enter options:
```

This line is displayed if the halt (h) option is on and ns or "nsdd.dd" is not used. The c,c,c,c field is the current options in force. The terminal pauses for input options.

Extended Status Output Done As Freestanding Test

One of the following messages is displayed whenever the extended status test is run as a freestanding test via a txxx request.

NO ERROR DETECTED

The following message occurs if there were no errors detected while fetching the extended status via a device I/O:

```
**a(iccddc) device extended status in hex
eeeeee
```

where eeeeee is the formatted extended status line. Refer to the description of POLTS Standard Error Message, Line 08 and 09 above for additional data.

ERROR DETECTED

The following message is displayed if a status error was detected while fetching extended status:

```
**a(iccddc) extended status unreadable, status was ssssss
```

where ssssss is the first word of IOM status in octal.

POLTS Input Error Message

This message is displayed whenever an input following "test" contains an error:

```
***polts executive (erroneous data) invalid input
(reason)
```

where (reason) is one of the following:

channel not assignable
The channel number input is not configured on the system.

invalid channel number
The channel number input was nondecimal or greater than 63 for an IOM.

invalid device number
The device number input was nondecimal or greater than 63, or that particular device number was not configured to the system.

invalid IOM number
The IOM number input was nonoctal or greater than the largest IOM configured on the system.

no page exists for that device type
A test has been called for a device type for which there is no test page.

unknown request for this executive
The input does not match any of the valid inputs to POLTS.

POLTS Option Error Messages

The following message is displayed when options input to the test are processed and some error is detected in those options. A request to enter new options is always appended to this error message.

```
**o(iccddc) invalid option: (options)
(reason)
(current options) enter options:
```

where (reason) is any of the following:

already a manual test page
A .man option has been input but the test page is already a manual test page.

can't sequence tests outside of this segment
A test number input in response to a .seqt message was outside of the range specified in the message.

can't turn off test in another segment
Test tables are an integral part of each segment, sequencing or turning OFF can only be done for the current segment.

illegal control mnemonic (.option) encountered
A request was received for an option that was not recognized by the option processor.

jump specified to a test not sequenced
A jump ("-" in front of a test number) was specified in response to a .seqt message but the test number specified was not included in the sequence.

more than 6 alpha characters for tdl instruction
When inputting a TDL instruction line to a manual test page in response to .seqxx (data lines are not checked) message, an instruction with more than six alpha characters was encountered.

more than 12 numeric characters for tdl instruction
When inputting a TDL instruction line to a manual test page in response to a .seqxx (data lines are not checked) message, an instruction with more than 12 numeric characters was encountered.

next test sequencing has selected data line for test start
The first line of a test was changed to a data line in a manual test page and a subsequent txxx option (or some other test ending due to nx) tried to go to that test.

no executable tests in this sequence
The use of .seqt with a back jump and ntxxx options has set up a sequence with no tests that can be executed.

nonnumerics input for line number for control mnemonic
A .seqxx, .seqaxx, .stnxx, or .strxx option was input for a manual test page but the xx characters were nonnumeric.

only a "1" is allowed following "e-"
A character other than a "1" was used to try and restore transient error recovery retry count to the test page standard.

only numerics, commas, minus, or a blank allowed
Only test numbers separated by commas (with a possible initial "-" for jumping) are allowed as input to a .seqt message.

only 9 words allowed for TDL lines
More than 54 valid characters were input in response to .seqxx or .seqaxx messages.

pass or recycle must be set to output error tallies
A .tal option was input but neither p nor r was set ON.

specifying a nonexecutable test
One of the tests specified in response to a .seqt message has been marked as "do not run" for the reason specified in the test page initialization message.

test number cannot be '0'
This results if the test number input was zero or if no test number was input or the t was the last character of the input line (including the initial test request or test communications request limitation of 11 characters input).

test number is not in the current sequence
A test number has been specified that lies outside (was not included in) the sequence specified in response to a .seqt message.

test number must follow 't'
The t option character was not followed by a one-, two-, or three-numeric test number.

this control mnemonic valid only for manual test pages
.seqxx, .seqaxx, .stnxx, or .strxx was input to a comprehensive test page.

too many entries input
More than the specified number of test numbers were input in response to a .seqt message.

trying to execute tdl on a non-tdl line
The line specified in .stnxx or .strxx is not a TDL line.

trying to go to a nonexecutable test
The test specified has been marked as "do not run" for the reason specified in the test page initialization message.

trying to go to a nonexistent test
The test number txxx is higher than the last test for this page.

unknown tdl instruction specified
When inputting a TDL instruction line to a manual test page in response to a .seqxx (data lines are not checked) message, an unknown TDL instruction was encountered.

'v' is an illegal option character
'v' is some character other than b, e, h, i, l, n, o, p, r, s, t, u, x, or z. An illegal option character has been detected by the option processor.

'.' must be the first option character
A control mnemonic was found embedded in an option string.

Sample output:

```
**0(01601c) t0 enter options: c
**0(01601c) illegal option:
'c' is an illegal option character
t0 enter options:
```

POLTS Invalid TDL Instruction Message

The following message is displayed on the terminal if an unknown mnemonic or erroneous instruction usage was encountered during sequential processing of the TDL instruction.

```
**p(iccddc) invalid tdl instruction, line xx, field yy, aaannn
rrr
c,c,c,c, enter options:
```

where

1. xx

   is the line number.

2. yy
   is the field number.

3. aaa
   is the isolated alpha part of instruction.

4. nnn
   is the isolated numeric part of instruction.

5. rrr
   is one of the following reasons:

   >6 alpha or >12 numbers
   More than 12 numbers (0-9) or more than six nonnumerics other than a "," or "space" were encountered on a TDL instruction line with no intervening "," or "space."

   cannot specify an iom non-data command
   An ic01 (IOM channel command 02) was encountered.

   cannot use tdcw or idcw as first dcw
   Either cwobn or cwoi was encountered.

   clkran must have some maximum clock limit specified
   The "clkran" TDL instruction was input to a manual test page with no numerics or zeros.

   end of line sequencing would proceed on a non-tdl line
   The end of the current TDL instruction line was encountered and the last instruction processed required continuing to the next successive line and that latter line was found to be flagged as other than a TDL instruction line in the test page.

illegal line tally on chx
One of the TDL chx family of instructions specifies a tally of 64 or more characters, which is illegal.

inst. illegal in manual mode
The test page is attempting to execute a TDL instruction in manual mode. This is invalid due to system problems that could result if the instruction were used improperly.

insufficient fixed numerics
A TDL instruction was isolated that had less than the required number of numerics that are defined for the instruction.

inv. arg.
The instruction has been set up with invalid arguments.

invalid hexadecimal character in uhdln or phdln
Detected during data setup, possibly after the uhdlnxx or phdlnxx instruction. Some character other than 0-9 or a-f was detected in the lines covered by the range of the TDL instruction considering the current length.

last dcw cannot be an idcw
The test page is attempting to make the last DCW in a list an IDCW, which is illegal.

last word of dcw would be outside of write area
An ldd or control word instruction such as "cwdsddd.ddd" was encountered such that the resulting last word of the transfer area defined by the starting offset and length would be outside of the data transfer area (considered to be 320 maximum).

last word of message/data-from-line past line 99
One of the following instructions has specified data from TDL lines but the word count extends past the TDL line area: dlnxx, nsxx.yyy, phdlnxx, tyxx.zzz, uhdlnxx, yesxx.yyy.

last word of message would be outside of message area
An instruction to output a message line such as "tydd.dddd" was made with of a length of >72 characters.

line number specified exceeds 99
A TDL instruction requiring a line number was input to a manual test page with more than two numeric digits.

only 36 flags available, 0-35
An sflxx, ckfxx or rflxx instruction was encountered where xx was >35.

only octal numerics allowed
A TDL instruction was encountered that must have only octal numerics (0-7) and an 8 or 9 was found to be in the instruction.

positioning valid only for mag tape
A pos instruction was found in a test page for other than a mag tape.

record count must be 0-63
A rdd instruction was encountered with dd>63.

residual word count cannot exceed 320
A rwdd was encountered with dd>320.

retep illegal if not in eep
A retep instruction was encountered where POLTS was not in an "automatic" extended status fetch mode (not fetching extended status due to the x option after an error).

tally exhausted
One of the chx type instructions has attempted to store more than 63 characters.

tdl language lockup fault, no i/o for 275 major instructions
Greater than 275 TDL instructions were isolated without ever causing a test i/O to be issued.

the pass option must be set to output this message
The TDL instruction "tal" was input to a manual test page but the pass (p) option is not set.

trying to add offset or length to a tdcw or idcw
A "cwdaddd.dddd" instruction was encountered that pointed to a TDCW (cwdbd) or IDCW (cwdi).

trying to decrement the working module count to a negative value
The "dmod" TDL instruction was input to a manual test page and the working module count was never initialized via "smod" TDL instruction or the alternate return was ignored (the program goes to the next field if the working count is not a zero and skips one field if it is zero at the end of a "dmod").

trying to execute a ret without a previous sv
A retd was encountered without a svd having been used prior to the retd.

trying to execute tdl on a non-tdl line
A TDL loop instruction or "stnxx", "strxx", or "nxxx" was input to a manual test page but the line specified is not a TDL instruction line (either because it was input as a data line or because it was assembled as a data line). This could also result from a normal test page TDL instruction referring to a data line input to a manual test page.

trying to issue illegal op-code to device
A peripheral op-code defined as invalid for the device under test was found, resulting from a peripheral mnemonic such as rew or special I/O instruction such as swoo.

trying to jump to a nonexecutable test
The TDL instruction "nxxx" was executed in a test page but the test was turned OFF by the ntxxx option or test page initialization.

trying to jump to a nonexistent test
The TDL instruction "nxxx" was executed in a test page but no test starts on that line (could result from a .stnxx or .strxx option setting single test mode thus eliminating all other tests).

trying to jump to a test not in the current sequence
The TDL instruction "nxxx" was executed in a test page but the test starting on the specified line was not included in the sequence input in response to a .seqt message.

trying to set up the last dcw as an iontp or iotp
A "cwdpddd.dddd" or "cwdnddd.dddd" was encountered that would select DCW #9.

trying to start a dcw string with a tdcw (cwxby)
The TDL instruction "lcwx" was input to a manual test page but the control word specified was previously set up as a TDCW by the use of the TDL instruction "cwxby". This could also result from a normal test page "lcwx" command referring to a control word changed by the input of the TDL instruction "cwxby" to a manual test page.

trying to type 4096 words (0)
A TDL instruction to type message data such as "tydd.dddd" was encountered that specified a word count of 0.

trying to use a dcw with word count of 4096 (0)
A DCW setup such as via "cwdsddd.dddd" was attempted that had a word count of 0.

trying to use a sing. char. iom cmd with a read or nondata opcode
During I/O setup to issue a test I/O, it was found that an ic04 instruction had been used but the peripheral command is a read or nondata type op-code.

unknown iom command or bootload
An invalid IOM channel command or "boot" was specified in an ic00 instruction.

unknown tdl instruction specified
This message should never occur unless a bad patch was applied to the test page or the test page was loaded incorrectly. All manual pages have the TDL input checked at input time and an illegal options message would have resulted.

SECTION 6

ISOLTS EXECUTIVE

The ISolated OnLine Test Subsystem (ISOLTS) provides a means of testing Multics processors online, in an isolated environment under the TOLTS executive program.

FUNCTIONAL CAPABILITIES

The ISOLTS subsystem is two isolated and independent pieces of procedure code (driver and test programs) that communicate with each other for service.

The ISOLTS driver operates from an active Multics process and controls the testing and test sequencing of a processor that is under test.

Test programs, loaded into memory by the ISOLTS driver and executed by the processor under test, do the actual testing. The ISOLTS driver communicates with the test programs through a communication region, located at a predefined unpaged area in memory.

Communication from the ISOLTS driver to the test programs might include such requests as:

execute test program
halt operation immediately
execute program options
.
.
.

Communication requests from the test programs to the ISOLTS driver might include:

test program complete
load next test program
display error message
.
.
.

Because very few functions on a processor under test can be assumed to be operating correctly, precautions are taken to ensure that the processor under test does not degrade system integrity.

Isolation is provided because the memory area used for the test programs is dedicated and is not used for demand paging as most other system memory is. The base 64K words of memory on a nonbootload system controller is used for testing (the entire system controller and its memory is usurped from the system until

erroneous address above 64K is not generated). The operator is requested to reconfigure the processor under test such that the selected system controller and memory becomes zero based memory for that processor. The operator is also asked to disable all other processor ports and insert a 64K address plug in place of the current address plug for the selected system controller port.

The active Multics process in which the ISOLTS driver is executing, has access to the dedicated memory area by means of an unpaged Segment Descriptor Word (SDW) with the absolute system address of the dedicated memory area, and a bounds of 64K. The ISOLTS driver process references the dedicated memory as contiguous storage by using a pointer to the reserved hardcore segment. The ISOLTS driver process loads test programs into the dedicated memory area and causes the processor under test to execute these programs by simulating an interrupt with a Set Memory Controller Interrupt Cell (SMIC) instruction. The communication region is checked periodically for test completion or an error condition. This sequence is repeated for the next test and continues until no tests remain to be executed, an error occurs, or the ISOLTS user requests program options. If an error occurs, the error message is written to an error message file, which can be selectively displayed on the ISOLTS user terminal or dprinted.

When ISOLTS testing is complete, or if the ISOLTS process is wrapped up (either by error or by the user requesting termination), the reconfiguration process is reversed. The dedicated memory is returned to the system, the processor under test is made available for dynamic reconfiguration, and the SDW for the reserved hardcore segment is reset.

ISOLTS OPERATING INSTRUCTIONS

ISOLTS is entered by answering the TOLTS query:

```
***enter "polts", "molts", "isolts", "colts", "quit", or "msg"
???
```

with the keyword "isolts". The ISOLTS greeting message is followed by an input prompt and request for input:

```
***isolts executive version 781201 on 781202 at 17.235

***enter "test cpu <tag>", "display_error", "test pcd", "msg", or "quit"
???
```

If "quit" is typed, a signoff message is displayed as below, and control is returned to TOLTS:

```
***isolts executive version 781201 off 781202 at 19.594
```

If "test pcd" is typed, the current processor and memory configuration is displayed on the terminal, indicating the resource's relative availability for testing. An example of the test pcd request is included in the "ISOLTS Terminal Session" later in this section.

Actual processor testing is initiated by specifying the "test cpu" request. If the request:

```
test cpu <tag>
```

is typed, ISOLTS determines which of the available SCU/memory units to use for testing based on the one with the smallest amount of memory configured. If a problem is suspected in the processor port interface with a particular SCU/memory unit, the request:

```
test cpu <tag> -memory <tag>
```

-or-

test cpu <tag> -mem <tag>

can be used to specify the use of a particular SCU/memory unit.

Test Capabilities

Processor testing is accomplished in three phases. Each of these three test phases are described below.

TEST PHASE 1

This is the reconfiguration test phase. Upon recognition of a CPU test request (i.e., test cpu <tag>), the ISOLTS executive determines if the resources required for processor testing are available. (The processor to be tested must be configured in the off state and at least two SCUs must be online and available.) The system operator is asked for permission to test the requested processor using the designated SCU/memory unit. If permission is granted, manual reconfiguration instructions are displayed on the operator console (refer to "Communications with the System Operator" later in this section). After manual reconfiguration is completed, ISOLTS dynamically reconfigures the designated SCU, setting appropriate interrupt masks and port enable registers, to allow the processor under test to communicate with the lower 64K of address space of the designated SCU/memory unit. All memory in the designated SCU unit is removed from the system so that any reconfiguration test failure does not result in a catastrophic system failure. The actual reconfiguration test is conducted as follows:

1. The lower 256K of memory on the selected system controller is filled with a two-instruction sequence of STA *, DIS (* is equal to the current address).

2. The first two pages of all other nonbootload system controllers are "borrowed" from the system and the STA *, DIS sequence is stored in each pair of locations of these two pages.

3. The connect lock is set and all other active processors are forced to cease operation by sending them a connect.

4. Selected fault and interrupt vectors in the system zero-based memory are overlaid with the STA *, DIS sequence after saving their original contents in the process stack. The fault and interrupt vectors overlaid are:

   processor_start interrupt
   op_not_complete fault
   start_up fault
   lock_up fault
   trouble fault

5. An SMIC instruction is now sent through the selected system controller to the processor to be tested with a processor_start_pattern (interrupt cell 0).

6. A wait loop is entered, waiting for location 0 in the selected system controller's memory to change as a result of the STA 0 instruction previously stored there. If this location changes values, it means that the manual reconfiguration was done correctly.

7. If the wait loop times out, either the processor to be tested was not manually reconfigured correctly or the processor is not responding to

the SMIC instruction correctly. Tests are made by searching the memory of the other nonbootload system controller and looking at the fault and interrupt cells that were overlaid in the memory of the bootload system controller.

8. If location 0 of one of the other system controllers has changed, the operator has enabled the wrong port. In this case, the operator is informed to recheck the configuration panel on the processor to be tested.

9. In any case, the fault and interrupt vectors in the system zero-based memory are restored to their original contents and the connect lock is reset, allowing other processors to resume operation.

10. If the test of the other nonbootload system controller's base 2K of memory does not find any of the cells changed, it must be assumed that the processor did not answer the interrupt correctly. In any case, the base 2K of memory on each of these system controllers is given back to the system.

11. Each memory location of the selected system controller is tested to see if any location has changed. If some location has changed other than location 0, it could mean that the processor to be tested has an addressing problem.

12. In any case, an error code is returned to the caller, indicating the nature of the failure. All reconfiguration that has been done up to this point is undone before returning.

13. If the interrupt was answered correctly by the processor to be tested, ISOLTS ensures that other switches are set correctly. This is accomplished as follows:

    a. An RSW "n", DIS sequence is set up in locations 0 and 1 of the selected system controller's memory.

    b. An SMIC instruction is issued to the processor to be tested.

    c. After a short time delay an STA 0 is written over the RSW "n" instruction.

    d. An SMIC instruction is issued to the processor to be tested.

    e. Location 0 should now contain information that was stored in the A register as a result of executing the RSW "n" instruction.

    f. This information is saved and the RSW, STA sequence is repeated for each of four tags (1 through 4) of the RSW instruction.

14. The read switch data is exclusive ORed with the switch template and ANDed with the switch mask. The result of this boolean operation yields a value of zero if all the processor configuration switches are set correctly. If discrepancies are found, the resultant switch data is returned along with an appropriate error code.

15. If the read switch test finds no discrepancies, a test is made to determine if an "LDA 2" instruction operates correctly. This is accomplished as follows:

    a. Location 2 of the dedicated memory is set with a value of zero.

    b. An LDA 2, DIS instruction pair is set in locations 0 and 1 of the dedicated memory.

    c. An SMIC instruction is issued to the processor to be tested.

d. After a short time delay an STA 0 is written over the LDA 0 instruction.

e. An SMIC instruction is issued to the processor to be tested.

f. Location 0 (and the A register) should now contain a value of zero. (If not true, an appropriate error code is returned to the caller.)

16. Since all memory in the selected system controller above 64K is to be returned to the system paging pool, the ISOLTS subsystem verifies that the processor to be tested cannot address above 64K (because of the 64K size plug). This is accomplished as follows:

a. The "store fault" fault vector address in the dedicated memory is overlaid with an STA *, DIS pair.

b. Locations 0 and 1 of the dedicated memory are overlaid with an LDA 65536 (64K), DIS pair.

c. An SMIC instruction is issued to the processor to be tested.

d. After a short time delay, the store fault vector is checked, it should contain a value of zero. If it does not, an appropriate error code is returned.

17. Since the active test programs use a simulated IOM 0 terminate interrupt (interrupt cell 12) for testing, ISOLTS checks to see if the processor to be tested is capable of answering this interrupt. This is done as follows:

a. The scs$cpu_test_pattern is set with bit 13 (which is analogous to interrupt cell 12).

b. The IOM 0 terminate vector location in the dedicated memory is overlaid with an STA *, DIS pair.

c. An SMIC instruction is issued to the processor to be tested with the SMIC pattern in scs$cpu_test_pattern.

d. After a short time delay, the IOM 0 terminate interrupt vector location is checked, it should be equal to zero. If this is not true, an appropriate error code is returned.

18. The original port masks for the selected system controller and the port on which the tested processor is on are restored by an SSCR configuration instruction.

19. All memory above 64K in the selected system controller is given back to the system.

If an error is detected during the reconfiguration test, an appropriate error message is displayed and the dynamic reconfiguration of the SCU is reversed. The errors that may be detected at this time include:

1. Improper switch setting

2. Inability of the processor to respond to an interrupt

3. Answering an interrupt at the wrong location

4. Inability of the processor to generate a store fault when an address above 64K is referenced

TEST PHASE 2

This test phase executes the PAS2 Primitive Function Tests (PFTs). The PFTs are a group of go/no-go tests that run (in offline mode) as a result of booting the offline T&D tape. The PFT function is to verify the basic 20-instruction subset of the Series 6000 instruction repertoire that is used by the PAS2 executive. If the first PFT executes correctly, the second PFT is read in. This procedure repeats itself until the PAS2 executive is read in by the last PFT. If any of the PFTs fail, the processor halts at a DIS instruction. The FE can observe the instruction counter value of the DIS instruction from the processor maintenance panel and in conjunction with examining the listing for the failing PFT, determine the functional area of failure.

The ISOLTS executive times out five minutes after an error is detected by a PFT and an error message similar to:

isolts: time out after 300 seconds while executing PFT 04d

is displayed. At this time the user is given three choices of action with the query:

respond "quit", "retry", or "continue" -

If quit is entered, the processor under test is deconfigured and ISOLTS returns to ISOLTS command-level. If retry is entered, the test is reloaded and the timing loop is restarted at zero. If continue is entered, the timing loop is restarted at zero.

TEST PHASE 3

The PAS2 executive and the PAS2 test programs make up test phase 3. Together, these programs provide functional tests and limited diagnostics for all functions of a processor not tested by the PFTs.

The PAS2 executive provides control sequencing and utility service functions for the PAS2 test programs. In the offline mode, all requests to perform I/O to a tape unit, printer, or console are either communicated to the PAS2 executive from the test programs via Master Mode Entry (MME) faults, or initiated by the PAS2 executive directly. Program option control, operator interruption, and test termination are also handled by the PAS2 executive.

The PAS2 tests provide functional testing for all processor logic. If run in a fixed sequence, as they are designed to be, they can be thought of as building blocks. Each test checks out a particular functional area of the processor, and this functional area is used in the next test in the sequence to check out a functional area of greater complexity. This sequence continues through all of the tests, until the last test is encountered. This last test, the instruction sequence test, checks out processor instructions in random sequences to ensure that no instruction interaction or crosstalk occurred. If the PAS2 processor tests are run out of the designed sequence, a warning is issued indicating that the test called in uses unverified instructions. The warning is followed with a list of the unverified instructions (for inspection).

If a quit signal is issued by pressing the appropriate key on the terminal (e.g., QUIT, BRK, ATTN), ISOLTS interrupts the PAS2 Test Program/Executive and displays an "options" message. Any valid PAS2 option sequence can be typed in response. (Refer to the PAS2 Executive Program Listing in the Test and Diagnostic Microfiche Box for an explanation of the PAS2 options.)

ERROR MESSAGE PROCESSING

In the offline T&D system, when a PAS2 program detects an error, the resultant error message is sent to a printer. Under the ISOLTS subsystem, an error message is written to a file in the user's home directory with the name "isolts_err_log". This error file is created at first reference and is formatted with a 64K maximum size. The message entries are written into the file until it reaches the maximum size, at which time the oldest error message entries, at the beginning of the file, are overlaid with the new entry.

The user is notified that an error has occurred with the message:

*** an error has occurred ***

displayed on the terminal (unless the run option is in force). If the halt option is in force, PAS2 normally types out the option message and waits for entry of a valid PAS2 option.

To display the last error message (or any number of error messages) in the error file, enter one of the following requests:

display_error

-or-

display

The destination of the displayed error message is controlled by the type/print PAS2 option. If the default print option is in force, the requested error message is copied out of the isolts_err_log file into a uniquely named segment in the user's home directory. A daemon print request is then queued for the error message entry and an informative message is displayed on the terminal as follows:

error report file <unique_name>.ilog has been queued for printing

If the type option is in force, the error message is displayed on the terminal. The type option can be enabled when at isolts request level by using the -type control argument with the display_error command.

*** enter "test cpu <tag>", "display_error", "test pcd", "msg", or "quit"

??? display 1 -type

After the error message is output, the PAS2 options message is again displayed and any valid PAS2 option (or ISOLTS option) can be entered.

The display_error option also allows for extended display of the error message file. The request:

display_error 8

displays the last eight message entries in the isolts_err_log. The request:

display_error 8 2

displays two message entries that start with the eighth oldest message. The request:

display_error -all

displays all message entries in the isolts_err_log.

COMMUNICATIONS WITH THE SYSTEM OPERATOR

Since the ISOLTS user can be logged into the system from a location distant from the computer room, a method is needed to inform the system operator of the test request. The operator must respond with approval or denial of the request, and if approval is granted, the operator must perform manual reconfiguration of the processor to be tested. The sequence of messages and operator responses displayed on the operator console consists of one of the following scripts.

For an L68 cpu (where input typed by the operator is preceeded by "!"):

