

StorageTek Enterprise Library Software

VM Client Installation, Configuration, and Administration Guide

Release 7.3

E60733-02

September 2015

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

Preface	xiii
Audience	xiii
Documentation Accessibility	xiii
Related Documents	xiii
Conventions	xiii
What's New?	xix
1 Introduction	
Features	1-1
Data Flow	1-2
XAPI Client Interface to ACSLS Server	1-2
2 Preparing for Installation	
IBM VMSES/E	2-1
VM Client Installation Package	2-1
VM Client Installation Contents	2-1
Software and Hardware Requirements	2-2
Software Requirements	2-2
Hardware Requirements	2-2
MVS Requirements	2-2
Application Program Interface Verification	2-3
DASD Storage and User ID Requirements	2-3
3 Installing VM Client	
IBM VMSES/E	3-1
Summary of Installation Steps	3-1
Step 1: Determine VM Client Resource Requirements	3-2
Creating a PPF Override File	3-3
Step 2: Allocate VM Client Resources	3-3
Step 3: Install VM Client Product Files	3-3
Step 4: Build the VM Client Executable Code	3-4
Step 5: Create the VM Client Service Machine	3-5
Step 6: Customize VM Client Service Machine Files	3-5

Step 7: Test the VM Client.....	3-5
Step 8: Place the VM Client into Production	3-5
4 Installing VM Client Maintenance	
IBM VMSES/E.....	4-1
Summary of Installation Steps	4-1
Step 1: Prepare to Receive Maintenance	4-2
Step 2: Receive Maintenance.....	4-3
Step 3: Apply Maintenance	4-3
Step 4: Build New Levels	4-3
Step 5: Place the New Maintenance into Production	4-4
5 Starting the VM Client	
SMCBINT Module Parameters.....	5-1
TRACE Keyword Value Pair	5-1
OPERATOR Keyword Value Pair.....	5-2
MAXRC Keyword Value Pair.....	5-2
VM Client Command Files.....	5-2
SMCPARMS.....	5-2
SMCCMDS	5-2
VM Client Customer Exits.....	5-3
CP DETACH Support.....	5-3
6 VM Client Commands	
Issuing VM Client Commands	6-1
VM Client Commands.....	6-1
AUTHorize.....	6-1
Syntax	6-2
Parameters.....	6-2
Example.....	6-3
CMS.....	6-3
Syntax	6-3
Parameters.....	6-3
Example.....	6-3
COMMtest.....	6-3
Syntax	6-3
Parameters.....	6-4
Example.....	6-5
CP	6-5
Syntax	6-5
Parameters.....	6-5
Example.....	6-5
DISMount	6-5
Syntax	6-5
Parameters.....	6-6
Example.....	6-6

Display DRIve.....	6-6
Syntax	6-6
Parameters.....	6-7
Example.....	6-8
Display RC	6-8
Syntax	6-8
Parameters.....	6-8
Example.....	6-8
Display Volume.....	6-9
Syntax	6-9
Parameters.....	6-9
Example.....	6-9
DRIVemap.....	6-9
Syntax	6-9
Parameters.....	6-10
Example.....	6-10
DUMP	6-11
Syntax	6-11
Parameters.....	6-11
Example.....	6-11
DUMPOpts.....	6-11
Syntax	6-11
Parameters.....	6-12
Example.....	6-12
EXIT.....	6-12
Syntax	6-12
Parameters.....	6-12
Help.....	6-12
Syntax	6-13
Parameters.....	6-13
Example.....	6-13
List.....	6-13
Syntax	6-13
Parameters.....	6-14
LOGdisk.....	6-14
Syntax	6-14
Parameters.....	6-14
Example.....	6-15
MOunt.....	6-15
Syntax	6-15
Parameters.....	6-16
Example.....	6-17
MSGDef	6-17
Syntax	6-17
Parameters.....	6-17
Example.....	6-18
OPERator	6-18

Syntax	6-18
Parameters.....	6-19
Example.....	6-19
POOLmap.....	6-19
Syntax	6-19
Parameters.....	6-20
Example.....	6-20
READ	6-20
Syntax	6-20
Parameters.....	6-21
Example.....	6-21
RESYNChronize	6-21
Syntax	6-21
Parameters.....	6-21
Example.....	6-22
Route	6-22
Syntax	6-22
Parameters.....	6-22
Example.....	6-23
SERVer	6-23
Syntax	6-23
Parameters.....	6-24
Server Path Parameters	6-25
Example.....	6-26
TAPEPlex.....	6-26
Syntax	6-26
Parameters.....	6-27
Example.....	6-28
TCPip	6-28
Syntax	6-28
Parameters.....	6-28
tcpip parms	6-29
Example.....	6-30
TRace.....	6-30
Syntax	6-30
Parameters.....	6-30
Example.....	6-31

7 ELS Server Considerations

SMC HTTP Server Component	7-1
Scratch Subpools	7-1
VTCS Management Classes	7-2
VM:Tape Allocation Exit	7-2

8 Messages

Message Descriptions.....	8-1
---------------------------	-----

9 VM Client Tape Management Interface

TMS Responsibilities	9-2
User Interface	9-2
Tape Resource Allocation	9-2
Operator Interface	9-3
TMS Decision Points	9-3
TMS Initialization	9-3
Drive Allocation	9-3
Scratch Allocation	9-3
Volume Movement	9-3
Returning a Volume to Scratch Status	9-3
TapePlex Information Returned to the TMS	9-3
Configuration Information	9-4
Volume Status	9-4
Volume Location	9-4
Eligible Drives	9-4
Movement Status and Error Codes	9-4
LSM and ACS Status	9-4
Inter-user Communications Vehicle (IUCV) Considerations	9-5
Additional Considerations	9-6
TMS and VM Client Interaction	9-7
TMS to VM Client Initial Connection	9-7
Initial Connection Dialog	9-7
Drive Allocation	9-8
Allocation Interaction	9-8
Allocation Dialog	9-8
Termination of allocation Interface	9-9
Operation Message Processing	9-9
Operator Message Interaction	9-9
Operator Message Dialog	9-9
Termination of Operator Message Interface	9-10
PROP-Detected Dismount	9-10
Scenario A - Normal Dismount	9-11
Scenario B - Dismount Processed Automatically	9-11
ACSRQ Macro	9-12
ACSRQ Requests	9-12
ACSRQ Macro Syntax	9-13
DISMOUNT	9-14
Considerations	9-14
Syntax	9-14
Parameters	9-15
Request Response	9-16
EJECT	9-16
Considerations	9-16
Syntax	9-17
Parameters	9-17
Request Response	9-19

MOUNT	9-19
Considerations.....	9-20
Syntax	9-20
Parameters.....	9-20
Request Response.....	9-22
MOVE	9-22
Considerations.....	9-22
Syntax	9-22
Parameters.....	9-23
Request Response.....	9-25
QCAP	9-25
Considerations.....	9-25
Syntax	9-25
Parameters.....	9-25
Request Response.....	9-27
QCONFIG.....	9-27
Considerations.....	9-27
Syntax	9-28
Parameters.....	9-28
Request Response.....	9-29
QDRIVES.....	9-29
Considerations.....	9-29
Syntax	9-29
Parameters.....	9-30
Request Response.....	9-31
QDRLIST	9-31
Considerations.....	9-31
Syntax	9-31
Parameters.....	9-32
Request Response.....	9-34
QSCRATCH	9-34
Considerations.....	9-34
Syntax	9-34
Parameters.....	9-35
Request Response.....	9-36
QVOLUME.....	9-37
Considerations.....	9-37
Syntax	9-37
Parameters.....	9-37
Request Response.....	9-39
QVOLUME	9-39
Considerations.....	9-39
Syntax	9-39
Parameters.....	9-39
Request Response.....	9-40
SCRATCH	9-40
Considerations.....	9-40

Syntax	9-41
Parameters.....	9-41
Request Response.....	9-42
SELSCR	9-42
Considerations.....	9-42
Syntax	9-42
Parameters.....	9-43
Request Response.....	9-45
UNSCRATCH	9-45
Considerations.....	9-45
Syntax	9-45
Parameters.....	9-45
Request Response.....	9-46
Interface Data Areas.....	9-46
SLX Macro	9-47
SLX Macro Mapping.....	9-48
Cross-Reference	9-56
ACSINT Request DSECT	9-62
Cross-Reference	9-66
IUB Record Format	9-70
IUB - IUCV Request Block	9-70
Cross-Reference	9-73

A MEDia, RECtech, and MODel Values

Media Type (MEDia)	A-1
Recording Technique (RECtech)	A-3
Model Type (MODel)	A-6

B Diagnostics

Index

List of Examples

9-1	Initial Connection Dialog.....	9-7
9-2	PROP-Detected Dismount Scenario A - Normal Dismount	9-11
9-3	PROP-Detected Dismount Scenario B - Dismount Processed Automatically.....	9-11
9-4	SLX Record Format.....	9-48
9-5	ACSINT Record Format.....	9-62

List of Figures

1-1	VM Client Data Flow	1-2
6-1	AUTHorize command syntax	6-2
6-2	CMS command syntax	6-3
6-3	COMMtest command syntax	6-4
6-4	CP command syntax.....	6-5
6-5	DISMount command syntax.....	6-6
6-6	Display DDrive command syntax	6-7
6-7	Display RC command syntax.....	6-8
6-8	Display Volume command syntax	6-9
6-9	DRIVemap command syntax	6-10
6-10	DUMP command syntax.....	6-11
6-11	DUMPOpts command syntax	6-11
6-12	EXIT command syntax	6-12
6-13	Help command syntax	6-13
6-14	List command syntax	6-14
6-15	LOGdisk command syntax.....	6-14
6-16	MOunt command syntax	6-16
6-17	MSGDef command syntax.....	6-17
6-18	OPERator command syntax	6-19
6-19	POOLmap operator command	6-20
6-20	READ command syntax.....	6-21
6-21	RESYNChronize command syntax.....	6-21
6-22	Route command syntax.....	6-22
6-23	SERVer command syntax	6-24
6-24	TAPEPlex command syntax	6-27
6-25	TCPip command syntax.....	6-28
6-26	TRace command syntax	6-30
9-1	ACSRQ macro syntax	9-13
9-2	ACSRQ DISMOUNT request syntax.....	9-15
9-3	ACSRQ EJECT request syntax	9-17
9-4	ACSRQ MOUNT request syntax	9-20
9-5	ACSRQ MOVE request syntax.....	9-23
9-6	ACSRQ QCAP request syntax	9-25
9-7	ACSRQ QCONFIG request syntax.....	9-28
9-8	ACSRQ QDRIVES request syntax	9-29
9-9	ACSRQ QDRLIST request syntax.....	9-32
9-10	ACSRQ QSCRATCH request syntax	9-35
9-11	ACSRQ QVOLUME request syntax	9-37
9-12	ACSRQ QVOLUSE request syntax.....	9-39
9-13	ACSRQ SCRATCH request syntax.....	9-41
9-14	ACSRQ SELSCR request syntax	9-43
9-15	ACSRQ UNSCRATCH request syntax	9-45

List of Tables

2-1	VM Client Installation Media Contents	2-1
2-2	VSMC730A Minidisk Layout	2-3
3-1	VM Client Service Machine Files	3-5
9-1	SLS Macro - Parameter Matrix	9-47
9-2	SLX Macro Cross-Reference	9-56
9-3	ACSINT Macro Cross-Reference	9-66
9-4	IUB Request Block Reference	9-70
9-5	IUB Cross-Reference	9-73
A-1	Media Types.....	A-1
A-2	Media Type Defaults	A-3
A-3	Recording Techniques	A-4
A-4	Recording Technique Defaults.....	A-6
A-5	Model Types	A-6

Preface

This publication describes how to install, configure, and manage Oracle's StorageTek VM Client software.

Audience

This document is intended for storage administrators, system programmers and operators responsible for maintaining the VM Client.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at

<http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit

<http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit

<http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

Related Documents

Visit the Oracle Technical Network (OTN) at the following URL to access related documentation for StorageTek libraries, tape drives, and associated software and hardware:

<http://docs.oracle.com>

Conventions

The following text conventions are used in this document:

Typographic Conventions

Typographic conventions include the following:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.

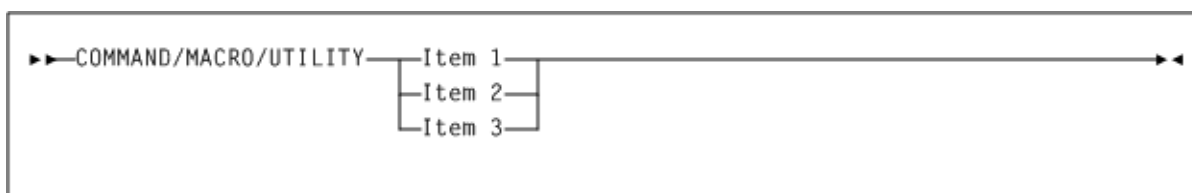
Convention	Meaning
<i>italic</i>	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
monospace	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Syntax Conventions

Syntax flow diagramming conventions include the following:

Flow Lines

Syntax diagrams consist of a horizontal base line, horizontal and vertical branch lines, and the text for a command, control statement, macro, or utility. Diagrams are read left to right, and top to bottom. Arrows indicate flow and direction. For example:



Single Required Choice

Branch lines (without repeat arrows) indicate that a single choice must be made. If one of the items to choose from is positioned on the baseline of the diagram, one item must be selected. For example:



Single Optional Choice

If the first item is positioned on the line below the baseline, one item may be optionally selected. For example:



Defaults

Default values and parameters appear above the baseline. For example:



Some keyword parameters provide a choice of values in a stack. When the stack contains a default value, the keyword and the value choices are placed below the base line to indicate that they are optional, and the default value appears above the keyword line. For example:



Repeat

A repeat symbol indicates that more than one choice can be made or that a single choice can be made more than once. The following example indicates that a comma is required as the repeat delimiter. For example:



Keywords

All command keywords are shown in all upper case or in mixed case. When commands are not case sensitive, mixed case implies that the lowercase letters may be omitted to form an abbreviation.

Variables

Italic type indicates a variable.

i

Alternatives

A bar (|) separates alternative parameter values.

Optional

Brackets [] indicate that a command parameter is optional.

Delimiters

If a comma (,), a semicolon (;), or other delimiter is shown with an element of the syntax diagram, it must be entered as part of the statement.

Ranges

An inclusive range is indicated by a pair of elements of the same length and data type, joined by a dash. The first element must be strictly less than the second element.

A hexadecimal range consists of a pair of hexadecimal numbers (for example, 0A2-0AD, or 000-0FC).

A decimal range consists of a pair of decimal numbers (that is, 1-9, or 010-094). Leading zeros are not required. The decimal portion is an incremental range. The character positions of the incremental portion of both range elements must match, and the non incremental characters of the first element must be identical to those of the second element.

A numeric VOLSER range (vol-range) consists of a pair of VOLSER elements containing a decimal numeric portion of 1 to 6 digits (for example, ABC012-ABC025, or X123CB-X277CB). The decimal portion is an incremental range. The following additional restrictions apply:

- The character positions of the incremental portion of both range elements must match.
- The non incremental characters of the first element must be identical to those of the second element.
- You cannot increment two portions of a range element. If 111AAA is the first element, you cannot specify 112AAB for the second element.
- If a VOLSER range contains more than one decimal portion, any portion is valid as the incremental range. For example:
 - A00B00 - The largest range that can be specified is A00B00 through A99B99.
 - A0B0CC - The largest range that can be specified is A0B0CC through A9B9CC.
 - 000XXX - The largest range that can be specified is 000XXX through 999XXX.

An alphabetic VOLSER range (vol-range) consists of a pair of VOLSER elements containing an incremental portion of 1 to 6 characters (for example, 000AAA-000ZZZ, or 9AAA55-9ZZZ55). This portion is an incremental range. The following additional restrictions apply:

- The character positions of the incremental portion of both range elements must match.
- The non incremental characters of the first element must be identical to those of the second element.
- You cannot increment two portions of a range element. If 111AAA is the first element, you cannot specify 112AAB for the second element.
- The alphabetic portion of the VOLSER range is defined as being from character A to Z. To increment multi-character sequences, each character increments to Z. For instance, ACZ is part of the AAA-AMM range. Examples are:
 - A00A0-A99A0
increments VOLSERs A00A0 through A09A0, then A10A0 through A99A0.
 - 9AA9A-9ZZ9A
increments VOLSERs 9AA9A through 9AZ9A, then 9BA9A through 9ZZ9A.
 - 111AAA-111ZZZ
increments VOLSERs 111AAA through 111AAZ, then 111ABA through 111ZZZ
 - 999AM8-999CM8
increments VOLSERs 999AM8 through 999AZ8, then 999BA8 through 999CM8
 - A3BZZ9-A3CDE9
increments VOLSERs A3BZZ9 through A3CAA9, then A3CAB9 through A3CDE9
 - AAAAAA-AAACCC
increments VOLSERs AAAAAA through AAAAAZ, then AAAABA through AAACCC

– CCCN–DDNN

increments VOLSERs CCCNN through CCCNNZ, then CCCNOA through DDNN. This is a very large range.

The number of volumes in an alphabetic VOLSER range depends on the number of elements in the incrementing portion of the VOLSER range. For an A to Z range in each character position, the number of volumes can be calculated by 26 to the power of the number of positions that are being incremented.

- A-Z is equivalent to 26^1 or 26 volumes.
- AA-ZZ is equivalent to 26^2 or 676 volumes.
- AAA-ZZZ is equivalent to 26^3 or 17,576 volumes.
- AAAA-ZZZZ is equivalent to 26^4 or 456,976 volumes.
- AAAAA-ZZZZZ is equivalent to 26^5 or 11,881,376 volumes.
- AAAAAA-ZZZZZZ is equivalent to 26^6 or 308,915,776 volumes.

Lists

A list consists of one or more elements. If more than one element is specified, the elements must be separated by a comma or a blank space, and the entire list must be enclosed in parentheses.

Blanks

Keyword parameters and values may be separated by any number of blanks.

Control Statement Conventions

The standard syntax conventions for control statements are as follows:

- The only valid control statement information area is from column 1 to column 72. Columns 73-80 are ignored.
- Parameters may be separated by one or more blanks or a comma.
- A value is associated with a parameter by an equal (=) sign or by enclosing the value in parentheses, and concatenating it immediately after the parameter.
- Case (upper or lower) in actual control statements is ignored.
- Continuations are supported by including a plus (+) sign at the end of the line to be continued. A control statement is terminated if the statement is not continued.
- Use /* and */ to enclose comments in the job stream. HSC PARMLIB members and definition data sets must specify comments in this format.
 - A comment is not required as the first control statement of any PARMLIB member.
 - Comments can be continued over multiple lines, but cannot be nested.
- The maximum length for any control statement is 1024 characters.

What's New?

This revision includes the following updates:

- VM Client now supports an XAPI client interface to an ACSLS server (Release 8.4 or later) with the XAPI service enabled.

See "[XAPI Client Interface to ACSLS Server](#)" for more information. Also, refer to the ELS publication *XAPI Client Interface to ACSLS Server Reference*.

- Updated the `Route` command description to note support for the XAPI client interface to an ACSLS server.

See "[Route](#)" for more information.

Introduction

This chapter describes VM Client software features and data flow.

Features

VM Client enables a client running on a VM system to request real and virtual tape services from a StorageTek TapePlex server executing on MVS.

Refer to the publication *Introducing ELS* to learn about the MVS software that enables you to manage the TapePlex server.

VM Client software provides the following features:

- IUCV interface to accept VM Tape Management Interface (VMTMI) requests

Oracle's StorageTek VM Client replaces VM/HSC as the component that provides an interface to allow VM Tape Management Systems (TMS) to use the StorageTek Automatic Cartridge System (ACS). In addition, the VM Client provides an interface to allow VM Tape Management Systems to use the StorageTek Virtual Storage Manager (VSM).

VM Client acts as the target for VMTMI requests originating from the VM TMS. All responses are returned to the TMS in VMTMI format. VM Client uses the VM Inter-user Communications Vehicle (IUCV) to communicate with the TMS service machines.

Not all VMTMI requests are supported by the VM Client. See [Chapter 9, "VM Client Tape Management Interface"](#) for a complete list of supported VMTMI requests.

- A TCP/IP interface to MVS based TapePlexes (HSCs)

StorageTek Enterprise Library Software (ELS) provides an XML interface (XAPI) to control the Storagetek ACS and VTCS systems. XAPI communications occur through TCP/IP. VM Client converts VMTMI requests to the new XAPI format. XAPI responses are converted to VMTMI responses.

Note: VM Client can only communicate with ELS 7.1 or later (SMC/HSC/VTCS).

- Operator commands to control the VM Client

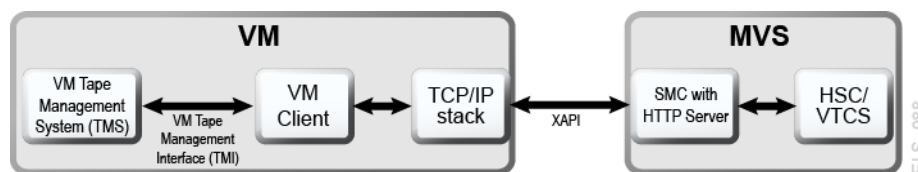
See [Chapter 6, "VM Client Commands"](#) for information about VM Client commands.

Data Flow

Figure 1–1 illustrates the following VM Client data flow:

1. A VM Tape Management System (TMS) request is sent to the VM Client through the VM Tape Management Interface (VMTMI).
2. The VM Client converts the VMTMI request to XAPI format and routes the request to the MVS server using TCP/IP communications.
3. SMC/HSC/VTCS software on the MVS server processes the request and returns all responses to the VM Client in XAPI format.
4. The VM Client converts the XAPI responses to VMTMI format and routes these responses to the TMS.

Figure 1–1 VM Client Data Flow



XAPI Client Interface to ACSLS Server

The XML API (XAPI) is Oracle's StorageTek API that enables StorageTek clients and servers to communicate using a common protocol over TCP/IP.

With the introduction of this XAPI, clients who were previously required to use an MVS based server (Oracle's StorageTek Host software Component) for real tape processing can now use ACSLS 8.4 or later (with XAPI support enabled) as follows:

- An SMC client on MVS can now request real tape requests from an ACSLS server with XAPI support enabled (without requiring MVS/CSC).
- A VM Client can now request real tape services from an ACSLS server with XAPI support enabled.

If you are using VM Client to connect to an ACSLS server with XAPI support enabled, you must use VM Client TAPEPlex and SERVER commands to define the ACSLS application as a TapePlex and define the TCP/IP control path between the client and server.

- See ["TAPEPlex"](#) for more information about the TapePlex command.
- See ["SERVER"](#) for more information about the SERVER command.

The majority of client/server interactions between VM Client and an ACSLS server with XAPI are transparent to the end user. Requests for volume information, mounts, and dismounts are generated automatically by the VM Client and are processed without operator intervention.

In addition to these automatic interactions, the ACSLS server with XAPI provides additional administrator, configuration, and operator commands that enable you to manage the XAPI component. Refer to the ELS publication *XAPI Client Interface to ACSLS Server Reference* for information about these commands.

Preparing for Installation

This chapter describes the VM Client installation package and its pre-installation requirements. It includes the following topics:

IBM VMSES/E

VM Client is installed using IBM's VMSES/E (Virtual Machine Serviceability Enhancements Staged/Enhanced), included as part of IBM VM/ESA.

Refer to the IBM publication *VMSES/E Introduction and Reference* for information about VMSES/E operation.

VM Client Installation Package

The VM Client installation package includes a VM Client installation ZIP file or CD-ROM (SERVLINK envelope) containing VM Client software.

As part of the installation, you must obtain and install the latest VM Client cumulative maintenance (PTFs and HOLDDATA).

Note: PTFs and HOLDDATA may not exist at ELS release launch but will follow in time and are released monthly to MOS.

Download cumulative maintenance from the My Oracle Support (MOS) site:

<http://www.myoraclesupport.com>

Visit this site frequently for HOLDDATA and PTF updates and install cumulative maintenance updates on a regular schedule. PTFs are released monthly to MOS.

See [Chapter 4, "Installing VM Client Maintenance"](#) for information about installing ELS cumulative maintenance.

VM Client Installation Contents

[Table 2–1](#) lists the files or SERVLINK groups included in the VM Client installation ZIP file or CD-ROM:

Table 2–1 VM Client Installation Media Contents

CD-ROM SERVLINK Group	Description
1	Header
2	Header

Table 2–1 (Cont.) VM Client Installation Media Contents

CD-ROM SERVLINK Group	Description
3	VSMC730A (Product Identifier File)
4	VSMC730A MEMO (Memo to Users)
5	Service Apply Lists (AXLIST)
6	PARTLISTS
7	Service (DELTA)
8	Service (APPLY)
9	Base Code (BASE)
10	Help Files (HELP)

Software and Hardware Requirements

VM Client software and hardware requirements are as follows:

Software Requirements

Operating System: Any IBM supported version of IBM z/VM

Independent Software Vendor Products (optional):

- ACF/VTAM
- CA-DYNAM/TLMS for z/VM
- DFSORT
- EPIC VSE
- Multi-Image Manager (MIM)
- SYNCSORT
- VM:Tape (see "[Application Program Interface Verification](#)" below)
- IBM Tape Manager for z/VM

Hardware Requirements

VM Client requires an IBM or compatible processor capable of running IBM z/VM (any IBM supported version).

MVS Requirements

VM Client communicates with a StorageTek TapePlex server executing on MVS. This server is managed by Oracle's StorageTek ELS software including SMC, HSC, and optionally, VTCS.

VM Client requires that all ELS software be at release 7.1 or later.

Refer to the publication *Installing ELS* for information about ELS hardware and software requirements.

Application Program Interface Verification

If you use VM:Tape as your tape management system, ensure that VM:Tape release 1.8 or later is installed.

Additionally, you must use the VM Client `AUTHorize` command to add VM:Tape to the VM Client authorized operators list. See ["AUTHorize"](#).

If any other tape management system is currently installed, you may need to write special routines to communicate with the VM Client. See [Chapter 9, "VM Client Tape Management Interface"](#) for more information about the interface to tape management systems, including recommended allocation and message processing, commands and responses, data areas, and interrupt handling required to communicate with the TapePlex.

DASD Storage and User ID Requirements

VM Client requires you to add both the VM Client installation user ID and VM Client service machine user ID to your VM directory. The requirements of these user IDs are further defined during the VM Client installation process.

The default VM Client installation user ID is VSMC730A. It is recommended that you use this default user ID to install and service VM Client.

If you choose to change the name of the VM Client installation user ID, you must create a Product Parameter Override (PPF). See ["Creating a PPF Override File"](#) for more information.

[Table 2–2](#) describes the VSMC730A minidisk layout:

Table 2–2 VSMC730A Minidisk Layout

Owner (User ID)	Default Address	Size (3390 Cyl)	Disk Name and Description
VSMC30A	2B2	20	BASE Contains all VM Client base code
VSMC30A	2C2	5	LOCALSAM Contains customization files.
VSMC30A	2D2	20	DELTA Contains services files.
VSMC30A	2A6	10	APPLY TEST Contains AUX files and software inventory tables that represent the test service level of VM Client.
VSMC30A	2A2	10	APPLY PRODUCTION Contains AUX files and software inventory tables that represent the production service level of VM Client.
VSMC30A	29D	10	BUILD4 Contains HELP files.
VSMC30A	201	20	BUILD1 Test build disk for VM Client. Contains load libraries and modules.

Table 2–2 (Cont.) VSMC730A Minidisk Layout

Owner (User ID)	Default Address	Size (3390 Cyl)	Disk Name and Description
VSMC30A	202	20	BUILD2 Production build disk for VM Client. Contains load libraries and modules.
VSMC30A	191	10	INST191 Installation user ID 191 minidisk
VSMC30A	191	10	VMSMC191 VM Client Service Machine 191 minidisk
VSMC30A	200	20	VMSMCRUN VM Client Service Machine RUN disk

Installing VM Client

This chapter describes how to install VM Client software.

Before installing ELS, verify ELS requirements and review pre-installation considerations described in [Chapter 2, "Preparing for Installation"](#).

After the VM Client is installed, you must obtain and install VM Client cumulative maintenance. See [Chapter 4, "Installing VM Client Maintenance"](#) for more information.

IBM VMSES/E

VM Client is installed using IBM's VMSES/E (Virtual Machine Serviceability Enhancements Staged/Enhanced), a component of IBM VM/ESA.

VMSES/E includes the VMFINS installation aid, designed to make installation of products consistent.

Refer to the IBM publication *VMSES/E Introduction and Reference* for more information about VMSES/E operation.

Summary of Installation Steps

The following is a summary of installation steps. Each step is described in detail in the sections that follow.

1. Determine VM client resource requirements.
Use the VMFINS command to load several VMSES/E files from the product SRVLINK file to obtain VM Client resource requirements.
2. Allocate VM client resources.
Use the information obtained from the previous step to allocate the appropriate minidisks and user IDs needed to install and use VM Client.
3. Install VM Client product files.
Use the VMFINS command to load the VM Client product files from the product SRVLINK file to the BASE minidisk.
4. Build the VM Client executable code.
Use the VMFINS command to build the VM Client test BUILD minidisk.
5. Create the VM Client service machine.
Create a directory entry for the VM Client Service Machine.
6. Customize VM Client service machine files.

Edit the VM Client Service Machine samples.

7. Test the VM Client.

Test the VM Client on the test build disk.

8. Place the VM Client into Production.

After VM Client testing, copy the VM Client files from the test build disk to the production build disk.

Step 1: Determine VM Client Resource Requirements

Use the `VMFINS` command to determine VM Client resource requirements.

1. Log on as the installer/planner.

Use any user ID with read access to `MAINT 5E5` and write access to the `51D` disk that will contain the VM Client software inventory.

2. Enter the following commands to establish read access to the `VMSES/E` code:

```
LINK MAINT 5E5 5E5 RR
ACCESS 5E5 B
```

3. Enter the following commands to establish write access to the Software Inventory disk, `MAINT 51D` in this example.

```
LINK MAINT 51D 51D M
ACCESS 51D D
```

Note the following:

- The Software Inventory disk can be the system Software Inventory disk. It is recommended that a `51D` disk be allocated to the maintenance/install user ID (`VSMC730A`).
- If another user is currently linked to the Software Inventory disk in write mode (R/W), the `LINK` command will fail. If this occurs, direct the other user to re-link to the Software Inventory disk in read-only mode (RR), and then re-issue the above `LINK` and `ACCESS` commands. Do not proceed until you establish a Read/Write link to the Software Inventory (`51D`) disk.

4. Load the VM Client product control files to the `51D` minidisk.

Enter the following:

```
VMFINS INSTALL INFO ( NOMEMO ENV VSMC730A
```

The `INSTALL INFO` command loads various product control files and creates the `VMFINS PRODLIST` file.

5. Obtain resource planning information for `VMCLIENT`.

Enter the following:

```
VMFINS INSTALL PPF VSMC730A VMCLIENT ( NOMEMO PLAN ENV
VSMC730A
```

The file, `VMFINS PLANINFO`, is created on the A-disk. This file contains information about the user IDs and minidisks required to install VM Client.

6. Review the install message log file, `$VMFINS $MSGLOG`. All installation messages are written to the installation user's A-disk. Correct any errors before proceeding.

Creating a PPF Override File

If the maintenance or service machine user IDs must be changed, use the following procedure to create a PPF override file:

1. At the following prompt, enter 1.

```
VMFINS2601R Do you want to create an override for :PPF VSMC730A
VMCLIENT :PRODID VSMC730A%VMCLIENT?
```

```
Enter 0 (No), 1 (Yes) or 2 (Exit)
```

2. At the following prompt, enter 0.

```
VMFMK02917R Do you want to use the defaults for this product?
```

```
Enter 0 (No), 1 (Yes) or 2 (Exit)
```

3. Update only the VM Client `INSTALL` User, only the VM Client `SERVER`, or both the VM Client `INSTALL` User and VM Client `SERVER` on the Make Override Panel. Press F3 to exit this panel.

4. Select 2 - Save as... and enter file name of the override file.

The override \$PPF and PPF files are copied to the D-disk (51D).

The override PPF now replaces the VSMC730A PPF file. Substitute this override PPF for all instances of VSMC730A in all remaining installation steps.

Step 2: Allocate VM Client Resources

Use the planning information in the VSMC730A `PLANINFO` file to create the VSMC730A user directory entry.

1. Create the VSMC730A user directory entry.

The VSMC720A user directory is located at the bottom of the `PLANINFO` file. These entries contain the links and privilege classes necessary for the VSMC730A user ID. Use the directory entry found in `PLANINFO` as a model for the VSMC730A directory entry.

2. Add the `MDISK` statements to the directory entry for VSMC720A. The minidisk layout can be found in the `PLANINFO` file.
3. Add the VSMC730A directory entry to the system directory. Change the password for VSMC730A from XXXXXX to a valid password, in accordance with your security guidelines.
4. Place the new directory online.

Step 3: Install VM Client Product Files

Use the `VMFINS` command to build the VM Client test `BUILD` minidisk.

1. Logon to the installation user ID VSMC730A, created in Step 2.
2. Create a `PROFILE EXEC` that contains the `ACCESS` commands for `MAINT 5E5` and `51D` minidisks.

```
XEDIT PROFILE EXEC A
==> input /**/
==> input 'access 5e5 b'
==> input 'access 51d d'
```

```
====> file
```

3. Execute the profile to access MAINT's minidisks.

```
PROFILE
```

4. Establish write access to the Software Inventory disk, if it is not already linked R/W.

```
LINK MAINT 51D 51D M
ACCESS 51D D
```

If another user is currently linked to the Software Inventory disk in write mode (R/W), the LINK command will fail. If this occurs, direct this user to re-link to the Software Inventory disk in read-only mode (RR), and then re-issue the above LINK and ACCESS commands. Do not proceed until you establish a Read/Write link to the Software Inventory (51D) disk.

5. Copy the VMSES/E files created on the installer/planner 191 disk to the VSMC730A 191 disk. This will place all the VM Client VMSES/E files in one location. The files to copy are:

- VSMC730A PLANINFO
- VSMC730A PRODLIST

6. Install VM Client.

Enter the following:

```
VMFINS INSTALL PPF VSMC730A VMCLIENT (NOMEMO NOLINK ENV
VSMC730A OVERRIDE NO
```

The NOLINK option indicates that VMFINS is not to link the maintenance minidisks, only access them if they are not already accessed.

The OVERRIDE NO option indicates that VMFINS is not to create an override PPF. If an override PPF was created above, replace VSMC730A with the name of the PPF created.

7. Review the install message log file, \$VMFINS \$MSGLOG. All installation messages are written to the installation user's A-disk. If necessary, correct any errors before proceeding.

Step 4: Build the VM Client Executable Code

Use the VMFINS command to build the VM Client test BUILD minidisk.

1. Enter the following command to build the test BUILD minidisk.

```
VMFINS BUILD PPF VSMC730A VMCLIENT (ALL NOLINK
```

2. Review the install message log (\$VMFINS \$MSGLOG). All installation message logs are written to the installation user's A-disk. Correct any errors before proceeding.

The following messages in the VMFINS BUILD log are normal:

- VMFBDC2178I
Object =.HELPMMSG cannot be rebuilt because it is not serviced
- VMFBDC2178I
Object =.HELPSMC cannot be rebuilt because it is not serviced
- VMFINB2173I

No verification exec found for this product

Step 5: Create the VM Client Service Machine

Create a directory entry for the VM Client Service Machine. The VM Client Service Machine must be able to issue the CP MSGNOH command (Privilege Class B).

Refer to the VSMC730A PLANINFO file for machine and minidisk requirements.

Step 6: Customize VM Client Service Machine Files

Copy the following files from the LOCALSAM minidisk to the VM Client service machine 191 minidisk (VMSMC191):

Table 3–1 VM Client Service Machine Files

Sample Name	Operational Name	Use
SMCPARMS SAMPLE	SMCPARMS (FILE)	Start parameters
SMCCMDS SAMPLE	SMCCMDS (FILE)	Start commands
SMCSTART EXEC	SMCSTART EXEC	VM Client start EXEC
SMCPRO SAMPLE	PROFILE EXEC	VM Client PROFILE EXEC

The SMCPARMS and SMCCMDS contain start parameters and commands. The file names of these files must match FIILDEF names in the SMCSTART EXEC.

See "VM Client Command Files" for information about updating VM Client parameter and command files.

Step 7: Test the VM Client

VM Client is now ready for testing. The executable code to be tested is on the VSMC730A 201 minidisk. The options to run the VM Client are:

- LINK to VSMC730A 201 and issue SMCSTART.
- Copy VSMC730A 201 to VSMC 200 minidisk and issue SMCSTART. The sample PROFILE EXEC (SMCPRO SAMPLE) assumes this option.

Step 8: Place the VM Client into Production

After VM Client testing, enter the following commands to copy the VM Client files from the test disk (201) to the production disk (202):

```
VMFSETUP VSMC730A VMCLIENT
```

```
VMFCOPY ** fm1==fm2 (PRODID VSMC720A%VMCLIENT SPRODID
VSMC730A%VMCLIENT OLDDATE REPLACE
```

fm1 is the file mode of the TEST build disk (BUILD1 - 201)

fm2 is the file mode of the PRODUCTION build disk (BUILD2 - 202)

The VM Client service machine will need to be modified to use the PRODUCTION version. The options are:

- LINK to VSMC730A 202 and issue SMCSTART.

- Copy `VSMC730A 202` to `VMSMC 200` minidisk and issue `SMCSTART`. The sample `PROFILE EXEC (SMCPRO SAMPLE)` assumes this option.

Installing VM Client Maintenance

This chapter contains instructions for installing VM client maintenance.

You must obtain and install the latest VM Client cumulative maintenance (PTFs and HOLDDATA). Download cumulative maintenance from the My Oracle Support (MOS) site at:

<http://www.myoraclesupport.com>

Visit this site frequently for HOLDDATA and PTF updates and install cumulative maintenance updates on a regular schedule. PTFs are released monthly to MOS.

Before attempting to install maintenance, contact Oracle Global Customer Services for information about the latest maintenance available. See "Preface" for information about contacting Oracle for assistance.

Note: PTFs and HOLDDATA may not exist at VM Client release launch but will follow in time and are released monthly to MOS.

IBM VMSES/E

VM Client cumulative maintenance is installed using IBM's VMSES/E (Virtual Machine Serviceability Enhancements Staged/Enhanced), a component of IBM VM/ESA.

VMSES/E includes the VMFINS installation aid, designed to make installation of products consistent.

Refer to the IBM publication *VMSES/E Introduction and Reference* for more information about VMSES/E operation.

Summary of Installation Steps

The following is a summary of maintenance installation steps. Each step is described in detail in the sections that follow.

1. Prepare to Receive maintenance.

Use the VMFMRDSK command to clear the alternate apply disk before receiving new maintenance., enabling you to easily remove maintenance if a serious problem is encountered.

2. Receive maintenance.

The VMFREC command receives maintenance and places it on the Delta disk.

3. Apply maintenance.

The VMFAPPLY command updates the VMSES/E version vector table (VVT), which identifies the service level of all the serviced parts. In addition, AUX files are generated from the VVT for parts that require them.

4. Build new levels.

The build task generates the serviced level of an object and places the new object on a test BUILD (201) disk.

5. Place new maintenance into production.

After being tested, the maintenance is placed into production by copying the new service to the production disk (202).

Step 1: Prepare to Receive Maintenance

Do the following:

1. Log on to the VM Client service ID VSMC730A.
2. Establish write access to the Software Inventory disk, if it is not already linked R/W.

```
LINK MAINT 51D 51D M
ACCESS 51D D
```

Note: If another user is currently linked to the Software Inventory disk in write mode (R/W), the LINK command will fail. If this occurs, direct this user to re-link to the Software Inventory disk in read-only mode (RR), and then re-issue the above LINK and ACCESS commands. Do not proceed until you establish a Read/Write link to the Software Inventory (51D) disk.

3. Establish the correct minidisk access order.

```
VMFSETUP VSMC730A VMCLIENT
```

VSMC730A is the PPF shipped with the product. If you have your own PPF override, you must substitute your PPF name for VSMC730A.

4. Receive the documentation. VMFREC with the INFO option loads the documentation and displays a list of all products in the package.

Enter the following:

```
VMFREC INFO ( ENV filename
```

5. Review the receive message log file, \$VMFREC \$MSGLOG, for warning and error messages.

```
VMFVIEW RECEIVE
```

6. Clear the alternate APPLY disk to ensure that you have a clean disk for new maintenance.

```
VMFMRDSK VSMC730A VMCLIENT APPLY
```

VSMC730A is the PPF shipped with the product. If you have your own PPF override, you must substitute your PPF name for VSMC730A.

7. Review the merge message log file, `$VMFMRD $MSGLOG`. If necessary, correct any errors before proceeding.

```
VMFVIEW MRD
```

Step 2: Receive Maintenance

Do the following:

1. Receive the maintenance.

Enter the following:

```
VMFREC PPF VSMC730A VMCLIENT ( ENV filename
```

2. Review the receive message log file, `$VMFREC $MSGLOG`. If necessary, correct any errors before proceeding.

```
VMFVIEW RECEIVE
```

Step 3: Apply Maintenance

Do the following:

1. Apply the new maintenance.

```
VMFAPPLY PPF VSMC730A VMCLIENT
```

This command applies the maintenance that you just received. The version vector table is updated with all serviced parts and all necessary AUX files are generated.