```
     permission asked to test cpu <tag> using memory <tag>
     respond: "x oqr grant"
          or  "x oqr deny"
  !  x oqr grant          (operator response)
     execute the following manual reconfiguration on cpu <tag>:
     1.   set all port and initialize enable switches and interlace switches to
          off.
     2.   set the assignment switches for all ports to 000.
     3.   remove the right free-edge connector on the 645pg wwb at slot ab28.
     4.   install the "cpu test" on the right free-edge connector at slot ab28.
     5.   press the initialize and clear push button.
     6.   set the port enable switch "on" for port <tag>.
     respond: "x oqr done" when reconfiguration is complete.
  !  x oqr done           (operator response)
```

For a dps8m cpu:

```
     respond: "x oqr grant"
          or: "x oqr deny"
  !  x oqr grant
     execute the following manual reconfiguration on cpu <tag>
     permission asked to test cpu <tag> using memory <tag>
     1.   set all port and initialize enable switches and interlace switches to
          off.
     2.   set all port assignment switches to 000 and the size switches to full.
     3.   set store size switches to 2222.
     4.   verify that the mode switch is in vms.
     5.   depress the initialize and clear push button.
     6.   set the port enable switch to "on" for port <tag>.
     respond:  "x oqr done" when reconfiguration is complete.
  !  x oqr done
```

The operator responses are made via the opr_query_response command, or the admin exec_com entry (x oqr <response>). The ISOLTS user's process is blocked while waiting for the appropriate operator response.

SPECIAL TOOL REQUIREMENTS

In order to ensure isolation of the processor under test from the Multics storage system, a special free-edge connector (provided by Field Engineering) must be used to replace the current physically configured port size free-edge connector installed on the 645PQ WWB at location slot 28L in the processor cabinet. This special free-edge connector (part no. 58052006-001) has all port groups wired for 64K. If this tool is not available, one can be made by taking a blank free-edge connector and inserting four 64K size plugs, in the four available slots. This free edge must be installed when the manual reconfiguration is performed or when the reconfiguration test fails.

ISOLTS TERMINAL SESSION

Following is a typical example of an ISOLTS terminal session. Numbers that appear in the left margin are comment identification numbers that correspond to the numbered explanations described in the "Notes" following the example.

```
***isolts executive version 781201 on 781202 at 17.235

***enter "test cpu <tag>", "display_error", "test pcd", "msg", or "quit"
??? test pcd

1   isolts configuration:

cpu a a 168/80 cpu on scu port 3 is online & unavailable for test
cpu b a 168/80 cpu on scu port 4 is offline & available for test
cpu c a 2K cache 168/80 cpu on scu port 5 is online & unavailable for test
cpu d a 32K cache dps8/70 cpu on scu port 6 is online & unavailable for test
cpu e a 32K cache dps8/70 cpu on scu port 7 is online & unavailable for test
scu d has 2048k words of memory & is the bootload scu & is unavailable
scu a has 1024k words of memory & is offline & unavailable for test
scu b has 2048k words of memory & is online & available for test
scu c has 4096k words of memory & is online & available for test


***enter "test cpu <tag>", " display_error", "test pcd", "msg" or "quit "
??? test pcd cpud

cpu configuration:

cpu d a 32K cache dps8/70 cpu on scu port 6 is online & unavailable for test


***enter "test cpu <tag>", " display_error", "test pcd", "msg" or "quit "
??? test pcd memc

mem configuration:

scu c has 4096k words of memory & is the bootload scu & is unavailable


***enter "test cpu <tag>", " display_error", "test pcd", "msg", or "quit "
??? msg

enter 1 line message of up to 80 characters
??? May I use cpu b?
ready       (operator response)


***enter "test cpu <tag>", "display_error", "test pcd", "msg" or "quit"
2   ??? test cpu c -memory c
asking operators permission to test cpu "c" using memory "c"
permission granted
asking operator to manually reconfigure cpu c

reconfiguration complete

start pft 01c

3   *** end 01c, next 01z ***
    *** end 01z, next 02a ***
    *** end 02a, next 02b ***
    *** end 02b, next 03b ***
    *** end 03b, next 04b ***
    *** end 04b, next 04c ***
    *** end 04c, next 04d ***
    *** end 04d, next 04e ***
    *** end 04e, next 04f ***
```

```
      *** end 04f, next 05c ***
      *** end 05c, next 061 ***
      multics pas executive rev. m     112078

4     rsw test
      check visually
      data switches = 000030710000
      configuration switches =
      ports(a-b-c-d)  001001041001
      ports(e-f-g-h)  001001001001
          port bits: a/e=0-8 b/f=9-17 c/g=18-26 d/h=27-35
         address assignment = bits 0-2, 9-11, 18-20, 27-29
         port enable = bits 3,12,21,30
         initialize control = bits 4,13,22,31
         interlace enable = bits 5,14,23,32
         memory size = bits 6-8, 15-17, 24-26, 33-35
      miscellaneous switches(rsw,2) = 000100000562
         fault base=6-12 bar mode=17 processor id.=26-33
         processor no.=34-35
      miscellaneous switches(rsw,4) = 000000020000
         ports(a thru h) interlace size bits:
           a=13 b=15 c=17 d=19 e=21 f=23 g=25 h=27
         ports(a thru h) half/full size bits:
           a=14 b=16 c=18 d=20 e=22 f=24 g=26 h=28
      the above processor identification code from the
      miscellaneous switches(rsw 000002) indicates a
      ******************************************
      *           model 6180   cpu in operation*
      *     cpu installed options include...   *
      *                 cache memory           *

      *options?test msg

      enter 1 line message of up to 80 characters
      ??? Pls set data switches to zero.
      *options?test msg

      enter 1 line message of up to 80 characters
      ??? are you done?
      yes        (operator response)

5     *options?
4     700 rev c    hstry reg/mode reg
      pm701 rev c    hstry reg/mode reg
      ps702 rev 00a  fixed add-a reg.
      ps705 rev 00a  fixed add-q reg.
      ps710 rev 00a  fixed add aq reg.
      ps715 rev 00a  fixed add * x-regs
      ps716 rev 00a  fixed add * x-reg.
      ps717 rev 00a  fixed add * x-reg.
      ps720 rev 00a  fixed add to store
      ps721 rev 00a  fixed add to store
      ps722 rev 00c  fixed add to store
      ps725 rev 00a  shift group
      ps730 rev 00a  fixed mult/div
      ps734 rev 00a  floating add 1
      ps735 rev 00a  floating add 2
      ps736 rev 00a  floating add
      ps737 rev 00a  floating add
      ps739 rev g    655 floating round
      ps740 rev 00a  conditional xfer
      ps745 rev 00a  floating mult/div
      ps750 rev 00a  control register
      ps755 rev 00a  misc. instructions
      ps760 rev 00a  index mod. r,ir,ri
      ps762 rev 00a  it mod. & rep ins.
      ps763 rev 00a  it mod. & rep ins.
      ps764 rev 00a  it mod.-scr
      ps766 rev 00a  multilevel mod....
```

```
ps768 rev 00a  it mod & rep inst.
ps769 rev b    it mod & rep inst.
ps770 rev 00a  misc. logic
pm776 rev h    6000 fault logic
pm785 rev a       master mode
pm785 uses unverified instructions:
smic
ps791 rev 00a  ill.procedure flt.
pm792 rev b    ill.procedure flt.
ps800 rev b    cntrl & xfer inst.
ps805 rev 00c  address reg tests
ps808 rev 00d  eis alphanumerics
ps810 rev 00b  eis numerics
ps815 rev 00b  eis multiply test
ps817 rev 00b  eis divide test
ps825 rev 00b  eis bit strings
ps830 rev 00c  edited move group
ps830 uses unverified instructions:
lpri
ps835 rev 00a  eis misc logic tst
ps835 uses unverified instructions:
lpri
ps836 rev 00a  eis misc logic tst
ps836 uses unverified instructions:
lpri
ps838 rev b    eis mod field test
ps840 rev 00a  eis ill. procedure
ps841 rev 00c  eis ill. procedure
pm842 rev h    eis interrupt
ps843 rev b    eis rewrt & align
ps844 rev 00a   character board
ps845 rev a     eis reg. test
ps846 rev b     eis adder test
ps847 rev a     eis btd test
ps848 rev a     eis dtb test
ps849 rev a     eis reg. test
pa851 rev d    reg inst test no 1
pa851 uses unverified instructions:
lpri
pa852 rev b    reg inst test no 2
pa853 rev b    multics instr mods
pa855 rev d    am instruct test
pa860 rev b    appending 1
pa861 rev c    appending 2
pa862 rev d    appending 3
pa863 rev d    appending 4
pa864 rev f    appending 5
pa865 rev f        bar mode
pa870 rev d      scu/rcu faults
pa875 rev c    eis append test
pa878 rev d    eis faults test
pa880 rev c    call & rtcd test
pa885 rev f      misc. logic
pa886 rev b       misc logic 2
pa887 rev c       misc. logic
pa890 rev d    multics ill. proc.
pm898 rev h    cache memory
*** presently executing prg898 with cpu voltage= low  ***
*** presently executing prg898 with cpu voltage= high  ***
ps905 rev b    random eis test

6   ************************************
ps940 rev d    slow/fast store sq
if type or atype option is in use set cp switch 25 up
ps945 rev d    mpy & div seq test
if type or atype option is in use set cp switch 25 up
ps955 rev g    pas sequence test
***start revision g  instruction sequence test
```