2. Review the apply message log file, `$VMFAPP $MSGLOG`. If necessary, correct any errors before proceeding.

Step 4: Build New Levels

Do the following:

1. Build the Build Status Table with serviced parts.

```
VMFBLD PPF VSMC730A VMCLIENT (STATUS
```

2. Review the build message log file, `$VMFAPP $MSGLOG`. If necessary, correct any errors before proceeding.

```
VMFVIEW BUILD
```

3. Rebuild VM Client serviced parts.

```
VMFBLD PPF VSMC730A VMCLIENT (SERVICED
```

4. Review the build message log file, `$VMFAPP $MSGLOG`. If necessary, correct any errors before proceeding.

```
VMFVIEW BUILD
```

Step 5: Place the New Maintenance into Production

After VM Client testing, copy the VM Client files from the test disk (201) to the production disk (202).

```
VMFSETUP VSMC730A VMCLIENT  
VMFCOPY ** fm1==fm2 (PRODID VSMC730A%VMCLIENT OLDDATE REPLACE
```

fm1 is the file mode of the TEST build disk (BUILD1 - 201).

fm2 is the file mode of the PRODUCTION build disk (BUILD2 - 202).

Starting the VM Client

This chapter describes how to start the VM Client software.

The VM Client executes in its own CMS virtual machine, called the VM Client service machine. [Chapter 3, "Installing VM Client"](#) describes the setup of the VM Client service machine, and the installation of the VM Client service machine software.

Once the VM Client service machine is installed, the VM Client is started by issuing the distributed `SMCSTART` command. The `SMCSTART` command is an EXEC that initializes the VM Client virtual machine environment, and then executes the `SMCBINT` module.

The `SMCBINT` module reads the VM Client command files, and initializes the VM Client environment to receive TMI commands from other virtual machines. See ["VM Client Command Files"](#) for more information about the VM Client command files.

SMCBINT Module Parameters

In addition to the VM Client command files, the `SMCBINT` module accepts optional command line parameters. The `SMCBINT` optional command line parameters allow VM Client `OPERATOR` and VM Client `TRACE` to be set before the VM Client command files are read, or to enable VM Client `MAXRC` processing at startup.

`SMCBINT` optional command line parameters are entered as keyword value pairs following the module name. For example, the following is an example of specifying `SMCBINT` parameters:

```
SMCBINT TRACE ON OPERATOR VMOPER MAXRC 4
```

Any errors encountered during command line parameter processing will result in termination of the VM Client initialization process.

The sections that follow describe each of the allowed keyword value pairs in detail.

TRACE Keyword Value Pair

TRACE{ON|OFF}

specifies whether VM Client trace processing is to be activated before VM Client command file processing. `ON` specifies that VM Client `TRACE` is to be enabled as early as possible.

OPERATOR Keyword Value Pair

OPERATOR *userid*

specifies whether a VM Client operator is to be set before VM Client command file processing. *userid* specifies the name of the virtual machine to receive VM Client messages

MAXRC Keyword Value Pair

MAXRC *nn*

specifies whether VM Client MAXRC processing is to be activated.

MAXRC processing determines whether the VM Client system is to be terminated on startup when the specified command return code is exceeded. If MAXRC is not specified, VM Client will always attempt to complete its initialization regardless of any startup command failure(s). This is the default behavior. The value *nn* specifies the highest allowed return code. If a VM Client command executed from the SMCPARMS or SMCCMDS data set exceeds this value, then the SMC0236 and SMC0237 messages will be produced, and the VM Client will be terminated. Allowable values are 0, 4, 8, and 12.

VM Client Command Files

VM Client commands can be specified in the SMCPARMS and SMCCMDS command files. During VM Client initialization, these command files are read and the commands included in these files are executed. By convention, the command files are named as follows:

- SMCPARMS FILE A1
- SMCCMDS FILE A1

The FILEDEFS for SMCPARMS and SMCCMDS in the SMCSTART EXEC can be modified if different file names are to be used.

The control statements in the VM Client command files must conform to the standard conventions for control statements as described in ["Control Statement Conventions"](#).

SMCPARMS

The SMCPARMS command file is read first. It is used for user-configured items that cannot be changed while the VM Client is active. SMCPARMS cannot be reprocessed using the READ command.

The following is a sample SMCPARMS member entry:

```
OPERATOR ID(nnnn)
LOGDISK ON
MSGDEF CASE(MIXED)
TCPIP TCPNAME(tcpname)
```

SMCCMDS

The SMCCMDS command file is used for user-configured items that can be changed while the VM Client is active. SMCCMDS can be reprocessed using the VM Client READ command. See ["READ"](#) for more information about this command.

The following is a sample SMCCMDS member entry:

```
TAPEPLEX NAME(tttttttt)
SERVER NAME(ssssssss) TAPEPLEX(tttttttt) PORT(pppp) +
IPADDRESS(nn.nn.nn.nn)
```

VM Client Customer Exits

The VM Client provides the following customer exits which may be implemented as CMS EXECs.

- SMCXIT00 Midnight Exit

If this exit is implemented, it executes each night at midnight.

See the installation sample `SMCXIT00.samp` for input parameters (if any), capabilities, and installation instructions.

- SMCXIT01 Command Authorization Exit

If this exit is implemented, it executes when a VM Client command or TMI request is received and the VM Client command or TMI request is not authorized by an appropriate VM Client `AUTHorize` command.

See "[AUTHorize](#)" for information about the VM Client `AUTHorize` command. See the installation sample `SMCXIT01.samp` for input parameters (if any), capabilities, and installation instructions.

Note: VMSES/E installation does not install these exits. Refer to the individual customer exit samples for installation instructions.

CP DETACH Support

The `SMCPROP EXEC` is provided for use as a `PRogrammable OPERator (PROP)` action routine to process all of the following type of messages, which CP sends to the system console:

```
TAPE raddr DETACHED....
```

VM (CP) causes a "Rewind Unload" command to be executed on any tape drive `DETACHED` by the CP commands `LOGOFF`, `FORCE`, or `DETACH`. This would leave any StorageTek `DETACHED` TapePlex volume in a "selected" state while still residing in an automated TapePlex transport. The volume would be unavailable to any requestor until removed (or `DISMOUNTED`) from the drive.

`SMCPROP EXEC` is an "action routine" which may be invoked from the VM `PROP` service to automatically issue VM Client `DISMOUNT` commands when a TapePlex transport is `DETACHED` from a virtual machine, enabling the volume to be available (that is, unselected) earlier than would otherwise be possible. `RTABLE SAMPLE` is provided to use the `SMCPROP` action routine with `PROP`. Refer to the IBM publication *CMS Planning and Administration* for information about the `PROP` service.

`SMCPROP EXEC` can also be used with `VMOPERATOR` to trap `DETACH` messages. `LOGTABLE SAMPLE` contains sample `VMOPERATOR LOGTABLE` statements to assist you. Consult `VMOPERATOR` documentation for information about tailoring the samples to your needs. The `SMCPROP EXEC` must be updated if the VM Client service machine user ID is not `VMSMC`.

`SMCPROP EXEC` must be available to the `PROP/VMOPERATOR` machine to operate properly.

VM Client Commands

This chapter describes VM Client operator commands and the methods used to issue them.

Issuing VM Client Commands

Use the following methods to issue VM Client commands:

- Specify VM Client commands in the SMCPARMS or SMCCMDS files to be processed at startup. See ["VM Client Command Files"](#) for more information.
- Send VM Client commands to the VM Client service machine using the CP Special Message (SMSG) facility.

Issue the following command from any virtual machine authorized to issue commands to the VM Client service machine:

```
CP SMSG userid command-string
```

where:

- *userid* is the name of the VM Client service machine defined in the CP directory.
- *command-string* is a character string containing any valid VM Client command.
- Log on to the VM Client service machine and issue commands from the connected console.

VM Client Commands

This section describes VM Client commands.

AUTHorize

The AUTHorize command enables you to identify VM user IDs that are authorized to execute TMI and VM Client command requests. This command also enables you to remove previously defined authorization entries.

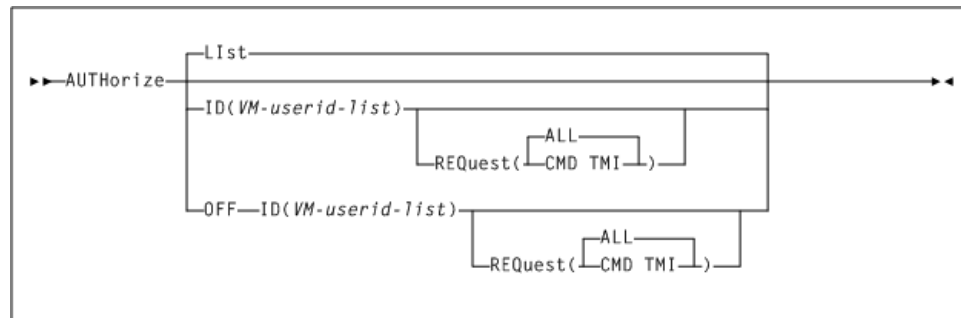
Note:

- The customer exit SMCXIT01 is provided to add override authorization capabilities to the VM Client. If an un-authorized VM user ID executes a VM Client command or TMI request, the SMCXIT01 exit may be used to override the absence of a matching AUTHorize command and provide the necessary authorization.
- There is no default SMCXIT01 EXEC exec installed as part of the VM Client installation.
- See the distributed SMCXIT01.samp file for a sample customer exit SMCXIT01 and installation instructions.

Syntax

The following figure shows syntax for the AUTHorize command:

Figure 6–1 AUTHorize command syntax

**Parameters**

As shown in [Figure 6–1](#), the AUTHorize command includes the following parameters:

List

optionally, lists all current AUTHorize mappings.

- List is the default when no other parameters are specified.
- List may be specified with other parameters. In this case, List is applied after all other parameters are processed.

ID(VM-userid_list)

optionally, specifies the VM user IDs to be authorized, as indicated in *VM-userid-list*.

OFF ID(VM-userid-list)

optionally, removes AUTHorize entries for the VM user IDs specified in *VM-userid-list*. A parameter value of "*" removes all AUTHorize entries.

REQuest (CMD | TMI | ALL)

optionally, specifies the VM Client privileges to be authorized or removed. You can specify this parameter with either the ID or OFF ID parameters. You must specify one or all of the following values:

- CMD indicates Authorize command requests received through the SMSG interface.
- TMI indicates Authorize Tape Management Interface API requests.

- ALL indicates Authorize CMD and TMI requests. This is the default if REQuest is not specified.

Example

In the following example, the AUTHorize command authorizes user ID VMTAPE to execute TMI requests, and list all AUTHORIZE entries:

```
AUTH REQ(TMI) ID(VMTAPE) LIST
```

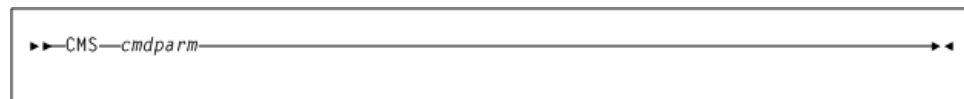
CMS

The CMS command enables you to transmit commands to the VM CMS (Conversational Monitor System) program environment without leaving VM Client.

Syntax

The following figure shows syntax for the CMS command.

Figure 6–2 CMS command syntax



Parameters

As shown in [Figure 6–2](#), the CMS command includes the following parameters:

cmdparm

any valid CMS command and parameter string.

Example

In the following example, the CMS command specifies a query of the file definitions in effect:

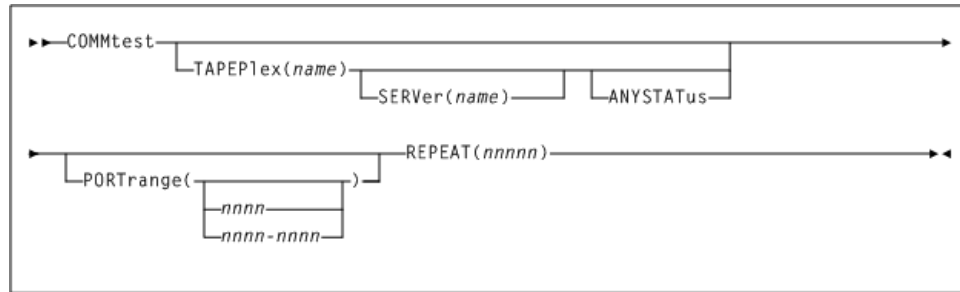
```
CMS QUERY FILEDEF
```

COMMtest

The COMMtest command enables you to test communications path(s) to one or more servers by executing a QUERY SERVER command to the specified communications path(s) and summarizing the results.

Syntax

The following figure shows syntax for the COMMtest command:

Figure 6–3 COMMtest command syntax

Parameters

As shown in [Figure 6–3](#), the `COMMtest` command includes the following parameters:

TAPEPlex(name)

optionally, specifies the TapePlex for the communication test. If you do not specify this parameter, communication is tested for all non-disabled TapePlexes.

name is the TapePlex name as defined by the VM Client `TAPEPlex` command. The following rules apply:

- The value must be between 1 and 8 characters in length.
- The first character must be either an alpha character or digit.
- The last character must be either an alpha character or digit.
- Any character between the first and last must be either an alpha character, digit, or hyphen.

You may specify the following subparameters:

SERVER(name)

optionally, specifies the server path for the communication test. If you do not specify this parameter, communication is tested for all non-disabled server paths for the named TapePlex.

name is the server path name as defined by the VM Client `SERVER` command. The following rules apply:

- The value must be between 1 and 8 characters in length.
- The first character must be either an alpha character or digit.
- The last character must be either an alpha character or digit.
- Any character between the first and last must be either an alpha character, digit, or hyphen.

ANYSTATUS

optionally, communication is tested for all communication paths, including paths that have been disabled by an operator command or by the VM Client.

PORTrange(nnnn/nnnn-nnnn)

optionally, specifies that communication for a remote server path be tested from the specified port or range. The specified `PORTrange` may be different from the `TCPip` `PORTrange` specification to allow testing of a firewall setup.

nnnn or *nnnn-nnnn* is the port number or port number range to be used for communication. Each port number can have a value of 1-65535. However, The maximum port number range that can be specified is 100 (for example, 6401-6500). If

omitted, a port in the defined `TCPip PORTrange` is used. If no such port is defined, any ephemeral port is used. If a port range is specified, then communication is attempted on each port number.

REPEAT (*nnnnnn*)

optionally, specifies the number of times to repeat the communication test. Valid values for *nnnnn* are 1 to 99999.

Example

In the following example, a user issues the `COMMtest` command to test the communications path to TapePlex `PRODHC1` using `SERVER PATHHC1`, and repeat the `XAPI QUERY SERVER` communication operation 100 times.

```
COMMTEST TAPEPLEX(RODHSC1) SERVER(PATHHSC1) REPEAT(100)
```

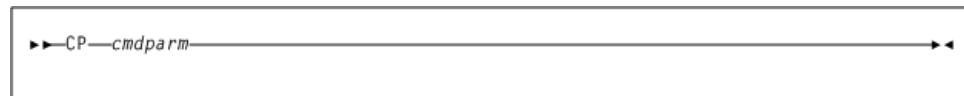
CP

The **CP** command enables you to transmit commands to the VM CP (Control Program) environment without leaving the VM Client.

Syntax

The following figure shows syntax for the CP command:

Figure 6–4 CP command syntax



Parameters

As shown in [Figure 6-4](#), the CP command includes the following parameters:

cmdparm

any valid CMS command and parameter string.

Example

In the following example, the `CP` command specifies a query of the `CPLEVEL` attributes in effect.

CP QUERY CPLEVEL

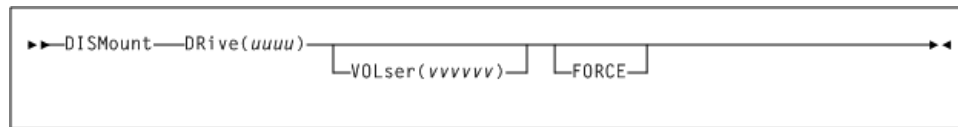
DISMount

The DISMount command dismounts a volume from a drive.

Syntax

The following figure shows syntax for the DISMount command:

Figure 6–5 *DISMMount command syntax*



Parameters

As shown in [Figure 6-5](#), the DISMOUNT command includes the following parameters:

DRive (uuuu)

specifies a tape drive address of the transport from which the volume is to be dismounted. This parameter is required.

uuuu is the tape drive address. If the `DRIVEMAP` command is used, this will be the `CLIENT` address. See "[DRIVemap](#)".

VOLser (vvvvvv)

specifies a tape drive address of the transport from which the volume is to be dismounted. This parameter is required.

vvvvvv is the tape drive address (volume serial number).

FORCE

optionally, specifies that the device is to be unloaded before the volume is dismounted. This parameter is not valid for virtual drives.

Example

In the following example, the `DISMOUNT` command dismounts volume AAA001 from drive 2900.

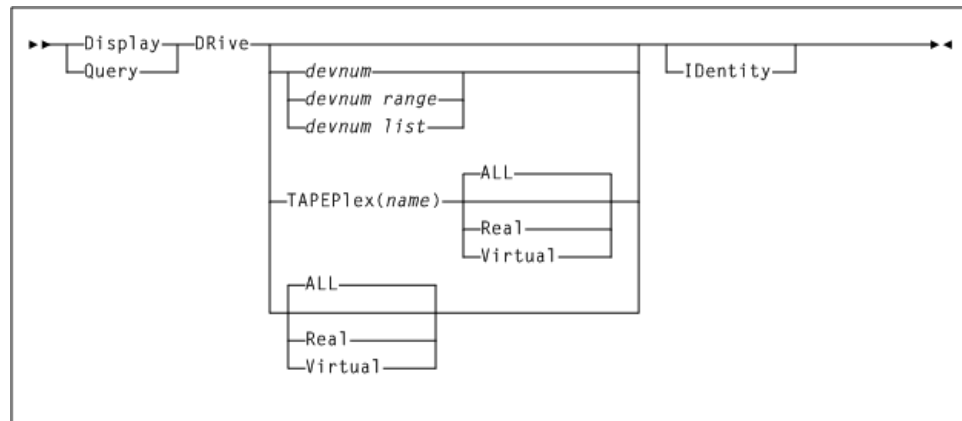
DISMOUNT DRIVE(2900) VOLSER(AAA001)

Display DRIve

The `Display Drive` command enables you to request VM Client drive attribute and TapePlex ownership information.

Syntax

The following figure shows syntax for the `Display Drive` command:

Figure 6–6 Display DRive command syntax

Parameters

As shown in [Figure 6–6](#), the Display DRive command includes the following parameters:

devnum, devnum-range, Or devnum-list

indicates a device number, range of device numbers, or list of device numbers to be displayed. Each device number must be a valid hexadecimal address in the format *ccuu*. If the DRIVEmap command is used this will be the CLIENT address(es).

TAPEPlex(name)

optionally, lists only devices owned by the specified TapePlex. If this parameter is not specified, then owned devices for all TapePlexes are displayed.

name is the TapePlex name. The following rules apply:

- The value must be between 1 and 8 characters in length.
- The first character must be either an alpha character or digit.
- The last character must be either an alpha character or digit.
- Any character between the first and last must be either an alpha character, digit, or hyphen.

You may also include the ALL, Real, or Virtual parameters to control the type of TapePlexes displayed.

ALL

optionally lists all devices owned by SMC defined TapePlexes. This is the default if no parameter is specified.

Real

optionally lists only "real" devices (that is, non virtual) owned by all defined TapePlexes.

Virtual

optionally lists only virtual devices owned by all defined TapePlexes.

IDentity

optionally, displays information identifying the drive serial number.

Example

In the following example, the `Display Drive` command lists only the “real” (that is, non virtual) devices known to the VM Client:

```
DISPLAY DRIVE REAL
```

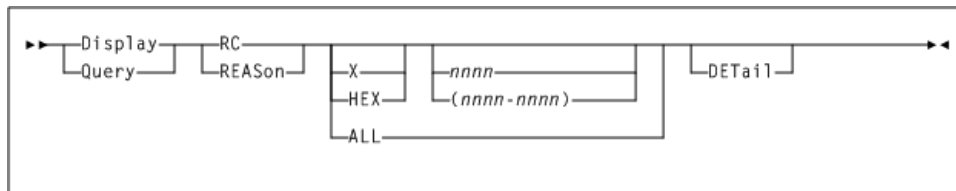
Display RC

The `Display RC` command enables you to display information about the meaning of an SMC return or reason code, or an HSC/VTCS UUI reason code.

Syntax

The following figure shows syntax for the `Display RC` command:

Figure 6–7 *Display RC command syntax*



Parameters

As shown in [Figure 6–7](#), the `Display RC` command includes the following parameters:

X or HEX

optionally, specifies that the reason/return code value or range is specified as a hexadecimal number.

nnnn or nnnn-nnnn

optionally, indicates a return code or list of return codes for which the explanation is to be displayed.

- If **X** or **HEX** is specified, the value may contain hexadecimal characters 0-9 and A-F.
- If **X** or **HEX** is not specified, the value may contain only numeric characters.

ALL

optionally, indicates that all defined return or reason codes are to be listed. **ALL** is permitted only from a utility.

Note: **ALL** and **X/HEX** are mutually exclusive.

DETAil

optionally, indicates that detailed information about the requested codes is to be listed.