```
     start sequence test
7

8    ****************************************

9    *options? margin
     isolts: margin option not supported by isolts
10   *options? pgm781
     isolts: 781 not supported by isolts
11   *options? pgm870
12   *options? alter 100023
     100023 - 000000000000
            - 000100000000
     100024 - 000000000000
            - opt
13   *options?
     pa870 rev d      scu/rcu faults
     end of program
14   *options? quit
     ***enter "test cpu <tag>",  "display_error", "test pcd", "msg", or "quit"
15   ???  quit
     ***isolts executive version 781201 off 781202 at 19.594
```

Notes

1. This output is produced by the test pcd request.

2. This is a cpu test request using the "-memory" control argument. Notice that the test pcd output (results of the first request) indicates that memory "d" has only 256K of memory configured while memory "c" has 512K. If the "-memory" control argument is not used, memory "d" is selected for testing since it has the smallest amount of memory.

3. This is the first PFT. The associated message indicates which PFT is executing.

4. This is output produced by the PAS2 executive, which normally goes to the printer in the offline environment. When running ISOLTS, all output that normally goes to the printer, except error messages, is displayed on the terminal.

5. This is an enter options request from PAS2. Any valid PAS2 options can be entered at this time. In this case, pressing the "return" key, simulates pressing the EOM button on the system console if the PAS2 tests were being run offline. At this point, depressing the return key on the terminal means the same as if the "seq" option was entered.

6. Program 905 executes for an extended period of time. Each star represents the fact that 10000 test cases have been completed. The stars are output so that the ISOLTS driver does not time out since there is no other interaction with the driver if no errors occur. This is different from the offline version where no star line is output.

7. This white space is output from program 955.

8. Program 955 runs indefinitely or until the user stops it. The line of stars again represents 10000 test cases per star have been executed.

9. An attempt to enter a PAS2 option not supported by ISOLTS results in an error message output. ISOLTS returns to the enter options state.

10. An attempt to call in a PAS2 program not supported by ISOLTS results in an error message output. ISOLTS returns to the enter options state.

11. A valid PAS2 program is requested. After the program is loaded, ISOLTS once again returns to the enter options state.

12. Request to patch a location within the slave program using the alter option. PAS2 responds with the current contents of the location to be patched. PAS2 then spaces over and outputs a "-" and then allows the user to input the patch. After the patch is entered, PAS2 responds with the contents of the next location. The "opt" request forces an exit of the alter loop. ISOLTS returns to the enter options state.

13. PAS2 puts the previously loaded program into execution when the return key is pressed. The program start message is output, the program runs to completion, and the "end of program" message is output. ISOLTS again returns to an enter options state.

14. The"quit" or "interrupt" request simulates pressing the operator request button in the offline environment. All testing stops and the PAS2 exec goes to an enter options request. In this case you want to quit or terminate ISOLTS testing, which is indicated by typing "quit" in response to the option request. ISOLTS deconfigures the processor under test and returns to ISOLTS command level.

15. Request to quit from ISOLTS. Control is returned to TOLTS command level.

SECTION 7

COLTS EXECUTIVE

355/6600 FNP adapters and remote devices in an online environment. This capability allows equipment to be tested without denying other users the use of the central computer system and aids in more effective scheduling and use of preventive maintenance time. Any FNP subchannel may be tested if it is not dialed, provided the user has proper access to the ".acs" segment for the channel.

FUNCTIONAL CAPABILITIES

The COLTS executive is part of TOLTS. COLTS issues all messages via the TOLTS executive messages. It performs the following services for the test pages:

- resource allocation
- memory allocation
- test page loading
- test page dispatching
- test sequencing
- option processing
- I/O set up and issue
- error checking and error message formatting
- test page termination
- memory deallocation
- resource deallocation

Hardware Tested

The following list includes the hardware tested by the COLTS Executive.

DEVICE DESCRIPTION	TEST PROGRAM NAME
High-Speed Line Adapter, HSLA	C03A, C03B, C03C

Documentation for the test programs is located in the T&D Microfiche Documentation Box.

COLTS OPERATING INSTRUCTIONS

COLTS is entered by answering the TOLTS query:

```
***enter "polts", "molts", "isolts", "colts", "quit", or msg"
???
```

with the keyword "colts". After a slight pause for the core image loading of the COLTS slave executive, an input prompt/request for input (???) is displayed. When the prompt is responded to with a valid test request, (for any channel not in a dialed-up state) COLTS responds with a greeting message.

SPECIAL NOTICE:

Due to hardware restrictions the following procedure is required when attempting to test an odd channel number.

The test must be started on the next lower even channel first and then the odd channel started. It is recommended that the even channel be started first, and either run with an "r" option or put into wait with a ".wait" option. It is important that this test remain active while the odd channel is running.

```
???  test x...
***colts executive versions wwwwww xxxxxx on yyyyyy at zz.zzz
```

where:

1. test x...
   can be any of the requests listed below. The maximum character length of any request is 11 characters (including options).

2. wwwwww
   is the Multics interface module version date expressed as yymmdd.

3. xxxxxx
   is the slave COLTS version date expressed as yymmdd.

4. yyyyyy
   is the current date expressed as yymmdd.

5. zz.zzz
   is the time in hours and thousandths of hours.

Any time a quit signal is issued, by pressing the apprropriate key on the terminal (e.g., QUIT, BRK, ATTN), COLTS responds by displaying "???" and pauses for entry of a new msg (refer to "Tolts Operating Instructions" in Section 2) or test request.

Test pcd Request

The user request:

```
???  test pcd
```

displays the current fnp configuration in a manner similar to the following example.

```
colts configuration:

fnp a (0) on iom b 18 is a dn355  with 32 k of memory and 2 hslas is up
fnp b (1) on iom a 18 is a dn6670 with 128 k of memory and 3 hslas is up
fnp c (2) on iom a 13 is a dn6670 with 128 k of memory and 3 hslas is up
```

```
fnp d (3) on iom b 14 is a dn6670 with 128 k of memory and 3 hslas is down
fnp e (4) on iom b 19 is a dn6670 with 128 k of memory and 3 hslas is up
fnp f (5) on iom b 15 is a dn6670 with 128 k of memory and 3 hslas is up

???  test pcd fnpc

fnp configuration:

fnp c (2) on iom a 13 is a dn6670 with 128 k of memory and 3 hslas is up
```

where the number in parenthesis represents the logical FNP number.

Test List Request

To list the active test pages in COLTS, enter one of the following requests:

```
test clstal
```

-or-

```
test lstal
```

Sample output:

```
???colts lstal
colts lstal:
in execution:
**0(00601)01 hsla 7/bi
```

If a waiting allocation message is present, it indicates that the device cannot be assigned to COLTS at this time because it is presently in use by some other process.

Test Wrapup Request

To wrapup (stop) all COLTS testing, enter one of the following requests:

```
test cw
```

-or-

```
test w
```

Sample output:

```
??? test w
***colts executive wrap-up code td2 p.t. 24148
Do you want to return the channel a.h001 to service?
Please answer yes or no
```

New Test Requests

Initiation of new test requests depend on the type of hardware to be tested. Following is the test initiation sequence for all COLTS supported hardware.

```
test cnccssooooo
```

where:

1. c
   is the COLTS executive designation.

2. n
   is the logical communications processor number.

3. cc
   is the communications processor IOM channel number in decimal.

4. ss
   is the subchannel number in decimal.

5. ooooo
   is the new option character set.

Sample output:

```
??? test c00601o
***colts executive versions 810301 800110 on 810812 at 17.69
```

Test Communication Request

To communicate with a test entry (an active COLTS test page), follow the executive designator (refer to "New Test Request" above for description of characters) with the character "o" followed by the test page header and the communications desired.

```
test conccssoooo
```

where the trailing o... is the new option character set (the "o" option interrupts a test page).

Sample output:

```
??? test co00601.i
**0(00601)01 hsla t0 enter options: .go
**0(00601)01 hsla,i end t1,next t2
**0(00601)01 hsla,i end t2,next t3
**0(00601)01 hsla,i end t3,next t4
**0(00601)01 hsla,i end t4,next t5
**0(00601)01 hsla,i end t5,next t6
**0(00601)01 hsla,i end t6,next t7
.
.
.
```

Test End Request

To terminate an active test page or a test entry, follow the executive designator with the character "e" followed by the test page header.

```
test cenccss
```

Sample output:

```
??? test ce00601
*** colts executive forced term 2:  3 status and 0 data errors
```

Standard Test Page Options

The following error and control options are the same for all TOLTS subsystems and can be entered in response to an enter options message or designated in the options string of a new test request.

OPTION CHARACTERS AND DESCRIPTION

a accumulate the error messages in the statistical collection file; test page start and termination messages unconditionally go there.

b bypass error message output. Bypass overrides a pass or cycle message unless halt is set. Halt forces these messages to be displayed (overriding the bypass option).

h halt for input of options following error messages, end test messages, end pass messages, and end cycle messages.

i inform the user of each normal "end test." If halt (h) is also specified, then an "enter options" is appended to the end test message. The end test message is overridden if the next segment is being called, an end pass message is being output, or if a cycle ends.

l loop on current test (cannot loop on test 0).

n negate the following option character (valid only if preceding a, b, e, h, i, l, p, r, or the t options).

o go to "enter options" following processing of the complete option string containing the "o".

p issue an end pass message any time a back jump is detected by next test sequencing. If halt (h) is also specified, then an "enter options" is appended to the end pass message. The error tallies for the current pass are reset when "end pass" or "end cycle" is reported or when "p" is being turned on.

r issue an end cycle message any time a normal test page termination occurs and cycle back to the first test in the current sequence. If halt (h) is also specified, then an "enter options" is appended to the end cycle message. The error tallies for both the current pass and current cycle are reset when end cycle is reported. The error tallies for the cycle are reset whenever "r" is being turned on.

s unconditionally skip to the next test.

txxx if turned on (no preceding negate), then unconditionally jump to the first occurrence of the test in the current sequence. The test number xxx must follow and must be nonzero, and can consist of one, two, or three characters. For segmented test pages, a value outside of the current segment causes a jump to the corresponding segment without further processing of the current option string. If turned off (ntxxx), then the test number must be in the current segment and sequence, and not the forced termination test number.

v specify device test instead of subchannel test. Can be used only on the initial test call and must be the first character following the subchannel number. Not used for page communications request.

Control Mnemonics (prefixed with ".")

NOTE: Only one control mnemonic (.option) can be input for any single "enter options message" and it is processed only if found at the beginning of the option string.

.go return to the test page where interrupted. If the skip (s), jump (txxx), ".rseq", or ".seqt" option is specified, the next test sequencing is done.

.opt an enter options message is output.

.pr2 .prt forthcoming test page error messages are appended to a uniquely named stream file, which is printed on test termination, test wrapup, or acceptance of the .typ control mnemonic.

.rseq resequence tests to their initial (at page call) order.

.seqt sequence tests in special order; a message is output:

**x(header) input up to zz test #'s from xxx to yyy in the sequence desired

where:

1. **x(header)
   is the test page header.

2. zz
   is the number of tests allowed.

3. xxx
   is the lowest test number allowed.

4. yyy
   is the highest test number allowed.

Test numbers are input in the sequence desired and are separated by commas. A "-" in front of a test number indicates a jump either forward or backward to the first occurrence of that test number in the new sequence. For example, test numbers 1, 2, 3, 4, -1, indicates running tests in the sequence 1, 2, 3, 4, 1, etc., with a back jump indicated at test 4 (affects the pass (p) option).

.tal a message with a tally of errors is output. Option p or r must be set. The message includes the information for the current pass or cycle, or both (depending on the state of p or r). The error tallies are reset for the pass or cycle, or both when reported.

.test e the test page is forcibly terminated.

.test w COLTS is wrapped up.

.typ forthcoming test page errors are output on the terminal.

.wait the test page is put in a wait condition.

SPECIAL TEST PAGE CONTROL MNEMONICS

If a control mnemonic cannot be recognized by the COLTS executive the mnemonic is passed to the indicated test page for processing. If not recognized by the test page, an invalid input message results. For detailed information check the test page documentation.

The following control mnemonics are valid for HSLA test pages.

.eloop loop on subtest in error.

.neloop turn off .eloop option.

.sloopxx restart current test and loop on subtest xx.

.nsloop turn off .sloopxx option.

The following control mnemonics are valid for the HSLA test page only.

.async configure and test a general-purpose subchannel as an asynchronous subchannel.

.synch configure and test a general-purpose subchannel as a TLPK synchronous subchannel.

.gp resets the .async and .synch options.

Any other option except carriage return or newline (CR and NL respectively) on a console is invalid.

SPECIAL TEST PAGE CONTROL OPTIONS

The following options are valid only in a test end or test communication request (not in response to enter options) and only when they are the first character(s). These options allow ending and communicating with multiple-called (either deliberately or accidently) test pages for the same device. The test page number (or queue designator) is found as the third character (x) of the standard header which is output with every message for that page and also in the lstal messages **x(nccss)id.

0-7 applies only to active test page 0-7 and only if the rest of the header matches. The character is deleted before passing the rest of the string to the option processor.

qa-qp applies only to queued test a-p and only if the rest of the header matches. The characters are deleted before passing the rest of the string to the option processor.

COLTS Informative Messages

COLTS LOG ON MESSAGE

A message is displayed on the terminal when COLTS is loaded in core. (Refer to the example included under "COLTS Operating Instructions" earlier in this section.)

COLTS LOG OUT MESSAGE

The following message is issued every time COLTS terminates with nothing more to do (not due to wrapup or abort):

```
***colts executive version wwwwww off xxxxxx at yy.yy p.t. zzzzzzzzzz
```

where:

1. wwwwww
   is the TTL card date in mmddyy.

2. xxxxxx
   is the current date in mmddyy.

3. yy.yy
   is the time in hours and hundredths.

4. z...
   is the processor time used in milliseconds.

COLTS WRAPUP MESSAGE

The following message is issued every time COLTS wraps up due to test w, test cw, .test w option, or TOLTS executive abort:

```
***colts executive wrapup code yyy p.t. zzzzzzzzzzzz
```

where:

1. yyy
   is one of the following codes:

   td1 - TOLTS is not in core (wrapped up).
   td2 - test w or test cw from local console.
   td3 - .test w option from the user.
   td5 - test w or test cw from the user.

2. z...
   is the processor time in milliseconds.

COLTS ABORT MESSAGE

The following message is issued if COLTS aborts:

***colts executive wrapup at xxxxxx code yyy p.t. zzzzzzzzzzz

where:

1. xxxxxx
   is the instruction count (IC) where the fault occurred (0 if abort).

2. yyy
   is the three-digit GCOS abort reason code.

3. z...
   is the processor time in milliseconds.

COLTS LSTAL MESSAGE (BUSY)

One of the following messages is issued in response to a test lstal or test clstal request.