Example

In the following example, the `Display RC` command displays information for SMC return code 302:

```
DISPLAY RC 302
```

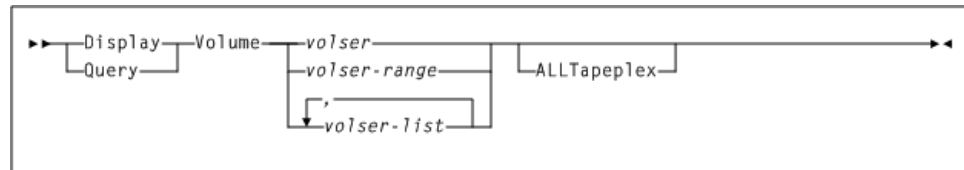

Display Volume

The `Display Volume` command enables you to request volume attribute and TapePlex ownership information.

Syntax

The following figure shows syntax for the `Display Volume` command:

Figure 6–8 *Display Volume command syntax*



Parameters

As shown in [Figure 6–8](#), the `Display Volume` command includes the following parameters:

volser, volser-range, Or volser-list

indicates the volser, volser range, or volser list to be processed. If multiple volumes are specified, only the first 100 are displayed.

ALLTapeplex

optionally, specifies that all active TapePlexes are queried for the specified volser(s). If specified, multiple display lines may be listed for the same volser if it is defined in multiple TapePlexes.

If this parameter is not specified, the `Display Volume` command queries TapePlexes in the order they are defined and lists only the first occurrence of the volume.

Example

In the following example, the `Display Volume` command lists volume serial number EVT100 that is found in any TapePlex accessible from the VM Client:

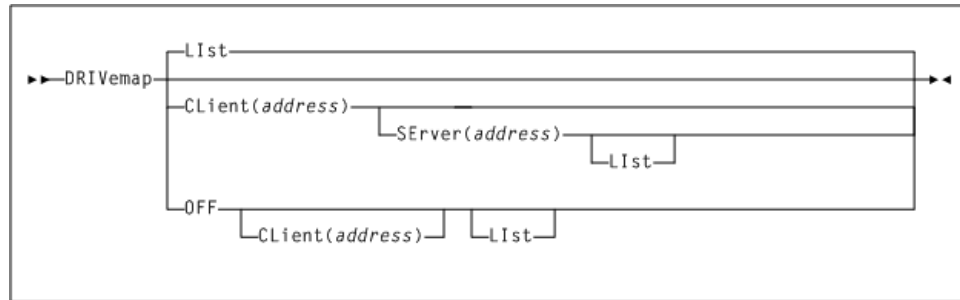
```
DISPLAY VOLUME EVT100 ALLTAPEPLEX
```

DRIVemap

The `DRIVemap` command enables you to map VM Client device addresses to server drive addresses. This command enables users to specify different addresses on the VM Client and on the server host for the same TapePlex real or virtual devices.

Syntax

The following figure shows syntax for the `DRIVemap` command:

Figure 6–9 *DRIVemap command syntax***Parameters**

As shown in [Figure 6–9](#), the `DRIVemap` command includes the following parameters:

List

optionally, lists all current `DRIVemap` mappings.

Client(address)

optionally, specifies the device numbers mapped by the `DRIVemap` command.

address is the device number, device number range, or device number list. Each device number is a hexadecimal number.

Additionally, you can include the `Server` sub-parameter:

Server(address)

optionally specifies the device numbers that are defined on the HSC server.

address is the device number, device number range, or device number list. The device number is a hexadecimal number.

- If `Client` is specified without the `OFF` parameter, then `Server` is required.
- If both `Client` and `Server` are specified, then the `Client` parameter must specify an equivalent address list or range as specified by the server parameter.

Note: Any VM Client commands entered that reference a device address (such as `DISPLAY DRIVE`, `DISMOUNT`, or `MOUNT`) must specify the client device address (or the address as known by the VM Client).

You can include the `List` parameter to list `DRIVemap` mappings for the specified device numbers.

OFF

optionally, removes all `DRIVemap` entries. If this parameter is specified with the `Client` parameter, then only the matching `Client` `DRIVemap` mappings are removed. The address specification (list or range) must match the defining specification exactly.

Example

In the following example, the `DRIVemap` command maps client device addresses 180-188 to server device addresses 280-288:

```
DRIVEMAP CLIENT(180-188) SERVER(280-288)
```

DUMP

The **DUMP** command enables you to force the production of a dump of the service machine storage at any time. All service machine storage is dumped. **DUMP** commands are for diagnostic purposes. Use only as directed by StorageTek Software Support.

The dump is produced through the CP **VMDUMP** command. The dump destination is the service machine's reader (class V).

Syntax

The following figure shows syntax for the **DUMP** command:

Figure 6–10 DUMP command syntax



Parameters

As shown in [Figure 6–10](#), the **DUMP** command includes the following parameters:

TITLE('comment')

optionally, describes the dump.

comment is the dump title, up to 72 characters in length and enclosed in single quotes. This title is only valid for this **DUMP** command. The default title is 'VM CLIENT DUMP COMMAND'.

Example

In the following example, the **DUMP** command specifies a dump of the VM Client service machine with the specified title:

```
DUMP TITLE('Sample Dump')
```

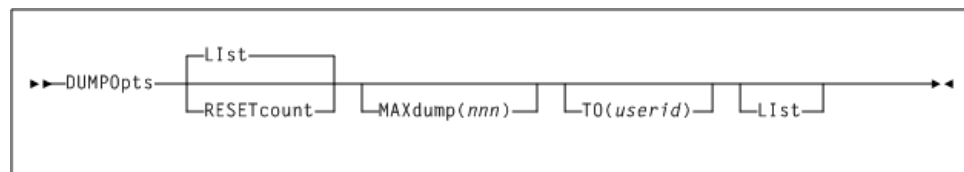
DUMPOpts

The **DUMPOpts** command enables you to specify or reset the maximum number of **VMDUMP** dumps to be generated. This command helps prevent VM spool space being exhausted in the unlikely event that a severe cycle of abnormal terminations occurs.

Syntax

The following figure shows syntax for the **DUMPOpts** command:

Figure 6–11 DUMPOpts command syntax



Parameters

As shown in [Figure 6–11](#), the `DUMPOpts` command includes the following parameters:

List

optionally, lists current `DUMPOpts` settings, including `DUMPS TAKEN`, `DUMPS MAX COUNT`, and `USERID`.

RESETcount

optionally, resets the number of dumps generated to zero.

MAXdump (nnn)

optionally, sets the threshold count for the number of dumps to allow before dump processing is disabled.

nnn is the number of dumps. This is a decimal number from 0 to 999. The default is 50.

TO(userid)

optionally, specifies the user ID to receive the dump.

userid is the user ID. This must be a defined VM user ID. If an asterisk (*) is entered, it is translated to the VM Client service machine id. The default is the user ID of the VM Client service machine.

Example

In the following example, the `DUMPOpts` command resets the number of dumps generated to zero and sets the threshold value to ten:

```
DUMPOPTS RESETCOUNT MAXDUMP(10)
```

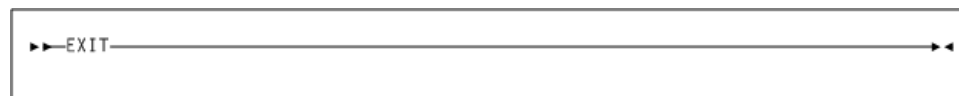
EXIT

The `EXIT` command enables you to terminate the VM Client service machine.

Syntax

The following figure shows syntax for the `EXIT` command:

Figure 6–12 *EXIT command syntax*



Parameters

None.

Help

The `Help` command enables you to display VM Client command and message information.

Note: If you enter the `Help` command without any parameters, information is displayed for all available VM Client commands.

Syntax

The following figure shows syntax for the `Help` command:

Figure 6–13 *Help command syntax*



Parameters

As shown in [Figure 6–13](#), the `Help` command includes the following parameters:

`command-name`
optionally, a VM Client command name.

`nnnn`
optionally, the four-digit numeric portion of a VM Client message identifier. Leading zeros are not required.

`nnnn-nnnn`
optionally, a range of VM Client messages specified using the four-digit numeric portion of the message identifier.

`SMCnnnn`
optionally, a full VM Client message identifier.

`SMCnnnn-SMCnnnn`
optionally, a range of VM Client messages specified using full message identifiers.

Example

In the following example, the `Help` command displays information for VM Client message `SMC0228`.

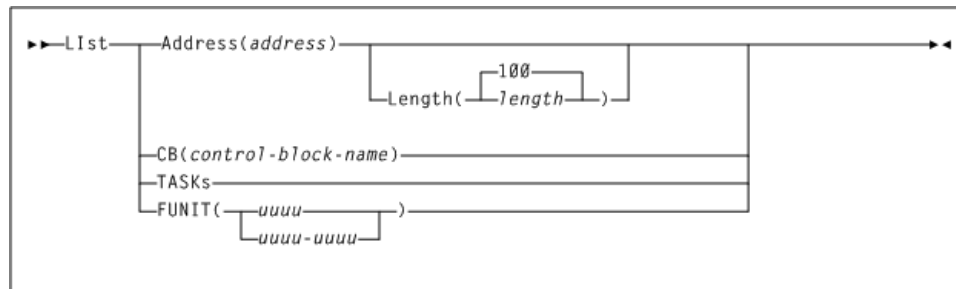
```
HELP SMC0228
```

List

The `List` command enables you to display storage contents within the VM Client virtual machine. This command is for diagnostic purposes. Use only as directed by StorageTek Software Support.

Syntax

The following figure shows syntax for the `List` command:

Figure 6–14 *List command syntax***Parameters**

As shown in [Figure 6–14](#), the `List` command includes the following parameters:

Address (address)

optionally, specifies the address at which to begin listing VM Client memory contents.

address is a hexadecimal address.

CB (control-block-name)

optionally, specifies the internal VM Client control block to be listed.

control-block-name is the control block name.

VM Client control blocks are listed for diagnostic purposes. Specify *control-block-name* only as directed by StorageTek Software Support

TASKs

optionally, lists the active VM Client system tasks.

FUNIT (uuuu | uuuu-uuuu)

optionally, lists VM Client control blocks associated with the specified unit address(es).

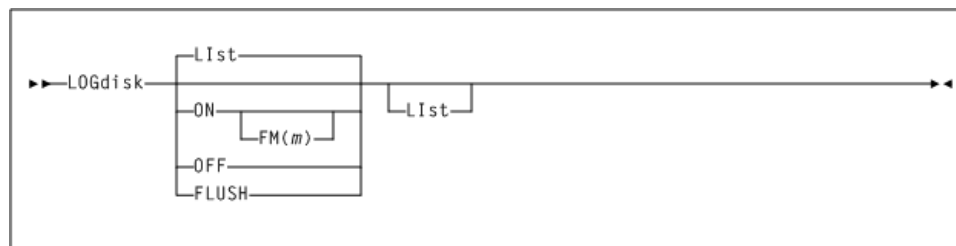
uuuu or *uuuu-uuuu* is a single unit address or range of unit addresses.

LOGdisk

The `LOGdisk` command enables you to control logging of console output to a disk file named `YYYYMMDD LOG`.

Syntax

The following figure shows syntax for the `LOGdisk` command:

Figure 6–15 *LOGdisk command syntax*

Parameters

As shown in [Figure 6–15](#), the LOGdisk command includes the following parameters:

LIst

optionally, displays current LOGdisk settings.

- LIst is the default when no parameters are specified on the LOGdisk command.
- LIst may be specified with other parameters. In this case, the LIst is applied after the other parameters are processed.

ON

optionally, enables the logging of console output with the listed options. When logging is on, all commands and messages are logged.

Additionally, you can enter the following parameter:

FM(*m*)

specifies the File Mode to receive the disk log file. The FM must specify a RW minidisk. This parameter is only valid with the ON parameter.

m is the file mode. This value must be an alphabetic character. The default is 'A'.

OFF

optionally, disables the logging of console output with the listed options. The log file is closed.

FLUSH

optionally, flushes the log file. The file is closed and reopened.

All messages are written to the VM Client service machine console. The handling of the VM Client service machine console can be controlled by the CP SPOOL command. It is recommended that the VM Client service machine console be started in the PROFILE EXEC and spooled to a maintenance ID. For example:

```
CP SPOOL CON START TO MAINT
```

Example

In the following example, the LOGdisk command enables logging to a disk file:

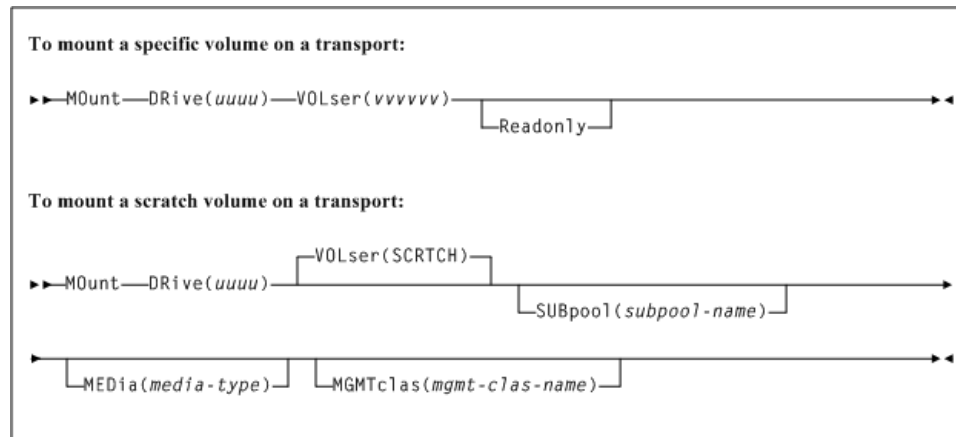
```
LOGDISK ON
```

MOunt

The MOunt command enables you to mount a volume on a drive.

Syntax

The following figure shows syntax for the MOunt command:

Figure 6–16 *MOunt command syntax*

Parameters

As shown in [Figure 6–16](#), the **MOunt** command includes the following parameters:

DRive(uuuu)

specifies a tape drive address of the transport on which the volume is to be mounted.

uuuu is the tape drive address. If the **DRIVemap** command is used, this is the **CLIENT** address.

VOLser(vvvvvv)

optionally, specifies the volume to be mounted. If this parameter is not specified, a scratch volume is mounted.

vvvvvv is the volume serial. Specify **SCRATCH** for a scratch volume.

Readonly

optionally, specifies that the volume is to be mounted for read-only access. This parameter is only valid for a specific mount.

SUBpool(subpool-name)

optionally, specifies that a scratch volume is to be taken from a scratch subpool. If this parameter is not specified, then the behavior depends on how scratch pools are defined on the HSC server. See the **HSC MOUNT** command description for details. This parameter is only valid for a scratch mount.

subpool-name is the subpool name.

MEDia(media-type)

optionally, specifies the type of media for the scratch volume. The specified media must be compatible with the request **DRive**. This parameter is only valid for a scratch mount.

media-type is the media type. See [Appendix A, "MEDia, RECtech, and MODel Values"](#) for a list of valid media-type values.

Note: If **MEDia** is not specified, the next scratch volume is selected without regard to media type.

MGMTclas(mgmt-clas-name)

optionally, specifies a Management Class defined in the HSC/VTCS `MGMTclas` control statement. This parameter is only valid for a scratch mount.

mgmt-clas-name is the Management Class name.

Example

In the following example, the `MOunt` command mounts volume AAA001 on drive 2900:

```
MOunt DRIVE(2900) VOLSER(AAA001)
```

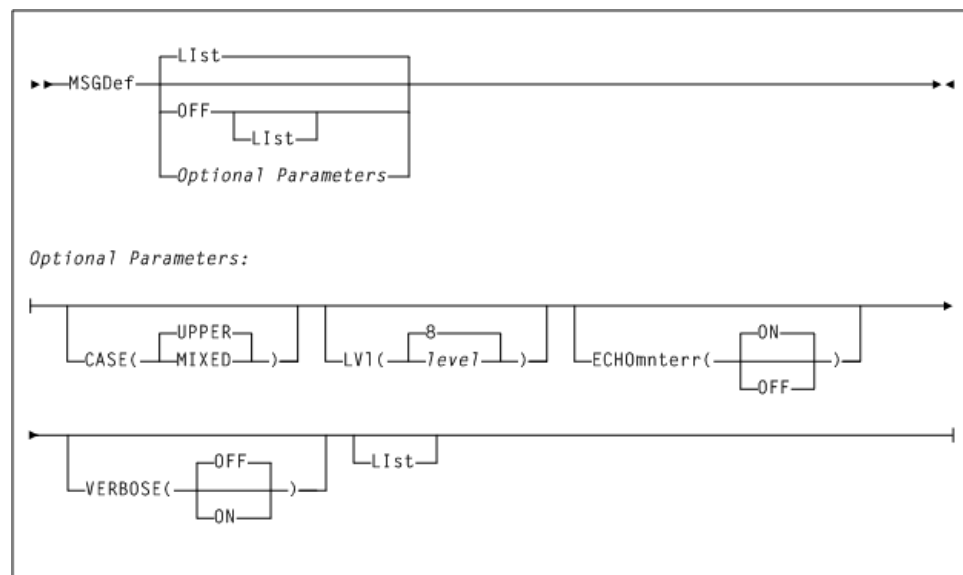
MSGDef

The `MSGDef` command defines the appearance of VM Client system messages, and controls which messages are displayed and suppressed.

Syntax

The following figure shows syntax for the `MSGDef` command:

Figure 6–17 *MSGDef command syntax*



Parameters

As shown in [Figure 6–17](#), the `MSGDef` command includes the following parameters:

List

optionally, lists current default VM Client message settings.

- `List` is the default when no other parameters are specified on the `MSGDef` command.
- `List` may be specified with other parameters. In this case, the `List` is generated after the other parameters are processed.

OFF

optionally, resets all `MSGDef` values to original VM Client default settings. Specify `List` with this parameter to list these settings.

CASE (UPPER | MIXED)

optionally, specifies the message case. Valid values are UPPER or MIXED.

- UPPER specifies upper case. This is the default.
- MIXED specifies mixed case.

LV1 (level)

optionally, specifies the default level used to control which VM Client messages are displayed and suppressed.

level is the default level. Valid values include the following:

- 0 - Display error messages only.
- 4 - Display error and warning messages from the VM Client service machine.
- 8 - Display all VM Client service machine messages and allocation job log warning messages. This is the default if the MSGDef parameter is not specified.

Note: Levels higher than 8 are used for diagnostic purposes and should only be specified as directed by StorageTek Software Support.

ECHOmnterr (ON | OFF)

optionally, specifies whether mount errors generated by the HSC are echoed directly to the console for the VM Client.

- ON specifies that Mount errors generated by the HSC are echoed to the console for the VM Client. This is the default.
- OFF specifies that Mount errors generated by the HSC are not echoed to the console for the VM Client.

VERBOSE (OFF | ON)

optionally, specifies whether SMC0190 and SMC0191 messages are displayed whenever VM Client settings are altered.

- OFF specifies that SMC0190 and SMC0191 messages are displayed.
- ON specifies that SMC0190 and SMC0191 messages are not displayed. This is the default.

Example

In the following example, the MSGDef command specifies that messages appear in mixed case, and that only error and warning messages from the VM Client service machine are displayed:

```
MSGD CASE(MIXED) LV1(4)
```

OPERator

The OPERator command specifies the virtual machine to receive VM Client messages.

Syntax

The following figure shows syntax for the OPERator command:

Figure 6–18 OPERator command syntax**Parameters**

As shown in [Figure 6–8](#), the OPERator command includes the following parameters:

List

optionally, displays current operator settings.

- List is the default when no other parameters are specified for the OPERator command.
- List may be specified with other parameters. In this case, the list is generated after the other parameters are processed.

ID(VM-userid)

optionally, specifies the name of the virtual machine to receive VM Client messages.

VM-userid is the user ID of the virtual machine. This must be a defined VM user ID. If an asterisk (*) is entered, it is translated to the VM Client service machine id. The default is the user ID of the VM Client service machine.

Example

In the following example, the OPERator command specifies the OPER machine to receive messages:

```
OPERATOR ID(OPER)
```

POOLmap

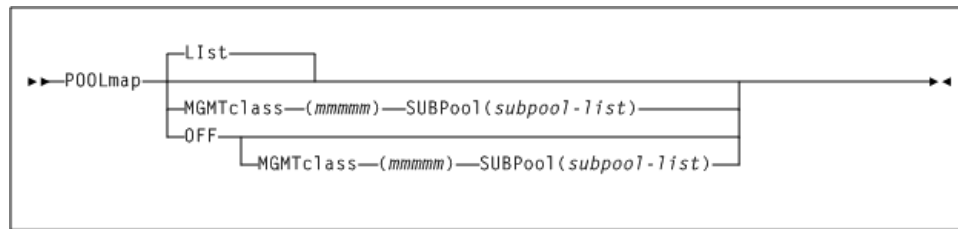
The POOLmap command enables you to map an HSC scratch subpool name to a VTCS management class.

Tape management systems that use the VM/HSC Tape Management Interface (VMTMI) usually specify only a subpool name for scratch requests and do not specify a management class. The POOLmap command provides a method of supplying a management class name for scratch mounts. The POOLmap command is especially recommended when VM Client is requesting virtual tape mounts.

Note: The POOLmap command validates the specified subpool and management class names by communicating with the TapePlex server. Therefore, POOLmap commands should not be specified before any VM Client TAPEplex and SERVer commands are processed.

Syntax

The following figure shows syntax for the POOLmap command:

Figure 6–19 POOLmap operator command**Parameters**

As shown in [Figure 6–19](#), the POOLmap command includes the following parameters:

List

optionally, displays current operator settings.

- **List** is the default when no other parameters are specified for the OPERator command.
- **List** may be specified with other parameters. In this case, the list is generated after the other parameters are processed.

MGMTclass (mmmm)

optionally, specifies the management class name defined on the HSC server.

mmmm is the 1-8 character alphanumeric management class name.

OFF

optionally, removes all POOLmap entries.

If this parameter is specified with the MGMTclass or SUBPool parameters, then only the matching POOLmap entries are removed.

SUBPool (subpool-list)

specifies the scratch subpool names defined on the HSC server.

subpool-list is one or more scratch subpool names to be associated with the specified management class name.

Example

In the following example, the POOLmap command maps the management class, DAILY, to the scratch subpools, VIRT CART1 and VIRT CART2:

```
POOLMAP MGMT(DAILY) SUBP(VIRT CART1, VIRT CART2)
```

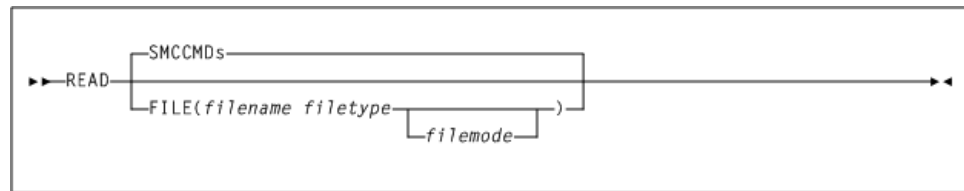
READ

The READ command enables you to enter a series of commands using an input data set instead of console commands.

Syntax

The following figure shows syntax for the READ command:

Figure 6-20 *READ command syntax*



Parameters

As shown in [Figure 6-20](#), the READ command includes the following parameters:

SMCCMDs

optionally, re-processes commands contained in the data set specified in the `SMCCMDS` `FILEDEF` of the VM Client START procedure.

FILE(*filename filetype*) and optionally, *filemode*.

optionally, specifies the file to READ.

- *filename* is the file name.
- *filetype* is the file type.
- *filemode* is the file mode. The default is A.

Example

In the following example, the READ command processes commands in the SMCCMDS FILEDEF of the VM Client startup EXEC.

READ SMCCMDS

RESYNChronize

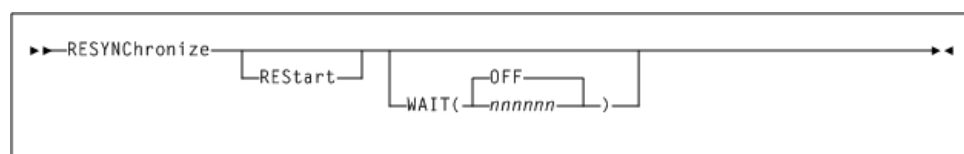
The `RESYNChronize` command enables you to reestablish connections to all defined TapePlexes to acquire drive configuration information from all TapePlexes.

This action is automatically performed when the VM Client first activates a new path to a TapePlex, or when an HSC server reports a configuration change.

Syntax

The following figure shows syntax for the RESYNChronize command:

Figure 6–21 *RESYNChronize command syntax*



Parameters

As shown in [Figure 6-21](#), the `RESYNChronize` command includes the following parameters:

REStart

optionally, starts the RESYNChronize attempt at the first server, regardless of the last active path.

WAIT (OFF | *nnnnnn*)

optionally, waits for a server to become available. This option is useful after the TAPEPLEX and SERVERs have been defined. The command will not complete until a server becomes available or the specified time, *nnnnnn*, has expired.

- *nnnnnn* is the wait time in minutes, from 0-999999.
- OFF specifies that the command does not wait for an available server. This is the default.

Example

In the following example, the RESYNChronize command specifies to restart communications from the first server:

```
RESYNC RESTART
```

In the next example, the RESYNChronize command specifies to wait for a server to become available:

```
RESYNC WAIT(9999)
```

Route

The Route command enables you to request routing of transactions from VM Client to a defined TapePlex name.

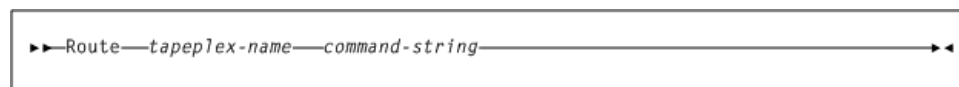
Note: You can also use the Route command to issue various commands from a VM Client to an ACSLS XAPI server.

Refer to the ELS publication *XAPI Client Interface to ACSLS Server Reference* for more information.

Syntax

The following figure shows syntax for the Route command:

Figure 6–22 Route command syntax

**Parameters**

As shown in [Figure 6–22](#), the Route command includes the following parameters:

tapeplex-name

a TapePlex name as defined on an VM Client TAPEPLEX command. VM Client routes the request to the specified TapePlex using the currently active TapePlex path.

command-string

the command string to be routed to the requested TapePlex.

- The VM Client will not attempt to validate the specified command string, but simply routes the command string as entered to the specified *tapeplex-name* and displays any responses.
- VTCS commands should not be prefixed with VT; the HSC UII interface routes VTCS commands to the correct functional processor without the VT prefix.
- The *command-string* must be a command supported by the HSC UII (except for VOLRPT) or any VTCS command (except for VTVRPT, DISPLAY MSG, and DISPLAY CMD).

Example

In the following example, the `Route` command routes the “D CDS” command string to TapePlex HSC8 for processing. Any responses received will be displayed by the SMC0173 message:

```
R HSC8 DI CDS
```

SERVer

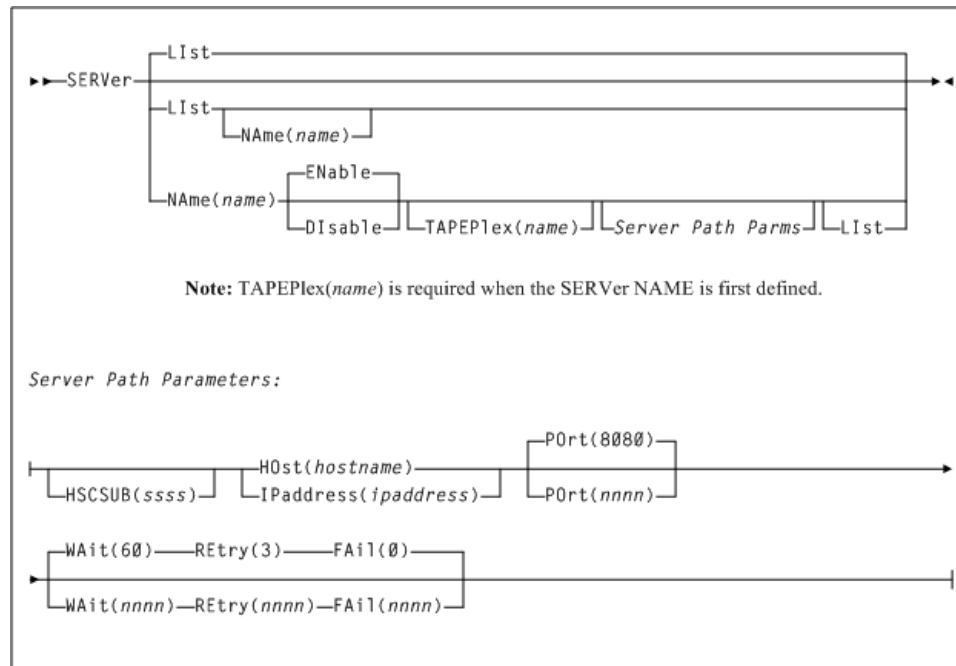
The `SERVer` command defines a named path to a remote library server. The `SERVer` command describes the communication path to a StorageTek HTTP server. The `SERVer` command can also list Servers defined to the VM Client.

Note the following:

- Before a `SERVer` is defined the TapePlex that it references must be defined using the `TAPEPlex` command.
- The TapePlex name associated with a `SERVer` cannot be changed. See ["TAPEPlex"](#) for more information.

Syntax

The following figure shows syntax for the `SERVer` command:

Figure 6–23 *SERVER command syntax*

Parameters

As shown in Figure 6–23, the SERVER command includes the following parameters:

LIST

optionally, displays status information for TapePlex server paths.

- LIST is the default when no parameters are specified on the SERVER command. In this case, all library server paths are listed.
- LIST may be specified with other parameters. When specified with parameters other than NAME, the LIST is generated after the other parameters are processed.

Optionally, you can also specify NAME (name) with this parameter. NAME specifies a TapePlex server path for which status is displayed. name is the server path name.

NAME (name)

optionally, specifies the communication path or route to the TapePlex server.

name is an identifier for the path parameters. This name is reported in any communication error messages. The following rules apply:

- The value must be one to eight characters in length.
- The first character must be an alpha character or digit.
- The last character must be either an alpha character or digit.
- Any character between the first and last must be either an alpha character, digit, or hyphen.

ENABLE

optionally, enables the specified server path to be selected for mount requests.

DISABLE

optionally, disables the specified server path. If this is the only path to the TapePlex, the TapePlex is unavailable for mount requests.

TAPEPlex(*name*)

optionally, specifies the TapePlex name associated with the ACS hardware configuration. The `TAPEPlex` parameter must be specified when a new server is defined.

name is the TapePlex name. This name is reported in any TapePlex server error messages. The following rules apply:

- The value must be one to eight characters in length.
- The first character must be an alpha character or digit.
- The last character must be either an alpha character or digit.
- Any character between the first and last must be either an alpha character, digit, or hyphen.

Note: You can define multiple paths to a single TapePlex.

Server Path Parameters**HSCSUB(*ssss*)**

optionally, specifies the name of the HSC subsystem that represents the TapePlex associated with the server. This parameter is required only when there is more than one HSC subsystem executing on the server host (HSC running in MULT mode).

ssss is the HSC subsystem name.

HOST(*hostname*)

optionally, specifies the IP resolver host name of the TapePlex server. For DNS lookup, the VM Client must have access to the `TCPIP DATA` file.

hostname is the name of the remote host.

Note: `HOST` and `IPaddress` are mutually exclusive.

IPaddress(*ipaddress*)

optionally, specifies the IP address of the TapePlex server.

ipaddress is the IP address of the remote host.

Note: `IPaddress` and `HOST` are mutually exclusive.

PORT(*nnnn*)

optionally, specifies the server port.

nnnn is the server port, from 0-65535. The default is 8080.

WAIT(*nnnn*)

optionally, specifies the maximum default wait time for requests before the VM Client times out the request.

nnnn is the wait time in seconds, from 0-9999. The default is 60.

Note: The default wait time does not apply to mount, dismount, eject, or move requests which have a default timeout value of 10 minutes, 10 minutes, 24 hours, and 1 hour respectively.

REtry (nnnn)

optionally, specifies the number of retry attempts for any single request before the task is allowed to resume, and a failure recorded.

nnnn is the number of retries, from 0-9999. The default is 3.

FAil (nnnn)

optionally, specifies the maximum number of failures after successful communication is established, before the specific server path is disabled or placed out of service.

nnnn indicates the number of failures. The default is 0.

If 0 is specified, the named SERVER will never be automatically disabled due to communications errors.

This value should be specified if there are no backup SERVER paths to a named library.

The FAIL limit count only applies after successful communication has been established on this SERVER path.

Example

In the following example, the `SERVER` command adds a server named `DENVER1` for TapePlex `DENVER`.

```
SERVER NAME(DENVER1) TAPEPLEX(DENVER) IP(11.22.33.44) PORT(7777)
```

TAPEPlex

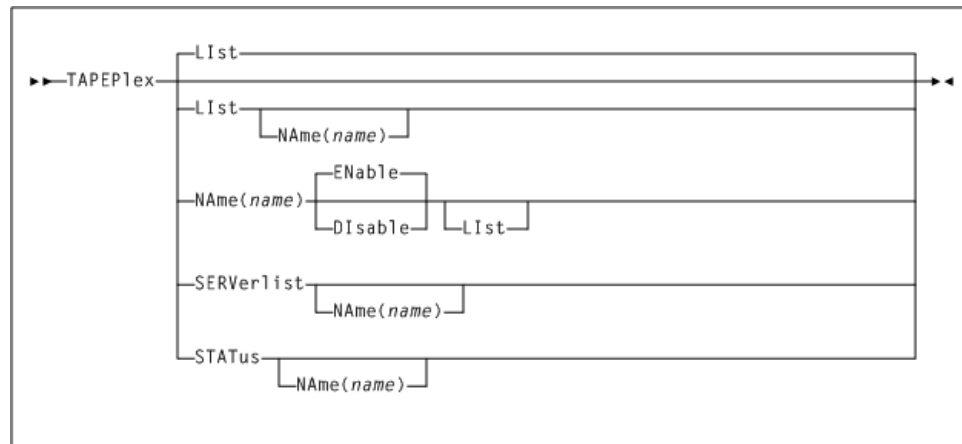
The `TAPEPlex` command defines a TapePlex; a specific StorageTek tape hardware configuration normally represented by a single CDS.

Note the following:

- `TAPEPlex` and `SERVER` commands are required to access HSC TapePlexes.
- The `TAPEPlex` command can also list TapePlexes that VM Client tries to communicate with, and report their status.

Syntax

The following figure shows syntax for the `TAPEPlex` command:

Figure 6–24 TAPEPlex command syntax**Parameters**

As shown in [Figure 6–24](#), the TAPEPlex command includes the following parameters:

List

optionally, lists the specified TapePlex.

Name (name)

optionally, specifies the TapePlex name to be defined or modified.

name is the TapePlex name. This name is reported in any TapePlex error message. The following rules apply:

- The value must be between one and eight characters in length.
- The first character must be either an alpha character or digit.
- The last character must be either an alpha character or digit.
- Any character between the first and last must be either an alpha character, digit or hyphen.

You can specify the following subparameters:

- **ENable** enables the specified TapePlex to be selected for mount requests. This is the default.
- **DIsable** disables the specified TapePlex. The TapePlex is not used for any mount requests.

SERVerlist

optionally, lists defined TapePlexes, their attributes and associated servers. The **SERVerlist** parameter may also be specified with the **NAME** parameter to limit the display to a single TapePlex.

You can specify the following subparameter:

- **NAME** specifies the TapePlex name for which servers are to be listed. *name* is the TapePlex name.

STATus

optionally, lists current status of all TapePlexes, or a single named TapePlex. The TapePlex status indicates whether a TapePlex is active, inactive, or disabled. For an active TapePlex, the status lists the name of the current server. **STATus** does not perform a **RESYNChronize** command.

You can specify the following subparameter:

- **NAME** specifies the TapePlex name for which status information is to be listed. *name* is the TapePlex name.

Example

In the following example, the `TAPEPLEX` command defines a TapePlex named `DENVER` (assuming it is not already defined).

```
TAPEPLEX NAME(DENVER)
```

Note: A `SERVer` command must be specified to define a communications path to TapePlex `DENVER`. See "[SERVer](#)" for an example.

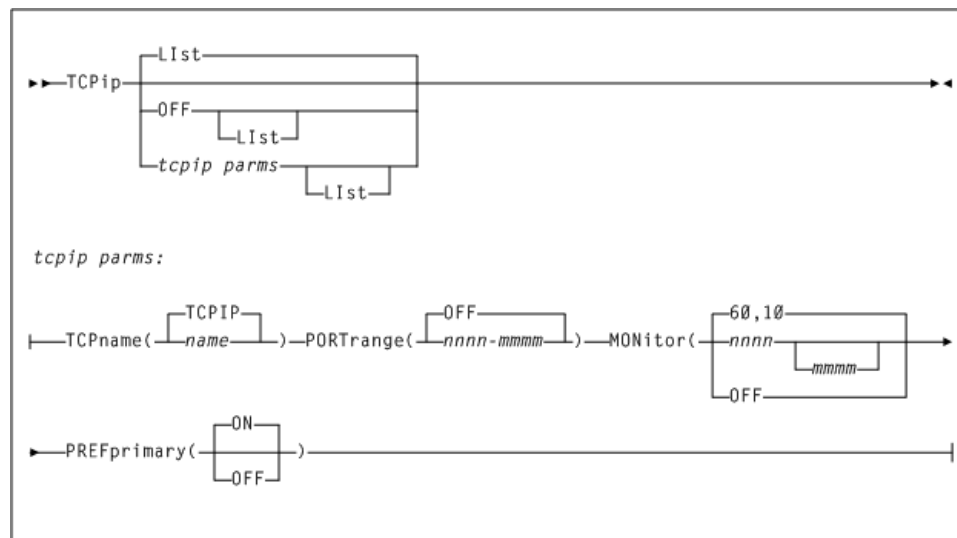
TCPip

The `TCPip` command alters or lists current settings for your TCP/IP communications environment. It enables you to direct TCP/IP requests to a specific TCP/IP stack on a VM host. The `TCPip` command can be issued at any time.

Syntax

The following figure shows syntax for the `TCPip` command:

Figure 6–25 *TCPip command syntax*



Parameters

As shown in [Figure 6–25](#), the `TCPip` command includes the following parameters:

List

optionally, displays the current TCP/IP settings. If a `PORTrange` is specified, `List` also displays the currently bound port numbers and the high-water port numbers indicating the largest number of concurrent communication subtasks executing at one time.

- `List` is the default when no parameters are specified on the `TCPip` command.

- **LIst** may be specified with other parameters. In this case, the **LIst** is generated after the other parameters are processed.

OFF

optionally, specifies that system defaults are used for VM Client TCP/IP communications.

tcpip parms**TCPname (name)**

optionally, specifies the TCP/IP service machine on a VM host.

name is the user ID of the TCP/IP service machine on VM to target TCP/IP communications. The default is **TCPIP**.

PORTrange (nnnn-mmmm) Or (OFF)

optionally, specifies a range of ports to be used by the VM Client to bind() sockets on the client when communicating on remote server paths.

When **PORTrange** is defined, the VM Client binds client sockets to one of the ports within the specified **PORTrange** and will not use client ports outside the **PORTrange**. Therefore, the VM Client can operate behind a firewall that restricts communication to known ports. A unique port is required for each concurrent subtask requiring communication services for a volume lookup, mount, and so on. If a **PORTrange** is not defined, then any ephemeral port is used by the VM Client.

Only one **PORTrange** can be active at a time, but you can dynamically re-define the **PORTrange** even if the new **PORTrange** overlaps with the old **PORTrange**.

- *nnnn-mmmm* is the port number range. Each port number can have a value of 1-65535. The minimum port number range that can be specified is 10 (for example, 6401-6410). The maximum port number range that can be specified is 1000 (for example, 6401-7400).
- **OFF** disables **PORTrange** logic. As a result, any ephemeral port is used. This is the default.

Note the following:

- It is recommended that if you specify a **PORTrange**, you specify a **PORTrange** that does not conflict with TCP/IP well-known ports.
- It is recommended that if you specify a **PORTrange**, you specify a **PORTrange** greater than the anticipated number of concurrent subtasks requesting communication services. For most installations, a **PORTrange** of 40 ports is sufficient. However, if SMC0128 messages are produced with a return code indicating “no free port” then a larger **PORTrange** is required.
- The **TCPIP LIST** command may be used to display the high-water port number, indicating the largest number of concurrent communication subtasks executing at one time.

MONitor (nnnn) and optionally mmmm

optionally, specifies the communication monitor subtask scan interval and communication monitor subtask message interval.

nnnn is the monitor scan interval in seconds. The communication monitor wakes every *nnnn* seconds to perform library communication validation. Specify a value between 10 and 9999. The default is 60.

It is recommended that you preserve the default setting of 60 to enable a monitor scan every minute. A value that is too low can potentially degrade performance when inactive libraries exist. A value that is too high can delay a return to the primary server if `PREFPRIMARY (ON)` is specified.

`mmmm` is optionally, the monitor scan interval in number of scans. Communication error messages are displayed according to this interval. Specify a value between 0 and 9999. The default is 10.

The default `MONITOR (60, 10)` setting specifies a monitor scan interval of 60 seconds, and a monitor message interval of 10 scans. A scan is performed every minute, but error messages are only produced once every 10 scans.

An `mmmm` value of 0 disables all non- irrecoverable or non-disabling error messages issued by the communication monitor subtask. However, errors resulting in the disabling of a server communication path are still issued.

PREFprimary (ON|OFF)

optionally, enables or disables automatic primary server switching. Automatic primary server switching requires the communication monitor subtask to be active. If `MONITOR (OFF)` is specified, primary server switching is disabled.