```
colts lstal:
in execution:
**x(header)ww zzz pp/ll


in hold (waiting):
**x(header)ww zzz pp/ll


waiting for memory:
**y(header)ww zzz pp/ll


waiting on active lsla page:
**y(header)ww zzz pp/ll


waiting for allocation:
**y(header)00 zzz pp/ll


colts executive entries:
```

where:

1. x
   is the active test page number 0-7.

2. y
   is the queue number A-P.

3. ww
   is the MCS DAC identifier assigned to this entry.

4. zzz
   is the hardware identifier mnemonic.

5. pp/ll
   is the physical/logical identifier.

The above messages have the following meaning:

in execution
the test page is actively running.

in hold
the test page is waiting for a test communications request.

waiting for memory
eight pages are active already, the user was denied memory by MCS, or waiting for Multics to transmit the HSLA test channel module to the DATANET 355/6600 processor. If the user is waiting on communications processor memory, an informative message for that test entry would have been previously outputted. If a queued entry is not waiting for allocation then it is assumed to be waiting for memory.

waiting for allocation
the resource requested is busy (allocated to another user).

COLTS executive entry
the remote device controlling entry is active for DATANET 355/6600 processor number n -- remote device test requests are honored.

COLTS LSTAL Message (Idle)

The following message is issued if the executive is ready to terminate normally when it receives a test lstal or test clstal request.

```
colts lstal:
executive idle
```

COLTS Test Page Messages

In explaining the test page messages, the sequence of "***(header)" is used to represent "**x(nccss)xx zzz" where xx and zzz are explained under "Test Page Start Message" below.

TEST PAGE START MESSAGE

The test page start message is issued whenever a test page is loaded.

```
**x(header)xx zzz start cccccc - nnnnnn,
     ttldat dddddd, phy./log.id pp/ll
```

where:

1. xx
   is the Multics communication system DAC identifier used for test I/O.

2. zzz
   is the hardware identifier mnemonic.

3. cccccc
   is the GCOS catalog name.

4. nnnnnn
   is the descriptive name.

5. dddddd
   is the ttl card date in mmddyy.

pp/ll
   is the physical/logical identifier.

TEST PAGE ENTER OPTIONS MESSAGE

The test page enter options message is issued whenever the "o" option is encountered in an option string after processing all of the options in the string or when the .opt option is encountered. Part of the message is appended to other messages whenever input options are required or desired. In this case the current options and enter options are appended.

**x(header) txxx,o,o,o, enter options:

where:

1. xxx
   is the last test in execution.

2. o,o,o,
   is the current options in effect.

TEST PAGE WAITING MESSAGE

The test page waiting message is issued every minute that a test page is waiting for a test communications request.

**x(header) waiting

TEST PAGE END TEST MESSAGE

The test page end test message is issued every time a test ends normally and the "i" option is on.

**x(header) o,o,o,o,o,o,o,o,o,o, end txxx, next tyyy

where:

1. o---o
   is the current options in force.

2. xxx
   is the test number of the test just finished.

3. yyy
   is the test number of the next test to be run.

TEST PAGE END PASS MESSAGE

The test page end pass message is issued when the next test sequencing detects a back jump in the test sequence and the "p" option is on.

**x(header) end pass ppp: ssssss status and dddddd data errors

where:

1. ppp
   is the pass number in this cycle.

2. ssssss
   is the number of status errors.

3. dddddd
   is the number of data errors.

TEST PAGE END CYCLE MESSAGE

The test page End Cycle message is issued when the end test terminates (if the page is not being force terminated or aborted) and the "r" option is on.

**x(header) end cycle ccc: ssssss status and dddddd data errors

where:

1. ccc
   is the cycle number.

2. ssssss
   is the number of status errors.

3. dddddd
   is the number of data errors.

TEST PAGE ERROR TALLY OUTPUT MESSAGE

The test page error tally output message is issued in response to the .tal option if the "p" option or "r" option is ON. The message depends on "p" or "r" or both for the format (for pass, for cycle, or for pass and cycle).

**x(header) for pass ppp: ssssss status and dddddd data errors
o,o,o,o, enter options

**x(header) for cycle ccc: ssssss status and dddddd data errors
o,o,o,o, enter options:

**x(header) for pass ppp: ssssss status and dddddd data errors
and cycle ccc: ssssss
status and dddddd data errors
o,o,o,o, enter options:

where:

1. ppp
   is the pass number in this cycle.

2. ccc
   is the cycle number.

3. ssssss
   is the number of status errors.

4. dddddd
   is the number of data errors.

5. o,o,o,o,
   is the current option set.

TEST PAGE NORMAL TERMINATION MESSAGE

The test page normal termination message is issued when the end test terminates normally (if the page is not being force terminated or aborted) and the "r" option is not on.

**x(header) normal term ccc: ssssss status and dddddd data errors

where:

1. ccc
   is the cycle number.

2. ssssss
   is the number of status errors.

3. dddddd
   is the number of data errors.

TEST PAGE FORCED TERMINATION MESSAGE

The test page forced termination message is issued when a test page is force terminated or aborts.

**x(header) forced term ccc: ssssss status and dddddd data errors
(reason)

where:

1. ccc
   is the cycle number.

2. ssssss
   is the number of status errors.

3. dddddd
   is the number of data errors.

4. (reason)
   is one of the following reason messages:

   controlling terminal disconnected
   the remote console controlling this test page was disconnected for some reason.

   icm data checksum error
   a data checksum error was detected in the transmit or receive portion of the test icm exchanged with the communications processor. The test page is terminated because COLTS cannot rely on proper execution of the test icm.

   no executable tests in this sequence
   an invalid option message with the same reason was ignored by not using the txxx, .seqt, or .rseq option to turn some test on.

test e request received
the controlling terminal input a test end request for the test page.

test line disconnect
Multics communication system disconnected the dac test line established for this test page. The disconnect was probably caused by some internal data error from which the Multics communication system cannot recover.

.test e request received
the controlling terminal for this test page input the .test e option in response to an enter options message.

Informative messages not common to COLTS can be output by the individual test pages under COLTS. For an explanation of these, if any, refer to the appropriate test page documentation.

TEST PAGE DEBUG ERROR MESSAGE

This message should only occur during test page debug.

The following messages may be appended to the message listed under forced termination above.

dcw specified is out of bounds for mme tadio

e action code cannot also have c s or r for mme tadio

ep action code cannot also have m for mme tadio

ep action code requested but printer in use flag not set for mme tadio

ep action code valid for last designator only for mme tadio

first word of dcw is out of bounds for mme tadio

ic value out of bounds for requested bar for mme gelbar

illegal mme detected

illegal op-code specified in mme 355exc call

illegal test page .option processing routine address

information pointer out of bounds for mme gelbar

invalid .option detected by test page but invalid reason iotd supplied

iontp dcws are illegal for mme tadio

last word of dcw is out of bounds for mme tadio

mme errset address limits exceeded

mme errset parameter error

mme gelbar cannot relocate or restrict in 512-word blocks

mme gelbar requested has last address outside of the program

mme gelbar timer value cannot be 0

mme gelbar timer value cannot exceed 35 binary bits

mme geloop timer value cannot be 0

mme geloop timer value converted to clocks cannot exceed 35 binary bits

mme gesnap illegal without trace set

mme gewake clocks tod cannot exceed 35 bits

mme illegal in master mode

multiple copies being sent to same terminal type for mme tadio

no action code for designator for mme tadio

no information set up for controlling terminal for mme tadio

no input buffer specified for mme 355oi

only 1-5 designators allowed for mme tadio

only 25 dcws are allowed in a string for mme tadio

only 1019 words allowed for mme tadio

p action code cannot also have c for mme tadio

p action code used but no printer available for mme tadio

printer in use flag set but no printer message issued for mme tadio

r action code must be the only one in a designator for mme tadio

r action code valid only for last designator for mme tadio

register pointer out of bounds for mme gelbar

starting address out of bounds for mme gesnap

test page printer available flag set but none available for mme tadio

too many words for mme gesnap

trying to get to consecutive i/o queues without link

trying to jump to a nonexecutable test

trying to jump to a nonexistent test

trying to jump to a test not in current sequence

use of e action code has caused more than 25 dcws for mme tadio

use of e action code has caused word count to exceed 1019 for mme tadio

user requested abort but invalid reason iotd

user requested abort but reason iotd pointer out of bounds

user requested abort without any reason iotd pointer

zero word count for mme gesnap

xxx fault occurred without user processing routine

where xxx is mme, mem, tag, zop, drl, luf, onc, ofl, div, tro or par.

COLTS Input Error Messages

COLTS INVALID INPUT MESSAGE

The following message is issued whenever the input following test is not in the correct format:

```
***colts executive (erroneous data) invalid input
(reason)
```

where (reason) is one of the following:

channel not implemented
: The channel number input is not configured on the communications processor.

colts system full
: There has been a new test request but there are already 16 queued entries. (Could be queued waiting for TOLTS to output error messages.)

invalid channel number
: The channel number input was nondecimal or greater than 15.

invalid option characters "qz"
: A test page communication or end request option qa-qp was detected with an invalid character following the "q".

invalid subchn number
: The subchannel number input was nondecimal, greater than 31 for HSLA type.

invalid 355 number
: The logical DATANET 355/6600 processor number was nonoctal or greater than the largest communications processor configured on the system.

no datanet 355 in system configuration
: Received a COLTS test request and there is no DATANET 355/6600 processor listed in the configuration (CDT).

no test program exists for that type device
: A test call was input for a hardware type for which COLTS does not have a test page.

subchannel not implemented
: The subchannel number input is not configured on the DATANET 355/6600 processor.

test entry not found
: A test page communication or end request was input and one of the following conditions exist:

  Active test page number option used (0-7), but rest of header did not match the header of the test page specified.

  Queued test entry option used (qa-qp), but the rest of the header did not match the header of the queued test specified.

  The header information does not match any entry.

unknown request for this executive
: The input does not match any of the valid inputs to COLTS.

COLTS INPUT ENTRY FORCE TERM MESSAGE

The COLTS input entry force terminate messages are issued whenever the input request is in the correct format but COLTS cannot honor the request at this time.

```
***colts executive (erroneous data) force term
(reason)
```

where (reason) is one of the following:

colts line disconnect
: The Multics communication system disconnected the executive DAC line between COLTS and COLTS/355 executives. The disconnect was probably caused by some internal data error from which the Multics communication system cannot recover. All current test entries not yet assigned a test page are terminated.

duplicate test request channel currently awaiting allocation
: The input request received was for a channel for which test allocation is already active.

duplicate test request channel currently under test
: The input request received was for a channel already under test.

entry time out connect to slave not received from 355
: The "accept new terminal" and "connect to slave" sequence from the DATANET 355/6600 processor was not completed in the allotted time. This should not occur unless a test request was made when the communications processor is down.

icm data checksum error
: A data checksum error was detected in the transmit or receive portion of the test allocation request icm with the COLTS/355 executive.

test e request received
: A test cenccss request has been received for an entry not as yet assigned a test page (the entry was found in the executive abeyance queue).

test line disconnect
: The Multics communication system disconnected the DAC test line established for testing on behalf of the entry being terminated. The disconnect was probably caused by some internal data error from which the Multics communication system cannot recover.

COLTS INPUT ENTRY WAITING MESSAGES

The COLTS input entry waiting messages are issued for a valid input request when COLTS is waiting for the resources necessary to process the request.