Example

In the following example, the `TCPIP` command directs TCP/IP requests to a VM service machine named `TCPIP` using any ephemeral port.

```
TCPIP TCPNAME(TCPIP) PORTRANGE(OFF)
```

TRace

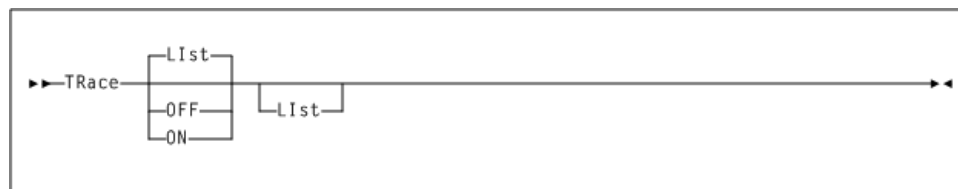
The `TRace` command enables VM Client tracing. The VM Client trace file is written to the `TRACE FILEDEF` file.

Note: This command may impact system performance. Use only as directed by StorageTek Software Support.

Syntax

The following figure shows syntax for the `TRace` command:

Figure 6–26 TRace command syntax



Parameters

As shown in [Figure 6–26](#), the `TRace` command includes the following parameters:

parameter

List

optionally, lists current VM Client trace settings.

- `LIst` is the default when no parameters are specified on the `TRace` command.
- `LIst` may be specified with other parameters. In this case the list is generated after the other parameters are processed.

OFF

optionally, disables VM Client tracing.

ON

optionally, enables VM Client tracing.

Example

In the following example, the `TRace` command enables VM Client tracing.

```
TRACE ON
```

ELS Server Considerations

This chapter describes ELS server considerations.

The VM Client is a thin-client and relies on the resources available within an ELS server TapePlex connected through TCP/IP.

Note: The ELS server must be at release 7.1 or later.

SMC HTTP Server Component

The VM Client specifies a server IP address and port number defined in the ELS SMC configuration. Refer to the publication *Configuring and Managing SMC* for information about how to configure an HTTP server using the SMC HTTP command.

The following is an example of the SMC HTTP command:

```
HTTP START PORT 4242
```

Scratch Subpools

Unlike VM/HSC, the VM Client does not allow management of scratch subpools. Instead, scratch subpools are defined and managed on the ELS server.

A tape management system (TMS), such as VM:Tape, cannot use TMI requests to define scratch subpools. A TMS must use existing scratch subpools defined on the ELS server. In the case of VM:Tape, an attempt to define scratch subpools fails yet VM:Tape initialization continues.

The following is an example of VM:Tape initialization:

```
VMTHSC693I The HSC interface is connecting.
VMTHSC000I Sending to VMCLIENT: QCONFIG with wrong length
VMTHSC000I Sending to VMCLIENT: QCONFIG with right length
VMTHSC000I Sending to VMCLIENT: QDRIVES
VMTHSC999I VMCLIENT completed QDRIVES command successfully
VMTHSC000I Sending to VMCLIENT: DEFSCR 5 4
VMTHSC697E HSC server VMCLIENT ACSRQ=DEFSCR RC=16
VTCS Management Classes
Reason=00001004
VMTHSC721E HSC scratch pool initialization failed.
VMTHSC704I The interface to HSC is ready for use.
```

If VM:Tape attempts to use a scratch subpool that is not defined to the HSC server, the VM Client replies with a nonzero TMI return code and reason code. For example:

```
VMTHSC000I Sending to VMCLIENT: QSCRATCH TEST
VMTHSC697E HSC server VMCLIENT ACSRQ=QSCRATCH
          RC=16 Reason=00008036
VMTHSC698R 'QSCRATCH TEST ' to VMCLIENT failed;
          Enter RETRY, CANCEL, or NOARM;; Reply 1
```

It is the customer's responsibility to keep scratch subpool names and volumes within the subpools synchronized with the VM TMS.

VTCS Management Classes

It is highly recommended that you use the VM Client `POOLmap` command to associate a VTCS management class with a scratch subpool that contains virtual tape volumes (VTVs). When the VM Client receives a TMI request that specifies a scratch subpool name, the VM Client will use the `POOLmap` management class for selecting and mounting VTVs.

See "[POOLmap](#)" for more information about the `POOLmap` command.

VM:Tape Allocation Exit

If the VM:Tape configuration contains multiple tape drives and these tape drives are defined in different ACSs within a TapePlex, scratch mount requests will require the installation of a VM:Tape allocation exit. As an example, the configuration may include two 9840C tape drives, one defined in ACS00 and the other defined in ACS01.

To install a VM:Tape allocation exit, complete the following steps:

1. Copy the `SMCVMTAP` sample file to `VMTAPE`'s 191 minidisk with a file name and file type of `SMCVMTAP EXEC`. You may need to modify the VM Client virtual machine ID in this file.

2. Modify the `VMTAPE` configuration file to enable the allocation exit. For example:

```
EXIT ALLOCATE SMCVMTAP EXEC
```

3. Modify `VMTAPE`'s `PROFILE EXEC` to access the VM Client run disk and execute the `SMCALLOC EXEC` to load the necessary programs into storage. Make sure you include these commands before invoking the `VMTAPE` startup `EXEC`. For example:

```
CP LINK VSMC730A 202 202 RR
ACCESS 202 J
EXEC SMCALLOC
```

This chapter describes system messages issued by VM Client. These messages are identified by the SMC prefix.

Message Descriptions

SMC0000

{{CCCCCCCC}} command string

Level: 0

Explanation: The VM Client received an input command from an operator virtual machine. The virtual machine ID, if available, is listed followed by the command string.

System Action: None.

User Response: None.

SMC0001

VM Client Vn.n.n system initializing

Level: 0

Explanation: The VM Client version n.n.n system initialization process has begun.

System Action: None.

User Response: None.

SMC0002

CCCCCCCCCCCC failed; return code=XXXX, reason code=XXXX

Level: 0

Explanation: Operating system facility CCCCCCCCCC completed with the specified non-zero return code and reason code.

System Action: Depending on the type of error, initiation/termination may try to continue.

User Response: Look for IBM related messages in the SYSLOG and refer to the appropriate IBM documentation for the explanation.

SMC0005

Invalid command CCCCCC [at line nnnn of SMCCMDS|SMCPARMS]

Level: 0

Explanation: An undefined command, *CCCCCCCC*, was encountered by the VM Client.

System Action: Processing continues. The command is ignored.

User Response: Ensure that the syntax in the command data set is correct, or enter the corrected command.

SMC0010

Unable to acquire storage for *CCCCCCCC*; return code=*XXXX*

Level: 0

Explanation: During initialization, the SMC subsystem could not acquire sufficient storage for the specified dynamic control block or module, *CCCCCCCC*.

System Action: The SMC subsystem terminates.

User Response: Ensure that there is sufficient CSA storage available. Refer to the appropriate IBM documentation for the explanation of return code *XXXX*.

SMC0011

Load failed for module *MMMMMMMM*

Level: 0

Explanation: The SMC subsystem could not load the required module *MMMMMMMM*.

System Action: The SMC subsystem terminates.

User Response: Ensure that the SMC startup procedure has access to all SMC distributed load libraries in its steplib concatenation.

SMC0013

TRACE settings:

CCCC...CCCC

Level: 0

Explanation: The TRACE command was specified with the LIST keyword. The SMC0013 multiline message lists the current settings for the VM Client.

System Action: None

User Response: None

SMC0014

Unmatched [quote|or invalid parenthesis] detected; command ignored [at line *nnnn* of *SMCCMDS*|*SMCPARMS*]

Level: 0

Explanation: The VM Client encountered a command containing an unterminated quoted string, or invalid or unmatched parenthesis.

System Action: Processing continues. The command is ignored.

User Response: Ensure that the syntax in the command data set is correct, or enter the corrected command.

SMC0015

Invalid keyword *KKKKKKKK* for the *CCCCCCCC* command[at line *nnnn* of *SMCCMDS*|*SMCPARMS*]

Level: 0

Explanation: The VM Client encountered a command that specified an invalid keyword *KKKKKKKK*.

System Action: Processing continues. The command is ignored.

User Response: Ensure that the syntax in the command data set is correct, or enter the corrected command.

SMC0016

Invalid value *VVVVVVVV* for keyword *KKKKKKKK* of the *CCCCCCCC* command [at line *nnnn* of *SMCCMDS*|*SMCPARMS*]

Level: 0

Explanation: The VM Client encountered a command that specified keyword *KKKKKKKK* with an invalid value *VVVVVVVV*.

System Action: Processing continues. The command is ignored.

User Response: Ensure that the syntax in the command data set is correct, or enter the corrected command.

SMC0017

Keyword *KKKKKKKK* of the *CCCCCCCC* command requires a value [at line *nnnn* of *SMCCMDS*|*SMCPARMS*]

Level: 0

Explanation: The VM Client encountered a command that specified keyword *KKKKKKKK* without an accompanying value (required by most keywords).

System Action: Processing continues. The command is ignored.

User Response: Ensure that the syntax in the command data set is correct, or enter the corrected command.

SMC0018

Keyword *KKKKKKKK* of the *CCCCCCCC* command is not allowed for *EEEEEEEE* [at line *nnnn* of *SMCCMDS*|*SMCPARMS*]

Level: 0

Explanation: The SMC encountered a command that specified keyword *KKKKKKKK*, which is not valid in the current operating environment *EEEEEEEE*. For example, some keywords or keyword=value pairs may be invalid depending upon whether the user is executing JES2 or JES3.

System Action: Processing continues. The command is ignored.

User Response: Ensure that the specified keyword is valid in your environment.

SMC0019

Duplicate keyword *KKKKKKKK* specified for the *CCCCCCCC* command [at line *nnnn* of *SMCCMDS*|*SMCPARMS*]

Level: 0

Explanation: The VM Client encountered a command that specified the same keyword, *KKKKKKKK*, more than once.

System Action: Processing continues. The command is ignored.

User Response: Ensure that the syntax in the command data set is correct, or enter the corrected command.

SMC0020

Keyword *KKKKKKK1* of the *CCCCCCCC* command is mutually exclusive with keyword *KKKKKKK2* [at line *nnnn* of *SMCCMDS*|*SMCPARMS*]

Level: 0

Explanation: The VM Client encountered a command *CCCCCCCC* that specified multiple keywords, two of which (*KKKKKKK1* and *KKKKKKK2*), are mutually exclusive.

System Action: Processing continues. The command is ignored.

User Response: Ensure that the syntax in the command data set is correct, or enter the corrected command.

SMC0022

Invalid format or missing keywords for the *CCCCCCCC* command [at line *nnnn* of *SMCCMDS*|*SMCPARMS*]

Level: 0

Explanation: The VM Client encountered a command *CCCCCCCC* that contained either too many or too few keywords in the command line.

System Action: Processing continues. The command is ignored.

User Response: Ensure that the syntax in the command data set is correct, or enter the corrected command.

SMC0023

CCCCCCCC command successfully processed [at line *nnnn* of *SMCCMDS*|*SMCPARMS*]

Level: 0

Explanation: The *CCCCCCCC* command was successfully validated and processed by the VM Client.

System Action: None

User Response: None

SMC0024

VM Client system initialization complete; RC=*nn*

Level: 0

Explanation: The VM Client system initialization process has completed with the return code indicated and the VM Client system is now ready to receive requests.

System Action: None

User Response: None

SMC0025

No {*CCCCCCCC*|control block} entries to list

Level: 0

Explanation: Command CCCCCCCC was specified with the LIST keyword. However, no entries were found in the VM Client queue for the specified command.

System Action: None

User Response: None

SMC0027

Keyword KKKKKKK1 of the CCCCCCCC command requires keyword KKKKKKK2 [at line nnnn of SMCCMDS|SMCPARMS]

Level: 0

Explanation: The VM Client encountered a command that specified keyword KKKKKKK1, but not the required co-requisite keyword, KKKKKKK2.

System Action: Processing continues. The command is ignored.

User Response: Ensure that the syntax in the command data set is correct, or enter the corrected command.

SMC0029

CCCCCCCC command processing error; [matching entry not found|command line truncated; will be ignored|parameter truncated; command ignored] [at line nnnn of SMCCMDS|SMCPARMS]

Level: 0

Explanation: An error was found processing the CCCCCCCC command.

System Action: Processing continues. The command is ignored.

User Response: Ensure that the syntax in the command data set is correct and enter the corrected command.

SMC0034

VM Client startup parameter PPPPPPPP must have a value

Level: 0

Explanation: During initialization, the VM Client system initialization program encountered a valid execution parameter, but it was not specified as a keyword value pair, and a value is required.

System Action: The VM Client system terminates.

User Response: Correct the VM Client system initialization exec to specify the correct execution parameters.

SMC0035

Error processing VM Client startup parameter PPPPPPPP; CCCCCCCCCC

Level: 0

Explanation: During initialization, the VM Client system initialization program encountered an error in the execution parameter string. The string CCCCCCCCCC indicates the type of error encountered.

System Action: The VM Client system terminates.

User Response: Correct the VM Client system initialization startup exec to specify the correct execution parameter string.

SMC0036

VM Client startup parameter *PPPPPPP* successfully processed

Level: 0

Explanation: During VM Client initialization, the execution parameter *PPPPPPP* was successfully verified and processed.

System Action: None

User Response: None

SMC0037

Invalid VM Client startup parameters; system terminating

Level: 0

Explanation: During initialization, the VM Client system initialization program detected an error processing the execution parameter string.

System Action: The VM Client system terminates.

User Response: Look for VM Client related messages in the service machine log. Associated messages may be (but are not limited to) SMC0033 SMC0034, or SMC0035.

SMC0041

{Command|Comment} beginning at line *nnnn* of {SMCCMDS|SMCPARMS} is unterminated

Level: 0

Explanation: A command or comment beginning at line *nnnn* of an input command file ended with a continuation character (+), but no continuation was found.

System Action: Processing continues. The command containing the unterminated string is ignored.

User Response: Ensure that the syntax in the command data set is correct.

SMC0053

**** VM Client U1099 ABEND AT CCCCCCn ****

Level: 0

Explanation: A VM Client task has abended in module CCCCCC at abend sequence number *n*.

System Action: If the abend occurs in the processing of a TMI request, the request will not be processed.

User Response: Save the associated logs and dumps, and contact StorageTek Software Support.

SMC0056

nn bytes:

AAAAAAAA +0000 | XX. .XX XX. .XX XX. .XX XX. .XX | CC. .CC |

Level: 0

Explanation: A VM Client LIST command was issued. The SMC0056 multiline message lists the *nn* bytes of storage in translated hexadecimal (*XX.XX*) and character (*CC.CC*) format, each line listing the next 16 bytes of storage beginning at hexadecimal address AAAAAAAA.

System Action: None

User Response: None

SMC0057

No {SMCPARMS|SMCCMDS} DDNAME statement found

Level: 0

Explanation: During VM Client initialization, the specified SMCPARMS DD or SMCCMDS DD was not present in the VM Client startup exec.

System Action: Initialization continues.

User Response: None

SMC0058

Error opening {DDNAME {SMCPARMS|SMCCMDS} |DSNAME DDDDDDDD}

Level: 0

Explanation: The VM Client encountered a READ command, but the specified DDNAME or DSNAME could not be opened.

System Action: The READ command is ignored.

User Response: Look for IBM related messages in the log, and refer to the appropriate IBM documentation for more information.

SMC0060

I/O error reading {DDNAME {SMCPARMS|SMCCMDS} |DSNAME DDDDDDDD}

Level: 0

Explanation: The VM Client received an I/O error attempting to read the SMCPARMS or SMCCMDS data set specified in the VM Client startup exec or a data set specified on a READ command.

System Action: The indicated data set is not processed.

User Response: Specify the correct data set name.

SMC0061

Command beginning at line *nnnn* of {SMCCMDS|SMCPARMS} is too long; input ignored

Level: 0

Explanation: The VM Client encountered a multi-line command beginning at line *nnnn* of the specified file. This command exceeds 1024 characters in length.

System Action: Processing continues. The entire multi-line is ignored.

User Response: Ensure that the command data set has the correct syntax.

SMC0062

Command CCCCCCCC [with parameter PPPPPPPP] is not allowed [{from console|at line *nnnn* of SMCCMDS|SMCPARMS}]

Level: 0

Explanation: The VM Client encountered a command or a command parameter that is not supported for the indicated command origin.

System Action: The command is ignored.

User Response: Issue the command from a valid command origin.

SMC0063

MSGDEF settings:

CCCC....CCCC

Level: 0

Explanation: A MSGDEF command has been issued with the LIST keyword. The SMC0063 multi-line message lists the current settings for the VM Client.

System Action: None

User Response: None

SMC0084

MMM DD YYYY HH:MM:SS UUUUUUUU active on hostid VVVVVVVV

Level: 0

Explanation: The Date (MMM DD YYYY), time (HH:MM:SS), service machine user ID (UUUUUUUU), and hostid (VVVVVVVV) are displayed once a day at midnight and during VM Client initialization.

System Action: None

User Response: None

SMC0086

SMC system tasks:

A(PCE) Thread Use C-S Userid Last

AAAAAAA TTTTTT UUUUUU C-S UUUUUUUU TTTTTTTT

Level: 0

Explanation: A VM Client LIST TASKS command was issued. The SMC0086 multiline message lists the SMC PCE address, thread ID, use count, and current status information for all VM Client tasks.

System Action: None

User Response: None

SMC0088

Unable to [acquire/release] resource CCCCCCCC; attempt by VVVVVVVV
XXXXXXXXX1 owned by XXXXXXXXX2

Level: 0

Explanation: A shared VM Client resource could not be acquired or freed successfully. A task servicing virtual machine VVVVVVVV is attempting to acquire or free the resource, but cannot because another task holds the resource.

System Action: The request for virtual machine VVVVVVVV may not be processed correctly.

User Response: Contact StorageTek Software Support.

SMC0093

TCPIP SETTINGS:

CCCC...CCCC
TCPIP TCPNAME=CCCCCCCC

Level: 0

Explanation: A TCPIP LIST command was issued. The SMC0093 multi-line message lists the current settings for the VM Client system.

System Action: None

User Response: None

SMC0105

Keyword KKKKKKKK of the CCCCCCCC command is required

Level: 0

Explanation: The command CCCCCCCC was issued without the required keyword KKKKKKKK.

System Action: The command is not processed.

User Response: Re-issue the command with the required keyword.

SMC0113

SERVER=SSSSSSSS CCCCCCCC

Status={active|never active|inactive|disabled}

Errors=nnnn

Messages=nnnn

Retries=nnnn

Level: 0

Explanation: A SERVER command has been issued with the LIST keyword. The SMC0113 multi-line message lists the server settings and status for each server defined to the VM Client. Refer to the SERVER command for the parameter descriptions.

- STATUS indicates the status of the server.
- Errors indicates the total number of errors on this server.
- Messages indicates the number of logical messages (volume lookup requests, mounts, dismounts) on this server path.
- Retries indicates how many message retries have been attempted.

System Action: None

User Response: None

SMC0116

Cannot find TAPEPLEX PPPPPPPP for SERVER SSSSSSSS [at line nnnn of SMCCMDS|SMCPARMS]

Level: 0

Explanation: A SERVer command was issued with a TapePlex name that was not previously defined.

System Action: The server is not added or updated.

User Response: Specify a TAPEPlex command to define the TapePlex, then specify the SERVer command.

SMC0117

Cannot change TAPEPLEX name for existing SERVER SSSSSSSS [at line nnnn of SMCCMDS|SMCPARMS]

Level: 0

Explanation: A SERVer command was issued with the NAME of an existing server and a TapePlex name, but the TapePlex name of the existing server did not match the TapePlex name in the new command.

System Action: The command is rejected.

User Response: Omit the TapePlex name, change the TapePlex name to match the existing server, or change the server name to add a new server to the specified TapePlex.

SMC0119

SERVER CCCCCCCC now disabled

Level: 0

Explanation: The VM Client detected TCP/IP errors in excess of the FAIL count. See the preceding SMC0128/SMC0129 messages for the reason for the disable.

System Action: None. If there are no additional server paths defined for the associated library, the library hardware is no longer accessible.

User Response: Correct the problem with the TCP/IP network, server, or host operating system, and re-ENABLE the SERVer.

SMC0123

Drive range mismatch between CLIENT(XXXX1-XXXX2) and SERVER (XXXX3-XXXX4)

Level: 0

Explanation: A DRIVemap command was issued. One of the specified CLient range did not match the format of the corresponding SERver range.

System Action: None

User Response: Reissue the command, ensuring that the CLient parameter and the SERver parameter have corresponding formats and number of drives.

SMC0128

TapePlex error:

```
{Fatal comm error detected|
Initialization error number nn or {nn|unlimited}|
Comm error number nn of {nn|unlimited}
Comm error limit exceeded}
USER=UUUUUUU TASK=XXXXXXXXXX {MSG=XXXXXXXXX}
TAPEPLEX=TTTTTTT SERVER=SSSSSSS REQUEST=FFFF
{Client {IP=NNN.NNN.NNN.NNN} socket=NN port={nnnn|ANY}}
{Server IP=NNNN.NNNN.NNNN.NNNN port=nnnn}
```

```
{Bytes out=nnnn in=nnnn}
{Error=EEEE...EEEE}
{Reason=RRRR...RRRR}
{Response from STK HTTP server follows: HHHH...HHHH}
VM Client comm RC=nnnn
```

Level: 0

Explanation: The VM Client encountered an interface or communication error attempting to communicate with a TapePlex. The SMC0128 multiline message first lists the VM user ID, transaction type, and TapePlex name associated with the error followed by the communication error and reason strings.