```
***colts executive (erroneous data) waiting
(reason)
```

where (reason) is one of the following:

waiting for subchannel to become inactive
: The subchannel designated in the test request is currently connected to a remote user. Allocation retry is attempted each minute if the subchannel is not available for test. However, the message is output only once. This response can also be received due to the subchannel being connected to a private line. In this case it may be necessary to physically disconnect the data set from the DATANET 355/6600 processor

in order to run the test. Use the test cenccss input to terminate entry if desired.

waiting for 355 memory
A subchannel was successfully allocated to COLTS and the COLTS executive is attempting to retransmit the DATANET 355/6600 processor HSLA overlay channel module to the communications processor. The Multics communication system does not currently have enough available memory to allow this transfer. Retry and message output is attempted every minute. A test cenccss input terminates the entry.

colts device test not available
This message results when a test request is received for a device for which COLTS has no test page.

***colts system full
This message results when a remote device test request is received and COLTS is currently full.

COLTS Illegal Operation Message

TEST PAGE ILLEGAL OPTION MESSAGE

The test page illegal option message is issued whenever an illegal option is detected.

```
**x(header) illegal option:
(reason)
o,o,o,o, enter options:
```

where:

1. o,o,o,o,
   is the current option set.

2. (reason)
   is one of the following reason messages:

cannot sequence tests outside of this segment
A test number input in response to a .seqt message was outside the range specified in the message.

cannot turn off forced term test
An invalid request was received to turn off the force termination test.

cannot turn off test in another segment
Test tables are an integral part of each segment, sequencing or turning off can only be done for the current segment.

illegal control mnemonic (.option) encountered
A request was received for an option which was not recognized by the option processor.

jump specified to a test not sequenced
A jump (a "-" in front of a test number) was specified in response to a .seqt message but the test number specified was not included in the sequence.

no executable tests in this sequence
The use of .seqt with a back jump and ntxxx options has set up a sequence with no tests that can be executed.

only numerics, commas, minus, or a blank allowed
Only test numbers separated by commas (with a possible initial "-" for jumping) are allowed as input to a .seqt message.

pass or recycle must be set to output error tallies
A .tal option was input but neither "p" nor "r" was set on.

specifying a nonexecutable test
One of the tests specified in response to a .seqt message has been marked as "do not run" for the reason specified in test page initialization message.

test number cannot be '0'
The test number was input as a zero, or no test number was input, or the "t" was the last character of the input line (including the initial test request or test communication request limitation of 11 characters).

test number is not in the current sequence
A test number was specified which lies outside (was not included in) the sequence specified in response to a .seqt message.

test number must follow 't'
The "t" option character was not followed by a one-, two-, or three-numeric test number.

too many entries input
More than the specified number of test numbers were input in response to a .seqt message.

trying to go to a nonexecutable test
The test specified was marked as do not run for the reason specified in the test page initialization message.

'x' is an illegal option character
An illegal option character was detected by the option processor. The 'x' is some character other than a, b, e, h, i, l, n, o, p, r, s, or t.

'.' must be the first option character
A control mnemonic was found embedded in an option string.

COLTS Test Page Error Message

The following message is the standard format of the first line of test page error output.

```
**x(header)yy zzz tt/ss dd-dd-dd ti.me c
(reason)
```

where:

1. x
   is the test page number 0-7.

2. yy
   is the the Multics communication system DAC test identifier.

3. zzz
   is the hardware identifier mnemonic.

4. tt
   is the current test number in decimal.

5. ss
   is the current subtest letter A-zz.

6. dd-dd
   is the date.

7. ti.me
   is the time of day.

8. c
   is the current cycle number.

9. (reason)
   is the test description (i.e., line two contains a statement of the function in error).

The remaining portion of a test page error message is not that of a COLTS standard. Each test page, depending on the type of hardware being tested, has a standard message format for its test functions. Refer to the appropriate test page documentation for the complete error message description.

APPENDIX A

FIRMWARE LOADING FACILITY

This section describes the Multics load_tandd_library command. The description contains the name of the command, discusses the purpose of the command, and shows the correct usage. Notes and examples are included as necessary for clarity.

Name: load_tandd_library

The load_tandd_library command loads MPC firmware, ITRs, MDRs, and T&D test pages from the Integrated, Firmware And Diagnostic (IFAD) Tape (formerly the Firmware Tape) into a keyed sequential file, with the name of tandd_deck_file (referred to as the "deckfile" in the remainder of this command description). In addition, MPC firmware modules are loaded into individual segments in a form acceptable to the generate_mst command, so that they may be written onto the BOS tape. Also, a DN6600 or DN6670 Binary Deck Tape may be loaded into the deckfile using the -fnp_tape control argument. This command also generates a printable ASCII segment which contains a directory of those modules loaded from the tape.

Usage

load_tandd_library tape_name {-control_args}

where:

1. tape_name
   specifies the name for the IFAD Tape to be used.

2. control_args
   determine which modules are to be selected from the tape and defines where they are to be stored. The control arguments may be chosen from the following:

   -config
   allows extensive tailoring of the deckfile by loading only those modules from the tape that conform to the current configuration.

   -copy <copy_vol> -copy_args, -cp <copy_vol> -copy_args
   produces a tailored copy of an IFAD tape for use with offline T&D. The control arguments preceding the -copy argument determine what modules from the input tape are written to the copy tape (i.e., if the -list argument is specified, the input tape is copied unchanged; if the -config argument is specified, only modules which pertain to the current configuration are written to the copy tape; if neither the -list nor -config argument is specified, only modules that are applicable to Multics are written to the copy tape). The copy_args pertain only to the <copy_vol> and may be chosen from the following:

   -density <value>, -den <value>
   sets the density of the copy tape to <value>. The allowable values for <value> are 6250, 11600, 800, 556, and 200. If the -density argument is not specified, the copy tape is set to the density of the input tape.

   -track <n>, -tk <n>
   where n is 7 or 9 (for 7- or 9-track tapes). If the -track argument is not specified, 9-track is assumed.

   -deckfile, -dkf
   specifies that no firmware modules are to be loaded into individual segments; only a deckfile is produced. This argument is mutually exclusive with the -firmware argument.

load_tandd_library

-density <value>, -den <value>
specifies the initial density setting for tape attachment. Although the density of the input tape is automatically determined, some tape subsystems may not have tape drives capable of handling the default density of 800 bpi. The allowable values for <value> are 6250, 1600, 800, 556, or 200.

-firmware, -fw
specifies that only firmware modules are to be loaded into individual segments; no deckfile or listing file is produced. This argument is mutually exclusive with the -deckfile, -list, -copy, and -fnp_tape arguments.

-fnp_tape
specifies that the input tape is not an IFAD tape but rather a DN6600 or DN6670 Binary Deck Tape which contains FNP tests for use by the PPAS T&D subsystem and the test_fnp online Multics command. Each FNP object deck image is read from the tape and written to the deckfile with a unique key that includes the fnp type (the fnp type is determined by hueristics from the first object deck on the tape). In addition, a catalog record is accumulated as each object deck image is read in (the catalog record is nothing more than an array of search key names), and written to the deckfile when the entire tape is read. This catalog record also has a unique key which includes the fnp type. A listing file is also produced, the format of which is slightly different than the IFAD listing file. The listing file has a name in the form <tape_name>.fnp.<fnp_type>.list.

-list, -ls
specifies that no modules are to be loaded from the tape; only a listing of the contents of the IFAD Tape is generated.

-output_dir path, -odr path
specifies the name of the directory in which the deckfile and the firmware module segments are to be created. If omitted, the user's current working directory is used.

-patches
allows patches with zero checksums to be accepted. The checksum is adjusted before the record is written to the deckfile.

-track <n>, -tk <n>
where <n> is 7 or 9 (for 7- or 9-track tapes). If the -track argument is not specified, 9-track is assumed.

Notes

The default action of this command, if no control arguments are specified, is to load all modules on the tape that are applicable to Multics into the deckfile and all firmware modules into individual segments.

Firmware segments are created (in memory image format) using a three component entry name:

fw.name.ident.fw_rev

where name is the name of the device, ident is taken from the MPC-assembler IDENT pseudo-op card, and fw_rev is the firmware revision.

Each module that is read from the tape and written to the deckfile is copied as-is in GCOS object deck format. Each object deck occupies one record of the keyed sequential deckfile. The record search key is formed from information obtained from the $ OBJECT card.

If the deckfile already exists in the specified directory, it is updated with the new modules read from the tape. This is done by first deleting records for which the record search key already exists and rewriting the contents of the record, leaving records in the file that do not have a corresponding entry read from tape unchanged.

Catalog records are built for all ITR and MDR tape files while the respective tape files are being read. These catalog records are merely a list of all of the search keys associated with each individual ITR or MDR tape file and are used by the respective ITR and MDR T&D drivers to determine test sequencing. Each catalog record is written to the deckfile immediately following each ITR or MDR tape file, and have record search keys in the form:

cata.itr.<mpc_name>.<fw_rev>.<file> -- for an ITR catalog

or

cata.mdr.<grp_name>.<file> -- for an MDR catalog

where:

<mpc_name>
is the name taken from the ident pseudo-op card image.

<grp_name>
is the name of the generic peripheral group and can be either tape, disk, print, or card.

<fw_rev>
is the revision of the mpc firmware in this file. For the urcmpc itr/firmware file, it is the revision of the common mpc firmware.

<file>
is the current IFAD tape file number.

load_tandd_library

A listing segment is created in the working directory containing a directory of ITR, MDR, Firmware, T&D test pages, catalog records loaded from the tape, and a corresponding entry for each deckfile entry. The listing segment has the name "tape_name.list".

APPENDIX B

FORMATTING DISKS WITH MTR

This appendix describes a procedure for formatting disk packs using MTRs. The procedure utilizes an annotated script which shows typical input and output. Because the formatting procedures differ for MSU0451 and MSU0500/0501 devices, separate scripts are provided. In the scripts, input typed by the user is preceded by an exclamation mark (!).

FORMATTING MSU0451 DISK PACKS

The following script shows how to run MTR tests 6 and 3 to format and test an MSU0451 disk pack, and to assign alternates to tracks found defective during testing.

1. Enter the Total OnLine Test System (TOLTS):

```
! bound_tolts_$tolts_

  ***tolts executive version 810301 on 820812 at 20.071
```

2. Enter the MPC OnLine Test Subsystem (MOLTS):

```
  ***enter "polts", "molts", "colts", "isolts", "quit", or "msg"

! ???  molts
```

3. List the disk configuration for the disk string in which formatting will be done (because of page constraints, the following message is not an exact copy of that which is displayed by the system):

```
! ??? test pcd dskb

  dsk configuration:

  dskb 451 16 units;  starting with device no. 17
   120xx  primary channel of 4 logical channels on mpc card mspb
   126xx  secondary channel of 4 logical channels on mpc card mspb
   024xx  secondary channel of 4 logical channels on mpc card mspa
   022xx  secondary channel of 4 logical channels on mpc card mspa
```

4. Enter MTR test 6 to format and test the MSU0451 device:

```
!  ???  test mmt12020t6
```

where "test mmt12020t6" is a sample of the input format "test mmtICCDDtT":

- mmt — identifies the MTR test package
- ICC — gives the IOM number (0 = IOM A, 1 = IOM B, etc) and channel number (in decimal) of a channel by which the device to be formatted can be addressed. It must be one of those shown in the output of "test pcd" in step 3. In the sample input above, "120" is IOM B, channel 20.

DD gives the device number (in decimal) of the device to be tested. In the sample input, it is device 20 (dskb_20).

T gives the number of the MTR test to be run. In this case, test 6 should be run to format/test a pack.

5. The following output describes steps taken by MTR test 6 to attach the disk drive and mount the pack for writing:

```
***molts executive versions 820601 820701 on 820812 at 19.97
**0(mmt12020) short wait, allocation queued
**0(mmt12020) start tmt65a-rmc1, ttldat 820331, phy./log. id t//04
**0(mmt12020) start tmt65b-rmc2, ttldat 820331, phy./log. id t//04
**0(mmt12020) start tmt65c-rmc3, ttldat 820401, phy./log. id t//04
**0(mmt12020) start tmt65d-rmc4, ttldat 820405, phy./log. id t//04
**0(mmt12020) start tmt65e-rmc5, ttldat 820421, phy./log. id t//04
**0(mmt12020) start tmt65f-rmc6, ttldat 820331, phy./log. id t//04
**0(mmt12020)
rmc6 is at your service to format a disk pack -
**0(mmt12020)
***** write permission granted *****
**0(mmt12020)
***** begin format pack *****
the test will format all tracks on the pack.  format will
defined by device type.  bad tracks will be marked defective
(no alt.  assigned).
**0(mmt12020)
system device code = .ds450
```

6. Answer MTR initialization questions (not a restart, normal formatting, and use 3 write patterns during testing):

```
    **0(mmt12020)
!   is this a restart?  enter (y or n) - n
    **0(mmt12020)
    select (f)ast or (n)ormal format?  (f)ast format is designed
    for data security erase and/or test purposes.  (n)ormal format
    is designed for disk packs that are going to be used in systems
!   applications.  enter (f or n) - n
    **0(mmt12020)
!   select from "1" to "7" write patterns?  enter (1 thru 7) - 3
```

7. At this point, formatting of the pack begins:

```
**0(mmt12020)
***** begin disk pack format *****
```

8. After the message in Step 7 is displayed, press the BREAK key to interrupt formatting operations. When MOLTS prompts for input, set test options to: report the current cylinder/head (CCC/HH) address; display CCC/HH for transient errors; report test progress every 100 cylinders, with summary reports attached.

```
!  <PRESS BREAK KEY>
!  ??? test momt12020.r
```

where "test momt12020.r" is a sample of the input format "test momtICCDD.O":

momt identifies request to set options

ICCDD are the IOM, Channel and Device numbers given in Step 4.

.r is the first option, to report current CCC/HH location.

Set the remaining options when prompted:

```
!  *0(mmt12020) t6 enter options: .i
!  *0(mmt12020) t6 enter options: .e
!  *0(mmt12020) t6 enter options: .s
```

```
!  *0(mmt12020) t6 enter options: .t
!  *0(mmt12020) t6 enter options: .go
```

9. When the .go option is entered in Step 8, MTR reports the current location being formatted and displays the defective tracks found. It then asks if formatting is to continue:

```
**0(mmt12020)
format function current addr. = 007/00
**0(mmt12020)
format function current addr. = 007/00
**0(mmt12020)
***** rmc6 - summary report *****
no tracks were formatted defective
**0(mmt12020)
!  do you want the test to continue?  enter (y or n) - y
```

10. After every 100 cylinders are formatted, MTR displays defective tracks found. For example, the final summary displayed just before formatting completes, looks like:

```
**0(mmt12020)
rmc6 has formatted tracks "000/00 thru 700/00"
**0(mmt12020)
***** rmc6 - summary report *****
no tracks were formatted defective
**0(mmt12020)
rmc6 has formatted tracks "000/00 thru 800/00"
**0(mmt12020)
***** rmc6 - summary report *****
no tracks were formatted defective
**0(mmt12020)
***** disk pack format complete *****
```

11. After formatting is complete, MTR begins testing the tracks on the formatted pack. Defective tracks are usually encountered only during the testing phase. Error summaries are displayed after every 100 cylinders have been tested.

```
start media test phase
**0(mmt12020)
rmc6 has tested tracks "000/00 thru 100/00"
**0(mmt12020)
***** rmc6 - summary report *****
no tracks were formatted defective
**0(mmt12020)
rmc6 has tested tracks "000/00 thru 200/00"
**0(mmt12020)
***** rmc6 - summary report *****
no tracks were formatted defective
**0(mmt12020)
rmc6 has tested tracks "000/00 thru 300/00"
**0(mmt12020)
***** rmc6 - summary report *****
defective - marginal data field on std track
217/10
**0(mmt12020)
***** rmc6 - summary report *****
defective - unrec. data field on std track
244/06,245/06
**0(mmt12020)
***** rmc6 - summary report *****
reclaimed - reformatted and certified
246/06
```

12. When testing is complete, termination summary reports are displayed:

```
**0(mmt12020)
***** normal termination summary reports *****
```

```
**0(mmt12020)
***** rmc6 - summary report *****
defective - marginal data field on std track
217/10
**0(mmt12020)
***** rmc6 - summary report *****
defective - unrec. data field on std track
244/06,245/06
**0(mmt12020)
***** rmc6 - summary report *****
reclaimed - reformatted and certified
246/06
```

13. MTR then asks if a new test is to be selected (answer "y" for yes):

```
**0(mmt12020)
want to select a new test?
! enter (y or n) - y
```

14. MTR then displays information describing how to select the next test:

```
**0(mmt12020)
rmc6 will go into waiting!
select test (t1 thru t6)
enter test no. thru standard option call (test momticcddtx) -
**0(mmt12020)
waiting
```

15. To actually select the next test, press the BREAK key and wait for the MOLTS prompt. Then select test 3, which assigns alternate tracks for those tracks found to be defective above.

```
! <PRESS BREAK KEY>
! ??? test momt12020t3
```

where "test momt12020t3" is a sample of the input format "test momtICCDDtT":

momt identifies request to set options

ICCDD are the IOM, Channel and Device numbers given in Step 4.

tT gives the number of the next test to run.

Test 3 initialization displays the following information:

```
**0(mmt12020) start tmt65e-rmc5, ttldat 820421, phy./log. id t//04
**0(mmt12020) start tmt65d-rmc4, ttldat 820405, phy./log. id t//04
**0(mmt12020) start tmt65c-rmc3, ttldat 820401, phy./log. id t//04
```

16. Select subtest 4 of test 3, to assign alternates to all defective tracks:

```
**0(mmt12020)
rmc3 is at your service
for track and cylinder reformat -
select a sub test
a) subtst 1 - reformat 1 track (good)
b) subtst 2 - reformat 1 cylinder (good)
c) subtst 3 - reformat 1 track (defective)
d) subtst 4 - assign alternate tracks
! enter (1 thru 4) - 4
```

17. MTR then briefly describes the subtest, and asks if the test is to continue (answer "y" for yes):

```
**0(mmt12020)
***** begin subtst 4 *****
assign alternate tracks on the device
a) subtst will search thru all standard tracks looking for
   tracks marked defective (no alternate assigned).
```

```
    b) when a track marked defective (no alternate assigned) is
       detected, the subtst will stop and process this track.
    c) the alternate track processor will go out to the alternate
       track cylinders and find the first available alternate.  it
       will mark the track as assigned alternate.  then it will
       mark the standard track as defective (alt.  assigned).
    d) the search process will terminate after the last standard
       track completes testing and/or processing.
  ! do you want the subtst to continue?  enter (y or n) - y
```

18. MTR then asks for permission to overwrite the pack's label (answer "y" for yes):

```
**0(mmt12020)
***** rmc3 - label obliterate warning *****
all sub tests in rmc3 will overwrite the system
label on track zero.
! do you want the sub test to continue?  enter (y or n) - y
```

19. MTR then asks if this is a restart (answer "n" for no):

```
**0(mmt12020)
system device code = .ds450
**0(mmt12020)
! is this a restart?  enter (y or n) - n
```

20. MTR then begins displaying summary reports after every 100 cylinders are checked for alternate assignments:

```
**0(mmt12020)
rmc3 has tested tracks "000/00 thru 200/00"
**0(mmt12020)
***** rmc3 - subtst 4 summary report *****
no alternate tracks were assigned
**0(mmt12020)
rmc3 has tested tracks "000/00 thru 300/00"
**0(mmt12020)
***** rmc3 - subtst 4 summary report *****
defective - alt assigned
 def          alt          def          alt
cyl/hd       cyl/hd       cyl/hd       cyl/hd
217/10       811/00       244/06       811/01
245/06       811/02
```

21. After alternate assignments are complete, MTR displays a summary report describing all alternates on the pack:

```
**0(mmt12020)
***** normal termination summary reports *****
**0(mmt12020)
***** rmc3 - subtst 4 summary report *****
defective - alt assigned
 def          alt          def          alt
cyl/hd       cyl/hd       cyl/hd       cyl/hd
217/10       811/00       244/06       811/01
245/06       811/02
```

22. MTR then asks if a new test is to be selected (answer "n" for no, and "quit" to exit from TOLTS).

```
**0(mmt12020)
want to select a new test?
! enter (y or n) - n
**0(mmt12020) normal term 1
***molts executive version 820701 off 820812 at 21.45 p.t. 119530

***enter "polts", "molts", "colts", "isolts", "quit", or "msg"
! ??? quit
```

```
***tolts executive version 810301 off 820812 at 21.375
r 21:37 1107.348 1162
```

FORMATTING MSU0500/MSU0501 DISK PACKS

The following script shows how to run MTR tests 6 and 7 to format and test an MSU0500 or MSU0501 disk drive, and to assign alternates to tracks found defective during testing.

The MPC normally treats an MSU0500 or MSU0501 disk drive as two separately addressable devices. However, MTR formats and tests both logical devices during a single invocation, referring to one as the "odd device" (e.g., dskc_27) and the second as the "even device" (e.g., dskc_28). MTR refers to both logical devices as a single "head assembly" or "hda".

1. Enter the Total OnLine Test System (TOLTS):

```
! bound_tolts_$tolts_

   ***tolts executive version 810301 on 820812 at 20.071
```

2. Enter the MPC OnLine Test Subsystem (MOLTS):

```
   ***enter "polts", "molts", "colts", "isolts", "quit", or "msg"
! ???  molts
```

3. List the disk configuration:

```
! ??? test pcd dskd

   dsk configuration:

   dskd 501 16 units;  starting withdevice no. 17
    128xx  primary channel of 4 logical channels on mpc card mspd
    030xx  secondary channel of 4 logical channels on mpc card mspc
```

4. Enter MTR test 6 to format and test the entire MSU0500 or MSU0501 device:

```
! ???  test mmt12827t6
```

where "test mmt12827t6" is a sample of the input format "test mmtICCDDtT":

- mmt — identifies the MTR test package
- ICC — gives the IOM number (0 = IOM A, 1 = IOM B, etc) and channel number (in decimal) of a channel by which the device to be formatted can be addressed.It must be one of those shown in the output of "test pcd" in step 3. In the sample input above, "128" is IOM B, channel 28.
- DD — gives the device number (in decimal) of the device to be tested. In the sample input, it is device 27 (dskc_27). Always give the device number of the "odd device" associated with the disk drive.
- T — gives the number of the MTR test to be run. In this case, test 6 should be run to format/test the drive.

5. The following output describes steps taken by MTR test 6 to attach the disk drive for writing:

```
***molts executive versions 820601 820701 on 820805 at 20.08
**0(mmt12827) short wait, allocation queued
**0(mmt12827) short wait, allocation queued
**0(mmt12827) start tmt67a-mtr1, ttldat 820401, phy./log. id t//04
**0(mmt12827) start tmt67b-mtr2, ttldat 820401, phy./log. id t//04
```

```
**0(mmt12827) start tmt67c-mtr3, ttldat 820402, phy./log. id t//04
**0(mmt12827) start tmt67d-mtr4, ttldat 820405, phy./log. id t//04
**0(mmt12827) start tmt67e-mtr5, ttldat 820421, phy./log. id t//04
**0(mmt12827) start tmt67f-mtr6, ttldat 820405, phy./log. id t//04
**0(mmt12827)
mtr6 is at your service to format a physical device -
**0(mmt12827)
***** write permission granted *****
**0(mmt12827)
***** begin upgrade/downgrade hda *****
the test will format all tracks on the hda.  the format will be
defined by device type.  bad tracks will be marked defective
(no alternate).
```

6. Answer MTR initialization questions (ok to format with 512 words per sector, not a restart, normal formatting, and use 3 write patterns during testing):

```
**0(mmt12827)
device pair are configured as msu0501's
the hda will be formatted in (512) words/sector.
is this correct?
! enter (y or n) - y
**0(mmt12827)
is this a restart?
! enter (y or n) - n
**0(mmt12827)
select (f)ast or (n)ormal format?
(f)ast format is designed for data security
erase and/or test purposes.
(n)ormal format is designed for hda's to be used
in systems applications.
! enter (f or n) - n
**0(mmt12827)
select from "1" to "7" write patterns?
! enter (1 thru 7) - 3
```

7. At this point, formatting of the pack begins:

```
**0(mmt12827)
***** begin hda format *****
```

8. After the message in Step 7 is displayed, press the BREAK key to interrupt formatting operations. When MOLTS prompts for input, set test options to: report the current cylinder/head (CCC/HH) address; display CCC/HH for transient errors; report test progress every 100 cylinders, with summary reports attached.

```
! <PRESS BREAK KEY>
! ??? test momt12827.t
```

where "test momt12020.r" is a sample of the input format "test momtICCDD.O":

momt identifies request to set options

ICCDD are the IOM, Channel and Device numbers given in Step 4.

.r is the first option, to report current CCC/HH location.

Set the remaining options when prompted:

```
! *0(mmt12827) t6 enter options: .e
! *0(mmt12827) t6 enter options: .s
! *0(mmt12827) t6 enter options: .i
! *0(mmt12827) t6 enter options: .r
! *0(mmt12827) t6 enter options: .go
```

9. When the .go option is entered in Step 8, MTR reports the current location being formatted and displays the defective tracks found. It then asks if formatting is to continue:

```
**0(mmt12827)
format function current addr. = 004/00
**0(mmt12827)
format function current addr. = 004/00
**0(mmt12827)
***** statistics from format of hda *****
summary for msu0501 devices (27/28)
no. of tracks with 1 defect skip  = 1
no. of tracks with 2 defect skips = 0
no. of tracks with 3 defect skips = 0
no. of new defect skips generated = 0
     total defect skips processed = 1
 odd device defective tracks      = 0
even device defective tracks      = 0
physical device defective tracks  = 0
                            total = 0
**0(mmt12827)
***** mtr6 - hda condition summary report *****
no tracks were marked defective.
**0(mmt12827)
do you want the test to continue?
! enter (y or n) - y
```

10. After every 100 cylinders are formatted, MTR displays defective tracks found. For example, the final summary displayed just before formatting completes, looks like:

```
**0(mmt12827)
mtr6 has formatted tracks "000/00 thru 800/00"
**0(mmt12827)
***** mtr6 - hda condition summary report *****
 --- msu0501 odd device report (27) ---
defective - error logging track info
006/19,028/19,284/01,370/12
**0(mmt12827)
***** mtr6 - hda condition summary report *****
 --- msu0501 even device report (28) ---
defective - error logging track info
008/19,026/19,096/05,174/16,381/06,736/09,778/19
**0(mmt12827)
***** statistics from format of hda *****
summary for msu0501 devices (27/28)
no. of tracks with 1 defect skip  = 33
no. of tracks with 2 defect skips = 2
no. of tracks with 3 defect skips = 0
no. of new defect skips generated = 0
     total defect skips processed = 37
 odd device defective tracks      = 4
even device defective tracks      = 7
physical device defective tracks  = 0
                            total = 11
**0(mmt12827)
***** hda format complete *****
```

11. After formatting is complete, MTR begins testing the tracks on the formatted pack. Error summaries are displayed after every 100 cylinders have been tested.

```
start media test phase
**0(mmt12827)
mtr6 has tested tracks "000/00 thru 100/00"
**0(mmt12827)
***** mtr6 - hda condition summary report *****
 --- msu0501 odd device report (27) ---
defective - error logging track info
006/19,028/19,284/01,370/12,816/04,818/04,832/04
**0(mmt12827)
***** mtr6 - hda condition summary report *****
 --- msu0501 even device report (28) ---
```

```
defective - error logging track info
008/19,026/19,096/05,174/16,381/06,736/09,778/19
**0(mmt12827)
***** mtr6 - hda condition summary report *****
 --- msu0501 even device report (28) ---
reclaimed - repaired data field
042/15
**0(mmt12827)
***** statistics from format of hda *****
summary for msu0501 devices (27/28)
no. of tracks with 1 defect skip  = 42
no. of tracks with 2 defect skips = 2
no. of tracks with 3 defect skips = 0
no. of new defect skips generated = 1
    total defect skips processed  = 47
 odd device defective tracks      = 7
even device defective tracks      = 7
physical device defective tracks  = 0
                            total = 14
```

12. When testing is complete, termination summary reports are displayed:

```
**0(mmt12827)
***** normal termination summary reports *****
**0(mmt12827)
***** mtr6 - hda condition summary report *****
 --- msu0501 odd device report (27) ---
defective - error logging track info
006/19,028/19,284/01,370/12,816/04,818/04,832/04
**0(mmt12827)
***** mtr6 - hda condition summary report *****
 --- msu0501 odd device report (27) ---
reclaimed - repaired data field
764/09,830/04
**0(mmt12827)
***** mtr6 - hda condition summary report *****
 --- msu0501 even device report (28) ---
defective - error logging track info
008/19,026/19,096/05,174/16,381/06,736/09,778/19
**0(mmt12827)
***** mtr6 - hda condition summary report *****
 --- msu0501 even device report (28) ---
reclaimed - repaired data field
042/15,762/09,818/08
**0(mmt12827)
***** statistics from format of hda *****
summary for msu0501 devices (27/28)
no. of tracks with 1 defect skip  = 42
no. of tracks with 2 defect skips = 2
no. of tracks with 3 defect skips = 0
no. of new defect skips generated = 5
     total defect skips processed = 51
 odd device defective tracks      = 7
even device defective tracks      = 7
physical device defective tracks  = 0
                            total = 14
```

13. MTR then asks if a new test is to be selected (answer "y" for yes):

```
**0(mmt12827)
want to select a new test?
! enter (y or ) - y
```

14. MTR then displays information describing how to select the next test:

```
**0(mmt12827)
mtr6 will go into waiting!
select test (t1 thru t7)
enter test no. thru standard option call (test momticcddtx) -
```

```
**0(mmt12827)
waiting
```

15. To actually select the next test, press the BREAK key and wait for the MOLTS prompt. Then select test 7, which assigns alternate tracks for those tracks found to be defective above.

    Test 7 assigns alternates for the complete head assembly, whereas test 3 (used in the procedure for formatting MSU0451 disks) only assigns alternates for a single logical device. Thus, test 3 would have to be run twice (once for the odd device and once for the even device) to assign alternates on an MSU0500 or MSU0501 disk.

```
!  <PRESS BREAK KEY>
!  ??? test momt12817t7
```

    where "test momt12827t7" is a sample of the input format "test momtICCDDtT":

    momt identifies request to set options

    ICCDD are the IOM, Channel and Device numbers given in Step 4.

    tT gives the number of the next test to run.

    Test 7 initialization displays the following information:

```
**0(mmt12827) start tmt67g-mtr7, ttldat 820405, phy./log. id t//04
```

16. Select subtest 1 of test 7, to assign alternates to all defective tracks:

```
**0(mmt12827)
mtr7 is at your service
for special physical device formatting -
select the subtst
a) subtst 1 - assign all alternate tracks
b) subtst 2 - create & write logging tracks
!  enter (1 thru 2) - 1
```

17. MTR then briefly describes the subtest, and asks if the test is to continue (answer "y" for yes):

```
**0(mmt12827)
***** begin subtst 1 *****
assign alternate tracks on the physical device the subtst will
search "all" standard tracks on the hda for defective (no alt.
assigned).  if any are found, it will assign the 1st available
alternate to them.
**0(mmt12827)
do you want subtst (1) to continue?
!  enter (y or n) - y
```

18. MTR then asks if this is a restart (answer "n" for no):

```
**0(mmt12827)
is this a restart?
!  enter 'y or n) - n
```

19. MTR then begins displaying summary reports after every 100 cylinders are checked for alternate assignments:

```
**0(mmt12827)
mtr7 has processed tracks "000/00 thru 100/00"
**0(mmt12827)
***** mtr7 - subtst 1 summary report *****
 --- msu0501 odd device report (27) ---
defective - alternate track assigned
 def          alt          def          alt
cyl/hd       cyl/hd       cyl/hd       cyl/hd
006/19       840/00       028/19       840/03
**0(mmt12827)
***** mtr7 - subtst 1 summary report *****
 --- msu0501 even device report (28) ---
defective - alternate track assigned
 def          alt          def          alt
cyl/hd       cyl/hd       cyl/hd       cyl/hd
008/19       840/01       026/19       840/02
096/05       840/04
```

20. After alternate assignments are complete, MTR displays a summary report describing all alternates on the pack:

```
**0(mmt12827)
***** normal termination summary reports *****
**0(mmt12827)
***** mtr7 - subtst 1 summary report *****
 --- msu0501 odd device report (27) ---
defective - alternate track assigned
 def          alt          def          alt
cyl/hd       cyl/hd       cyl/hd       cyl/hd
006/19       840/00       028/19       840/03
284/01       840/06       370/12       840/07
816/04       840/11       818/04       840/12
832/04       840/13
**0(mmt12827)
***** mtr7 - subtst 1 summary report *****
 --- msu0501 even device report (28) ---
defective - alternate track assigned
 def          alt          def          alt
cyl/hd       cyl/hd       cyl/hd       cyl/hd
008/19       840/01       026/19       840/02
096/05       840/04       174/16       840/05
381/06       840/08       736/09       840/09
778/19       840/10
```

21. MTR then asks if a new test is to be selected (answer "n" for no, and "quit" to exit TOLTS).

```
**0(mmt12827)
want to select a new test?
!  enter (y or n) -  n
**0(mmt12827) normal term 1
***molts executive version 820701 off 820806 at 00.27 p.t. 5185916

***enter "polts", "molts", "colts", "isolts", "quit", or "msg"
!  ???  quit

***tolts executive version 810301 off 820806 at 00.165
r 00:16 5188.584 1038
```

INDEX

CUT ALONG LINE

HONEYWELL INFORMATION SYSTEMS
Technical Publications Remarks Form

TITLE	MULTICS ONLINE TEST AND DIAGNOSTICS REFERENCE MANUAL	**ORDER NO.**	AU77-03
		DATED	MARCH 1984

ERRORS IN PUBLICATION

SUGGESTIONS FOR IMPROVEMENT TO PUBLICATION

Your comments will be investigated by appropriate technical personnel and action will be taken as required. Receipt of all forms will be acknowledged; however, if you require a detailed reply, check here. ☐

FROM: NAME _____ **DATE** _____

TITLE _____

COMPANY _____

ADDRESS _____

_____

PLEASE FOLD AND TAPE–
NOTE: U. S. Postal Service will not deliver stapled forms

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 39531 WALTHAM, MA02154

POSTAGE WILL BE PAID BY ADDRESSEE

HONEYWELL INFORMATION SYSTEMS
200 SMITH STREET
WALTHAM, MA 02154

ATTN: PUBLICATIONS, MS486

Honeywell

CUT ALONG LINE

FOLD ALONG LINE

FOLD ALONG LINE

Together, we can find the answers.

Honeywell

Honeywell Information Systems
U.S.A.: 200 Smith St., MS 486, Waltham, MA 02154
Canada: 155 Gordon Baker Rd., Willowdale, ON M2H 3N7
U.K.: Great West Rd., Brentford, Middlesex TW8 9DH **Italy**: 32 Via Pirelli, 20124 Milano
Mexico: Avenida Nuevo Leon 250, Mexico 11, D.F. **Japan**: 2-2 Kanda Jimbo-cho, Chiyoda-ku, Tokyo
Australia: 124 Walker St., North Sydney, N.S.W. 2060 **S.E. Asia**: Mandarin Plaza, Tsimshatsui East, H.K.

39984, 5C484, Printed in U.S.A.

AU77-03