Examples of the reason strings include:

- Specific TCP/IP function errors (connect, send, recv, and so on.)
- Data errors (incomplete or invalid data response)
- HSC server function errors
- SMC HTTP server errors

Certain communication errors may result in a display of the entire HTTP server response as follows:

```
HTTP 1.0 401 Unauthorized
```

- If the message indicates Comm error limit (*nnn*) exceeded then the SMC0128 message will be followed by an SMC0119 message and the server path will be disabled by the VM Client.
- If the message indicates an Initialization error then the error occurred before any successful communication to the named server path. Such errors are not counted against the cumulative error count on the server path, and will not result in the named server being automatically disabled by the VM Client. Also, Initialization error messages will not be generated for every request, but will only be generated at 5 minute intervals until the path is successfully activated.

System Action: The allocation or mount event may not be processed by the VM Client.

User Response: Use the specified error reason to determine the cause of the problem.

SMC0129

```
{ERROR|WARNING}: No cartridge transport(s) for XXXX1- [XXXX2] for
{UNITATTR|DRIVEMAP} {ADDRESS|CLIENT}
```

Level: 0

Explanation: A UNITATTR or DRIVEMAP command was issued specifying a device XXXX1 or range XXXX1-XXXX2. None of the specified devices is an MVS-defined cartridge transport device.

System Action:

- If the message indicates an ERROR, the command is not processed.
- If the message indicates a WARNING, the VM Client stores the value and may use it to translate addresses for non-MVS-defined RTD devices.

User Response: Review the devices specified in the command and re-issue the command if they are incorrect.

SMC0133

TAPEPLEX=PPPPPPP

CCCC...CCCC

Status={disabled|active|inactive|never active}

Requests=nnnn

[SERVER=SSSSSSS

Status={disabled|active|inactive|never active}]

Level: 0

Explanation: A TAPEPLEX command was issued with the LIST keyword. The SMC0133 multiline message lists parameters and status for each TapePlex defined to the VM Client. Optionally, if the SERVERlist keyword was specified, the server status for all servers associated with this TapePlex is also displayed.

- TapePlex status indicates the status of the TapePlex.
 - disabled indicates that the TapePlex has been disabled by an operator command.
 - active indicates that the last communication to this TapePlex was successful.
 - inactive indicates that a communication path to this TapePlex is no longer active, although one was previously active.
 - never active indicates that a communication path to this TapePlex was never successfully established.
- Requests indicates the total number of requests (configuration, volume lookup, mount, dismount, and swap) that were directed to the specified TapePlex.

If the SERVER keyword was specified, then each server path defined for this TapePlex will also be displayed, along with its status.

System Action: None

User Response: None

SMC0135

Mount/dismount message from TAPEPLEX=PPPPPPP SERVER=SSSSSSS

Level: 0

Explanation: The ECHOMNTERR(ON) option is in effect. A mount or dismount was performed on an HSC TapePlex but did not complete successfully. The SMC0135 message indicates the TapePlex name and server name where the mount or dismount was requested. The SMC0136 message merely echoes the HSC server mount or dismount message on the VM Client.

Note: Messages SMC0135 and SMC0136 are issued for all mount and dismount errors if the message level is 12 or greater regardless of the ECHOMNTERR setting, and for all HSC mount and dismount messages if the message level is 16 or greater.

System Action: None

User Response: Correct the problem indicated in the HSC mount or dismount message.

SMC0136

HSC mount or dismount message

Level: 4

Explanation: The ECHOMNTERR(ON) option is effect. A mount or dismount was performed on an HSC TapePlex but did not complete successfully. The SMC0135 message indicates the TapePlex name and server name where the mount or dismount was requested. The SMC0136 message merely echoes the HSC server mount or dismount message on the VM Client.

System Action: None

User Response: Correct the problem indicated in the HSC mount or dismount message.

SMC0138

XML {input|output} parse error RC=*nnn*; transaction=*TTTTTTTTT*
{TAPEPLEX|STORMNGR}=PPPPPPPP

Level: 0

Explanation: The SMC encountered an XML parse error. Input XML errors are produced when the input XML transaction cannot be parsed.

Output XML errors occur when transaction response data cannot be converted to XML.

System Action: Depending upon the type of error, and server characteristics, the allocation or mount event may not be processed by the SMC.

User Response: Contact StorageTek Software Support.

SMC0160

Invalid range *XXXX1-XXXX2* for keyword ADDRESS of the UNITATTR command

Level: 0

Explanation: A UNITATTR command has been issued specifying a device range *XXXX1-XXXX2*, where *XXXX1* is larger than *XXXX2*.

System Action: The UNITATTR command does not process this device range.

User Response: Reissue the command specifying a valid range.

SMC0161

Restoring all default settings for the CCC...CCC command

Level: 0

Explanation: The CCC...CCC command has been issued with the OFF parameter. All CCC...CCC values have been restored for the VM Client system.

System Action: None

User Response: None

SMC0162

CCC...CCC object successfully {added|updated|deleted}

Level: 0

Explanation: The CCC...CCC command has been successfully processed.

System Action: None

User Response: None

SMC0163

DRIVEMAP settings:

CLIENT=XXXX1{-XXXX2} SERVER=XXXX3{-XXXX4}

Level: 0

Explanation: A DRIVEMAP command has been issued with the LIST keyword. The SMC0163 multiline message lists the currently active DRIVEMAPs. One line is produced for each client/server range.

System Action: None

User Response: None

SMC0164

CLIENT range XXXX1{-XXXX2} not found for the OFF keyword of the DRIVEMAP command

Level: 0

Explanation: The DRIVEMAP command has been issued with the OFF parameter and CLIENT parameter. No matching DRIVEMAP range matching the CLIENT parameter was found.

System Action: None

User Response: None

SMC0165

Keyword {CLIENT/SERVER} range XXXX1{-XXXX2} overlaps with previous DRIVEMAP entry

Level: 0

Explanation: A DRIVEMAP command was issued containing a client or server range that overlaps a range specified on a previously issued DRIVEMAP command.

System Action: The DRIVEMAP command is not processed.

User Response: Issue the DRIVEMAP LIST command to view the list of currently active DRIVEMAP ranges. Correct the DRIVEMAP command to specify a new range. Or, use the DRIVEMAP CLIENT(XXXX1-XXXX2) OFF command to de-activate the existing overlapping range and re-specify the command with unique ranges.

SMC0166

Excessive READ depth at line nn of DSN DDDDDDDD

Level: 0

Explanation: A READ command was issued from a file. However, too many command files are already open, and the read command depth has been exceeded. Read command depth is defined as the number of files that can be open simultaneously due to imbedded Read commands.

System Action: The READ command is not processed.

User Response: Restructure your command files to reduce the READ command depth and ensure that the files referenced do not contain a recursive loop.

SMC0167

CCCCCCC summary:

TAPEPLEX P P P P P P P P is {disabled|inactive|active on
server S S S S S S S S}
{All TAPEPLEX(s) active|
n of n TAPE TAPEPLEX(s) active|
WARNING: All TAPEPLEX(s) inactive|
WARNING: No TAPEPLEX(s) defined|
WARNING: No TAPEPLEX(s) enabled}
Level: 0

Explanation: The CCCCCCCC command was issued and a TapePlex resynchronization was performed. Each TapePlex is represented by a line in the multiline WTO displaying its status.

System Action: None

User Response: None

SMC0172

Specified TAPEPLEX P P P P P P P P not {defined|HSC|active|enabled|valid for UUI}
Level: 0

Explanation: An VM Client command was issued specifying TAPEPLEX P P P P P P P P. However, the command cannot be completed because the TAPEPLEX is either not defined to the VM Client, or is not eligible.

System Action: The command is not processed.

User Response: Either name a valid TAPEPLEX, or correct the TAPEPLEX status and reissue the command.

SMC0173

Response from {TAPEPLEX|STORMNGR} P P P P P P P P:

CCCC...CCCC
Response RC=nn

Level: 0

Explanation: A VM Client Route command was issued that specified TAPEPLEX or STORMNGR P P P P P P P P. The SMC0173 message lists the TAPEPLEX or STORMNGR name, followed by the response, terminated by an SMC0173 message displaying the command return code.

System Action: None

User Response: None

SMC0175

Communication initialized on TAPEPLEX=name SERVER=name

Level: 0

Explanation: The VM Client has successfully communicated with the specified TapePlex for the first time.

System Action: Processing continues.

User Response: None

SMC0176

No active TAPEPLEX(s) for DISPLAY command

Level: 0

Explanation: A VM Client DISPLAY command has been entered. However, the VM Client cannot establish communication with any TapePlex.

System Action: If the DISPLAY (or QUERY) VOLUME was entered, the command terminates as there are no TapePlexes to direct the request. If the DISPLAY (or QUERY) DRIVE command was entered, the command continues although the drive information may not reflect TapePlex ownership.

User Response: None

SMC0177

VM Client {DISPLAY|QUERY} VOLUME

Volser	TapePlex	Location	Media	Rectech	Scr	Volume	Data
-----	-----	-----	-----	-----	-----	-----	-----
VVVVVV	PPPPPPPP	{AA:LL}	MMMMMMMM	RRRRRRRR	SSS	DDDDDDDD	

Level: 0

Explanation: A VM Client DISPLAY (or QUERY) VOLUME command was entered. The SMC0177 message(s) list the volsers that match the request. The displayed Rectech for a volume reflects a combination of the volume's media type, server VOLATTR settings (if any) and volume data such as density. For example, a volume with a displayed Rectech of STK1RC may have a server VOLATTR that specifies a RECTECH of STK1RC or may be known to have been mounted as a scratch on a 9840C drive. The Scratch status will be displayed as "Yes" or "No". The Volume Data for a volume reflects known density and encryption characteristics of the volume as stored in the HSC CDS. Volume Data DEN=1 through DEN=3 means low, high, and highest density respectively.

System Action: None

User Response: None

SMC0178

VM Client {DISPLAY|QUERY} DRIVE

Addr	TapePlex	Location	Model	Serv	VM Client	Status
-----	-----	-----	-----	-----	-----	-----
AAAA	PPPPPPPP	{AA:LL:PP:DD}	MMMMMMMM	SSSS	CCCCCCCC	

or:

Addr	TapePlex	Location	Model	Serv	S	Serial	Number
-----	-----	-----	-----	-----	-----	-----	-----
AAAA	PPPPPPPP	{AA:LL:PP:DD}	MMMMMMMM	SSSS	Z	NNNNNNNNNNNN	

Level: 0

Explanation: A VM Client DISPLAY (or QUERY) DRIVE command was entered. The SMC0178 message(s) list the drives that match the request. The address AAAA reflects the drive address as it is known to CP. VM Client will attempt to match the drive's VM Equivalency ID (EQID) to the serial number returned in the XAPI configuration request. If that is unsuccessful, DRIVEMAP entries are used to map server address (SSSS) to the client address (AAAA).

The DISPLAY DRIVE IDENTITY option replaces the "VM Client Status" field with the "Serial Number" field where:

- Z identifies the source or status of the serial number.
 - NNNNNNNNNNNN - Drive serial number or blank if unavailable
 - M - EQID serial number Matched to XAPI configuration
- NNNNNNNNNNNN - Drive serial number or blank if unavailable

System Action: None

User Response: None

SMC0179

{TAPEPLEX|ESOTERIC} VVVVVVVV not defined for CCCCCCCC

Level: 0

Explanation: The CCCCCCCC command was entered specifying TAPEPLEX or ESOTERIC VVVVVVVV. However VVVVVVVV is not defined to SMC or MVS.

System Action: None

User Response: Correct the specified command and re-enter.

SMC0189

CCCCCCCC entry EEEEEEE not found for {list|update|delete}

Level: 0

Explanation: A CCCCCCCC command was entered specifying that entry EEEEEEE be either listed, deleted, or updated. However, no entry matching EEEEEEE was found.

System Action: None

User Response: Issue the CCCCCCCC command with the LIST option to list all CCCCCCCC entries. Then re-issue the command specifying the correct entry name.

SMC0190

CCCCCCCC 00000000 set to {ON|OFF|XXXXXXXX}

Level: 0

Explanation: A CCCCCCCC command was entered specifying that option 00000000 be set to ON, OFF, or the specified value XXXXXXXX. If multiple options were specified on a single CCCCCCCC command, then multiple SMC0190 messages are issued, one for each specified option.

System Action: None

User Response: None

Note: SMC0190 messages are displayed only if MSGDef VERBose(ON) is specified.

SMC0191

CCCCCCCC 00000000 set to {ON|OFF|XXXXXXXX} for entry EEEEEEE

Level: 0

Explanation: A CCCCCCCC command was entered specifying that option 00000000 be set to ON, OFF, or the specified value XXXXXXXX for the CCCCCCCC entry EEEEEEEE. If multiple options were specified on a single CCCCCCCC command, then multiple SMC0191 messages will be issued, one for each specified option.

System Action: None

User Response: None

Note: SMC0191 messages are displayed only if MSGDef VERBose(ON) is specified.

SMC0195

READ processing started for {SMCPARMS|SMCCMDS|data set name}

Level: 0

Explanation: The VM Client has begun processing commands in the named file.

System Action: None

User Response: None

SMC0196

READ processing complete; RC=nn from {SMCPARMS|SMCCMDS|data set name}

Level: 0

Explanation: The VM Client has completed processing commands in the named file. The highest return code for any command is nn.

System Action: None

User Response: None

Note: SMC0196 messages are displayed only if VM Client MSGDef VERBose(ON) specified.

SMC0203

COMMTTEST:

USER=UUUUUUUU TASK=XXXXXXXXXXXXXXXXX {MSG=XXXXXXXXX}
 TAPEPLEX=LLLLLLLL SERVER=SSSSSSSS REQUEST=FFFF
 Client {IP=NNN.NNN.NNN.NNN} socket=NN port={nnnn|ANY}
 Server IP=NNNN.NNNN.NNNN.NNNN port=nnnn
 Bytes out=nnnn in=nnnn
 Error=EEEE...EEEE
 Reason=RRRR...RRRR
 {Response from STK HTTP server follows: HHHH...HHHH}
 Current LIBPATH status=
 {active|inactive|never active|disabled}
 VM Client comm RC=nnnn elapsed time=nn.nn

Level: 0

Explanation: A COMMtest command was entered. The SMC0203 message is displayed for each communication path attempted.

System Action: None

User Response: None

SMC0204

No eligible COMMPATH(s) found

Level: 0

Explanation: A COMMtest command was entered, but the specified TAPEPlex, SERVer, and status parameters resulted in no eligible communication paths selected for the test.

System Action: None

User Response: Correct and reissue the COMMtest command.

SMC0205

Disabling bind to PORTRANGE nnnn-nnnn; any ephemeral port will be used

Level: 0

Explanation: A TCPip PORTrange (OFF) command was entered. Sockets will no longer be bound to the fixed port range of nnnn-nnnn, but any ephemeral port will be used.

System Action: None

User Response: None

SMC0206

No PORTRANGE currently defined

Level: 0

Explanation: A TCPip PORTrange (OFF) command was entered but there is currently no active PORTrange specified to disable.

System Action: None

User Response: None

SMC0207

Specified SERVER SSSSSSSS not {found|defined for TAPEPLEX=TTTTTTTT}

Level: 0

Explanation: A COMMtest command was entered specifying a specific TapePlex and server. However, the server is either not defined to the VM Client, or is not defined for the specified TapePlex.

System Action: None

User Response: Correct and reissue the COMMtest command.

SMC0226

Path switch from server=SSSSSSSS to PPPPPPPP for TAPEPLEX=TTTTTTTT

Level: 0

Explanation: The VM Client automatically switched the communication path from the secondary server SSSSSSSS to the primary server PPPPPPPP for TAPEPLEX TTTTTTTT.

System Action: Processing continues.

User Response: None

SMC0227

Keyword KKKKKKKK of the CCCCCCCC command ignored; RRRRRRRR

Level: 0

Explanation: The CCCCCCCC command specified a keyword that is no longer acceptable. Keyword KKKKKKKK may be obsolete in the current version of the product, or it may be unacceptable in the current processing environment.

System Action: Keyword KKKKKKKK and any associated value are discarded, but the remainder of the command is still processed.

User Response: If the keyword is obsolete in the current release, delete the keyword from the command as it may be flagged in error in subsequent releases, invalidating the entire command.

SMC0228

Copyright nnnn, nnnn, Oracle and/or its affiliates. All rights reserved.

Level: 0

Explanation: The VM Client system is initializing.

System Action: Processing continues.

User Response: None

SMC0232

Warning: No TAPEPLEX command processed

Level: 0

Explanation: The VM Client system has completed initialization, but no TAPEPLEX commands were found in either the SMCPARMS or SMCCMDS data set.

System Action: Processing continues.

User Response: Enter TAPEPLEX and SERVER commands.

SMC0236

CCC...CCC command RC=XX exceeds MAXRC=NN at startup

Level: 0

Explanation: The VM Client was started with the MAXRC startup parameter and during VM Client initialization the CCC...CCC command returned a completion code that exceeded the MAXRC specification.

System Action: Processing continues for the remainder of the commands specified in the SMCPARMS or SMCCMDS data set. However, VM Client system initialization will be terminated with the SMC0237 message at the completion of the SMCPARMS or SMCCMDS processing.

User Response: Correct the specified CCC...CCC command and restart the VM Client.

Note: Multiple SMC0236 messages may be produced at startup as all VM Client commands in the SMCPARMS and SMCCMDS data sets are processed at startup, regardless of prior SMC0236 messages.

SMC0237

VM Client terminating due to MAXRC=nn exceeded at startup

Level: 0

Explanation: The VM Client was started with the MAXRC startup parameter, and during VM Client initialization an SMC0236 message was issued indicating that an VM Client command in the SMCPARMS or SMCCMDS data set returned a completion code that exceeded the MAXRC specification.

System Action: The VM Client terminates.

User Response: Review the VM Client log for the SMC0236 message(s) indicating the commands in error, correct the indicated commands, and restart the VM Client.

SMC0242

Cannot add STORMNGR CCC...CCC before TAPEPLEX(es)

Level: 0

Explanation: STORMNGR commands must be entered after TAPEPLEX commands.

System Action: Processing continues.

User Response: Enter TAPEPLEX commands before STORMNGR commands.

SMC0243

CCCCCCCC command specifies {TAPEPLEX|STORMNGR} NNNNNNNN; but NNNNNNNN is a {STORMNGR|TAPEPLEX} [at line nnnn of {SMCCMDS|SMCPARMS}]

Level: 0

Explanation: The CCCCCCCC command was entered and specified the named TAPEPLEX or StorageTek Storage Manager. However, NNNNNNNN is not the type of entity described.

System Action: The command is not processed.

User Response: Change the entity type from TAPEPLEX to STORMNGR or vice versa, and reissue the command.

SMC0244

METADATA command not supported for {non-UII origin|non-XML responses|command CCCC}

Level: 0

Explanation: A METADATA command was processed, but it is invalid for one of the following reasons:

- non-UII origin
indicates that the command was received from an operators console, or from the SMCPARMS or SMCCMDS data sets. The METADATA command is only allowed from the UII interface; either from the SMCUII or SMCUSIM utilities, or from the UII programmatic interface.
- non-XML responses
indicates that the METADATA command originated from the UII interface, but that XML responses were not requested. METADATA is only valid as a XML response.
- command CCCC
indicates that the specified command CCCC does not produce XML output, so metadata is not available.

System Action: None

User Response: Correct the METADATA command.

SMC0245

Code *nnnn* (X'*xxxx*') : *ssssssssss*

Level: 0

Explanation: A Display RC command was processed that specified reason code *nnnn*, or hex reason code *xxxx*. The corresponding reason is displayed. If the DETAIL option was specified, the reason code explanation is also displayed.

System Action: None

User Response: None

SMC0260

TAPEPLEX|STORMNGR CCCCCCCC commpath P P P P P P P P inactive; RC=RRRR, EEEEEEEEEEEE

Level: 0

Explanation: SMC is unable to communicate to the TAPEPLEX or STORMNGR using the specified commpath P P P P P P P P, where P P P P P P P P is the server name or (local). The value RRRR is the decimal return code, with a translated explanation EEEEEEEEEEEE.

System Action: The message is issued for each defined local commpath or SERVER and is non-scrollable as long as SMC is unable to communicate with the TAPEPLEX.

User Response: Correct the reported error for at least one communication path.

SMC0261

TAPEPLEX|STORMNGR CCCCCCCC inactive; no available communication paths

Level: 0

Explanation: The TAPEPLEX or STORMNGR CCCCCCCC has no defined communication paths, or all paths have a disabled status.

System Action: No communication is attempted to the TAPEPLEX or STORMNGR.

User Response: Add a communication path, or enable an existing local path or SERVER.

SMC0268

Unrecognized XML tag=TTTTTTTT for command=CCCCCCCC

Level: 0

Explanation: An input request in XML format contained a tag that was not recognized as valid for the command.

This message can be caused when the current software level does not support a tag that was valid in an earlier level, or has not been upgraded to support a new tag.

System Action: The parameter is ignored.

User Response: Verify that the command is specified correctly.

SMC0269

Value=VVVVVVVV is invalid type for keyword or tag=KKKKKKKK in command=CCCCCCCC [at line *nnnn* of SMCCMDS|SMCPARMS]

Level: 0

Explanation: An input command contained a value for a keyword or XML tag that was not of the required type, for example, not a valid number or a list for a parameter that does not allow a list.

System Action: Processing continues. The command is ignored.

User Response: Correct the error and re-specify the request.

SMC0270

Keyword or tag=KKKKKKKK may not have a value in command CCCCCCCC [at line nnnn of SMCCMDS|SMCPARMS]

Level: 0

Explanation: An input command contained a value for a keyword or XML tag that does not allow a value.

System Action: Processing continues. The command is ignored.

User Response: Correct the error and re-specify the request.

SMC0271

Length of value=VVVVVVV is invalid for keyword or tag=KKKKKK in command CCCCCCCC [at line nnnn of SMCCMDS|SMCPARMS]

Level: 0

Explanation: An input command contained a value for a keyword or XML tag that was shorter or longer than the required length.

System Action: Processing continues. The command is ignored.

User Response: Correct the error and re-specify the request.

SMC0272

Error parsing XML values for XML tag=TTTTTTTT in command=CCCCCCCC; RC=nnn

Level: 0

Explanation: An XML command contained a value or parse error related to the listed tag. The parse return code is included in the message for diagnostics.

System Action: Processing continues. The command is ignored.

User Response: Correct the error and re-specify the request.

SMC0300

Message|Command nnnnn Help Text:

Level: 0

Explanation: A Help command has been issued. The help text for the message or command is listed.

System Action: None

User Response: None

SMC0301

HELP for XXXXXX not found

Level: 0

Explanation: A Help command has been issued. The subject `XXXXXX` is not found.

System Action: None

User Response: Re-enter the Help command with a valid subject.

SMC0302

`XXXXXX` is an invalid range

Level: 0

Explanation: A Help command has been issued. The subject `XXXXXX` is an invalid range.

System Action: None

User Response: Re-enter the Help command with a valid subject.

SMC0805

VM Client failed setting ANCHOR: `rc=nnn`

Level: 0

Explanation: A non-zero return code was received executing the CMS ANCHOR SET macro.

System Action: VM Client terminates.

User Response: Contact StorageTek Software Support.

SMC0806

TCP/IP server available: `id=YYYYYY`

Level: 0

Explanation: VM Client successfully connected with TCP/IP id, `YYYYYY`.

System Action: None

User Response: None

SMC0807

TCP/IP server is unavailable; `id=YYYYYY` `errno=NNN` `errmsg`

Level: 0

Explanation: VM Client TCP/IP functions returned an error trying to establish a socket connection with `YYYYYY`.

System Action: None

User Response: Correct the TCP/IP id specified in the SMCPARMS file and re-start VM Client. If the TCP/IP id is correct, contact StorageTek Software Support.

SMC0810

Dynamic Allocation NOT supported

Level: 0

Explanation: The READ command attempted to dynamically allocate a file to read.

System Action: The command is not processed.

User Response: Re-enter the READ command, specifying the SMCCMDS file from the startup FILEDEF.

SMC0811

cp command

Level: 0

Explanation: The output from the CP command is displayed.

System Action: None

User Response: None

SMC0812

LOG command requires CONSOLE or DISK

Level: 0

Explanation: The LOG command requires that CONSOLE or DISK must be entered.

System Action: The command is not processed.

User Response: Reenter the LOG command with the proper options.

SMC0813

*mmmm+nnnn - aaaaa - PSW data ppppEvent eeee - Data dddd - Thread nnnnData at PSW
addr - xxxx<>xxxx*

Level: 0

Explanation: An abend has been detected.

- *mmmm+nnn* indicates the abending module and displacement
- *aaaa* indicates the abend code
- *pppp* indicates the PSW at abend
- *eeee* indicates the address of the VMERROR data return by the error event
- *dddd* indicates the address of the Data Area returned by the error event
- *nnnn* indicates the abending thread id
- *xxxx<>xxxx* indicates the data at the PSW address
- *<>* marks the PSW address

The registers at abend follow

System Action: The command or TMI request is terminated.

User Response: Contact StorageTek Software Support.

SMC0814

Dump sent to *nnnn* *ddd* dumps remain

Level: 0

Explanation: The system produced a storage dump in response to a DUMP command or because of an abending program.

System Action: The dump file is transferred to the user ID *nnnn*.

User Response: If the number of dumps allowed (*ddd*) approaches 0, issue the DUMPOPTS RESET command to reset the number of dumps allowed and/or change the maximum dump limit. If this message is not in response to a DUMP command, contact StorageTek Software Support.

SMC0815

Dump not taken due to dump Max Count - *nnnn*

Level: 0

Explanation: The maximum number of dumps allowed by the DUMPOPTS command was produced.

System Action: This dump request is ignored. Until a DUMPOPTS RESET command is received, no dump requests are honored.

User Response: Issue the DUMPOPTS RESET command to reset the number of dumps taken and/or change the maximum dump limit.

SMC0816

DUMPOPTS settings:

DUMPS TAKEN=*tttt*
DUMPS MAX COUNT=*mmmm*
TO=*uuuuuuuuu*

Level: 0

Explanation: A DUMPOpts command has been issued with the LIST keyword. The DUMPOPTS settings are listed.

System Action: None

User Response: None

SMC0817

LOG settings:

CONSOLE=ON|OFF
CLASS=*c*
TO=*uuuuuuuuu*
DISK=ON|OFF
FM=*a*

Level: 0

Explanation: A LOG command has been issued with the LIST keyword. The LOG settings are listed.

System Action: None

User Response: None

SMC0818

OPERATOR settings:

ID=*uuuuuuuuu*

Level: 0

Explanation: An OPERator command has been issued with the LIST keyword. The OPERATOR settings are listed.

System Action: None

User Response: None

SMC0819

Disk FM m is READONLY

Level: 0

Explanation: The disk specified in the FM parameter is READONLY. The disk must be writable.

System Action: The command is not processed.

User Response: Re-enter the command with proper parameters.

SMC0820

Disk FM m is not defined

Level: 0

Explanation: The disk specified in the FM parameter is not defined.

System Action: The command is not processed.

User Response: Re-enter the command with proper parameters.

SMC0821

UUUUUU is not a defined VM userid

Level: 0

Explanation: The value entered is not a defined VM user ID.

System Action: The command is not processed.

User Response: Re-enter the command with proper parameters.

SMC0822

XXXXXXXX has initiated VM Client termination

Level: 0

Explanation: An EXIT command has been received from the XXXXXXXX source.

System Action: VM Client is starting termination processing.

User Response: None

SMC0823

UUUUUUUU is not authorized for VM Client ZZZ requests

Level: 0

Explanation: A VM Client ZZZ request received from VM user ID UUUUUUUU has been denied. The user ID is not authorized to execute ZZZ requests.

System Action: VM Client ignores the request and continues processing.

User Response: Use the AUTHORIZE command to allow VM user IDs to execute VM Client TMI and command requests. Another option is to use the VM Client verification customer exit, SMCXIT01.

SMC0824

MOUNT|DISMOUNT command failed; RC=NNNN - Reason=MMMM

Level: 0

Explanation: The MOUNT or DISMOUNT command failed with a Return Code of *NNNN*. The Reason Code (*MMMM*) may be the HSC message number that describes the failure reason.

System Action: The command failed.

User Response: Correct the problem indicated in the HSC mount or dismount message.

SMC0825

FORCE parameter invalid for virtual drive *DDDD*

Level: 0

Explanation: The DISMOUNT FORCE parameter is not supported for virtual devices.

System Action: The dismount is not processed.

User Response: Re-issue the DISMOUNT command without the FORCE parameter. If necessary, issue the CMS TAPE RUN or CP DETACH command to unload the drive.

SMC0826

Authorized users: *UUUUUUUU* Requests: *req1 req2 ...*

Level: 0

Explanation: An AUTHorize command has been issued with the LIST keyword. The authorized users and the types of requests they are authorized for are displayed.

System Action: None

User Response: None

SMC0827

POOLMAP SCRATCH MANAGEMENT

SUBPOOL	CLASS
PPPPPPPPPPPP	MMMMMMMM

Level: 0

Explanation: A POOLmap command has been issued with the LIST keyword. The scratch subpool names, *PPPPPPPPPPPP*, are displayed with their corresponding management class names, *MMMMMMMM*.

System Action: None

User Response: None

SMC0828

POOLMAP validation failed; *reason*

Level: 0

Explanation: The POOLmap command received an error validating the management class and subpool name with the HSC server.

The possible *reasons* are:

- invalid management class
- invalid subpool name
- HSC server unavailable

System Action: None

User Response: Correct the invalid parameter and re-issue the command.

SMC0829

Mount of volume VVVVVV complete on drive DDDD

Level: 0

Explanation: Tape volume, VVVVVV, has been successfully mounted on tape drive DDDD.

System Action: None

User Response: None

SMC0830

Waiting for a TapePlex SSSS server to become active

Level: 12, 16, 20, 24, 28

Explanation: This message is displayed when the WAIT option is specified for the RESYNC command and no TapePlex servers are available.

System Action: None

User Response: None

SMC9999

MMMMMMMM Variable text

Level: 0

Explanation: SMC9999 messages are intended for StorageTek Software Support problem determination and resolution. MMMMMMMM is the name of the issuing module.

System Action: None

User Response: None. A message level (LVL) of 12 or higher should generally be specified only when directed by StorageTek Software Support.

VM Client Tape Management Interface

This chapter describes the VM Client Tape Management Interface (VMTMI).

A TapePlex refers to the complex of resources managed by StorageTek software, including library resources for real cartridges and VSM resources for virtual volumes.

A TapePlex is defined as the hardware managed by a single HSC CDS. VM Client provides access to a single TapePlex. VM Client uses TCP/IP to route transactions to an HSC server executing on z/OS. Multiple instances of HSC on multiple hosts can be defined as servers to provide redundancy. Transactions between VM Client and the HSC server use an XML-based API called the XAPI. The VM Client software converts Tape Management Interface transactions into XAPI format for interpretation by the server, and converts the output of these transactions into TMI format for the response.

Because the Tape Management Interface now interacts with a client component, which in turn interfaces with HSC (and VTCS) on z/OS, some TMI commands that were supported in previous releases are no longer supported. In some cases, command functionality cannot be supported in a client/server environment, while other commands may be supported in future releases.

The following TMI commands are not supported for VM Client 7.3:

- DEFSCR and DEFPOOL

These commands are superseded by the HSC POOLPARM/VOLPARM feature introduced in ELS 7.0. Refer to the publication *Configuring HSC and VTCS* for more information about this feature.

- QEJECT
- QREQUEST
- SETOPER

VM Client does not support operator responses; therefore, all commands are processed as though the SETOPER command was issued. Conditions that previously would have generated WTOR messages are returned as error messages.

- STOP

In addition, the VM Client does not support the use of TAPEREQ lookup keys, including job name, step name, program name, and data set name, to select media and recording technique values.

A general description of the interfaces between the tape management system (TMS) and VM Client includes the following topics:

- TMS Responsibilities

This section describes the services a TMS provides.

- TMS Decision Points

This section describes where TapePlex interaction assists TMS services for TapePlex-managed resources.

- TapePlex Information Returned to the TMS

This section describes information returned because of TapePlex interaction.

- Inter-user Communications Vehicle (IUCV) Considerations

This section describes the parameters used with the IUCV macro.

- TMS and VM Client Interaction

This section describes various scenarios involving TMS to VM Client interaction.

The sections that follow describe these topics in detail.

TMS Responsibilities

The Tape Management System has three major functions:

- User interface
- Tape resource allocation
- Operator interface

User Interface

Normally, an end user requests TapePlex facilities indirectly by requesting services from a Tape Management System (TMS). Such requests are routed by the TMS to the VM Client, and then to the TapePlex server. Direct interaction between an end user and the VM Client only occurs if the end user issues VM Client commands directly using the VM Special Message (MSG) facility, or invokes the VMTMI directly. See the distributed VMTMI SAMPLE for an example.

Tape Resource Allocation

The resources under TMS control include:

- Transports

The TMS normally has ownership of transports for Automatic Volume Recognition (AVR) and also assigns transports to users requesting tape services. The TMS determines the availability of tape transports for allocation requests. The TMS also knows the media type and density any transport supports. VM Client assists the TMS in selecting TapePlex-controlled transports, when necessary.

- Data sets

The TMS maps data sets to tape volumes and may map external labels to internal labels. The TapePlex contains no such information.

- Scratch volumes

The TMS is the final authority concerning the scratch status of volumes. This status also includes scratch subpool membership. To automate mount processing for a TMS generating "nonspecific" mounts (requests for scratch volumes that do not specify VOLSERS), the TapePlex also retains its own scratch status

information. The TMS scratch status list is not considered a list of all available scratches, but rather as a subset of the total number of available scratch volumes.

- Specific volumes

The TMS controls which users have access to any specific volume. The TapePlex handles volumes it controls at the request of an authorized operator or the TMS.

Operator Interface

Tape mounting, dismounting, and scratch pool selection is handled through message traffic between the TMS and the operator. The VM Client uses information supplied in messages to the operator to direct mounts, dismounts, and so on.

TMS Decision Points

The TapePlex server can influence TMS decisions when TapePlex services are available to the TMS through the VM Client. The TapePlex server influences the TMS decisions at the following points:

TMS Initialization

When the TMS is initialized, have the TMS establish an IUCV path to the VM Client machine to determine if both the VM Client and the TapePlex server are operational and communicating. If the VM Client machine is not operational at TMS startup, establish an IUCV path as soon as possible after VM Client and TapePlex initialization. It is possible to establish and break connection for each transaction, but this causes unnecessary processing. To use IUCV efficiently, a path must be established and maintained throughout the TMS communications session.

Drive Allocation

At allocation time, the VM Client can provide information about media and location of specific volumes and scratch counts. The TMS can use the result of queries to select compatible optimum drives for an allocation request.

Scratch Allocation

If the TMS requests, the VM Client can provide the VOLSER of a scratch volume before a MOUNT request. This selection can be rejected or used on the subsequent mount. If this information is not needed, a nonspecific MOUNT requests causes a scratch volume to be selected.

Volume Movement

When a mount, dismount, or other movement of a volume is required, the TMS decides whether the TapePlex performs the action or if a manual operation is required. The VM Client returns status information for volume movement requests.

Returning a Volume to Scratch Status

It is necessary to keep the TMS and TapePlex scratch status synchronized. The earliest time is at dismount, the latest when a TMS scratch pull list is generated. TMI requests and VM Client commands are available to coordinate this activity.

TapePlex Information Returned to the TMS

TapePlex information returned includes the following:

Configuration Information

Configuration information includes the following:

- Maximum number of transports in the largest ACS
- Number of transports under library control
- Number of ACSs
- Number of LSMs
- Response area sizes

Volume Status

Volume status information includes the following:

- Volume in a cell
- Volume in a drive
- Volume not in library
- Volume inaccessible
- Volume location uncertain (errant)

Volume Location

Volume location information includes the following:

- ACSid
- LSMid
- Panel location
- Row location
- Column location

Eligible Drives

Drives eligible from a TapePlex perspective are those in the same ACS or VTSS as the volume, and with a recording technique compatible with the volume. Not taken into account is the actual drive availability (attachability, online status).

Movement Status and Error Codes

Volume movement requests return a code indicating the success of the operation. If an error occurs, an additional code corresponding to the message issued to the operator is also returned.

LSM and ACS Status

LSM status is either online or offline. Online indicates that automated mounting can take place. Offline indicates only manual mounting is possible.

ACS status is connected or disconnected. Disconnected indicates that the ACS is not accessible from this host and any activity must be handled from another host.

- **VOLSERs for Scratch Management**

If scratch selection is requested, a VOLSER marked as scratch in the TapePlex control data set is returned. This selection causes the volume to be marked as nonscratch.

- **Library Notation for Virtual Drives and Volumes**

Using VM Client enables the TMS to access virtual drives and virtual volumes. The Tape Management Interface maps each VTSS to an ACS ID and a set of LSM IDs so that the TMS can handle virtual drives and volumes like real volumes. VM Client provides the VTSS to ACS mapping and ensures that ACS IDs that represent VTSSs do not overlap with real ACSs. Volume movement operations, such as EJECT, MOVE, and ENTER, are not allowed for virtual volumes.

Inter-user Communications Vehicle (IUCV) Considerations

The inter-user communication vehicle (IUCV) is an IBM-supplied communications interface.

Note: VMTMI SAMPLE is a sample program illustrating the use of the VM Client Tape Management Interface. It can be found on the MAINTSTK user ID.

To use IUCV to issue requests, follow these steps:

1. Establish a connection to the VM Client service machine using the IUCV CONNECT function.
 - Only authorized virtual machines may issue commands to the VM Client service machine. To obtain permission, the virtual machine issuing the TMI or VM Client command must be given privileges by a VM Client AUTHORIZE command issued to the VM Client service machine either in the SMCPARMS or SMCCMDS file at startup, or from a previously authorized virtual machine.
 - In addition, the virtual machine must be authorized to CP through an IUCV control statement in its CP directory entry. This is typically done by a systems programmer or administrator. Make sure that the OPTION MAXCONN specifies enough paths for your usage.
 - To establish this IUCV connection (path), the IUCV macro is issued with the following parameters:

```
IUCV CONNECT,
    PRMLIST=addr,      * address of IUCV parm list
    USERID=addr,       * address of CL8 'userid'
    USERDATA=addr,     * address of CL16 'ddname'
    PRMDATA=NO         * no parm data in IPARML
```

where:

userid indicates the name of the VM Client service machine.

ddname indicates the name of the VM Client IUCV interface that is requested for the connection. It is a 16 byte area as follows:

```
DC CL8 'SLSTLMS'      ddname
```

DC CL8' ' reserved

On execution of the function, check the PSW condition code. If the condition code is 0, save the path ID from the IPARML area passed to the macro. The program must wait for the VM Client service machine to IUCV ACCEPT the pending connection before sending any messages. If no "connection complete" or "path severed" is returned, either the VM Client is not active, is not fully initialized, or the IUCV CONNECT requester specified invalid parameters.

2. Send the message to the VM Client service machine using the IUCV SEND function. Specify the following parameters to the IUCV macro:

```
IUCV SEND,  
    PATHID=adpid,  
    TYPE=2WAY,  
    BUFLen=buflen,  
    RBUF=reply,  
    RLEN=reply length
```

where:

- *adpid* is the address of a data area containing the IUCV path ID.
 - TYPE=2WAY specifies that an IUCV reply is expected.
 - *buflen* is the length of "buffer".
 - *reply* is the address of the buffer containing the reply.
 - *reply length* is the length of the reply buffer.
3. When you are finished using a connection to the VM Client service machine, release the path using the IUCV SEVER function.

The following IUCV parameters are not supported for this interface:

TYPE=1WAY

IUCV REPLY must be issued by the VM Client.

TRGCLS= TRGCLS

is ignored by the VM Client

DATA=PRMSG CP

does not allow the SEND to occur.

PRMSG=address

CP does not allow the SEND to occur.

PRTY=YES

CP uses this to alter queuing to the VM Client service machine. The VM Client service machine does not give the message any special handling.

All other IUCV parameters may be used as desired.

Additional Considerations

Three fields in the IUCV parameter list (IPARML) deserve special mention:

USERID=

This parameter specifies the name of the service machine running the VM Client. Oracle recommends that your user ID be alterable, rather than hard-coded, to facilitate changes.

USERDTA=

This parameter specifies the name of the process in the service machine receiving TMS communication (ACSINT). This must be SLSTLMS.

UWORD=

This parameter specifies a word which will appear in R0 at interrupt time. It is useful for establishing addressability to a common data area. This contains an area listing pending requests, path status, and configuration values that are referenced in different routines.

Refer to the IBM publication *VM/SP System Facilities for Programming* or *VM/XA CP Programming Services* for additional information about the IUCV interface and the parameters listed above.

TMS and VM Client Interaction

VM Client requires several points of interface with a tape management system (TMS):

- TMS to VM Client initial connection
- Drive Allocation
- Operator message processing

The TMS provides a front end between VM Client and the user, maintaining allocation, data set, and scratch pool services. VM Client provides the TMS with mount/dismount handling and scratch volume selection, and influences the allocation of those volumes and drives under TapePlex control.

An invocation macro (ACSRQ) prepares a call to the Inter-User Communications Vehicle (IUCV) to communicate between the TMS and VM Client. The IUCV call itself is made by the TMS.

The following tape management system facilities support the communication:

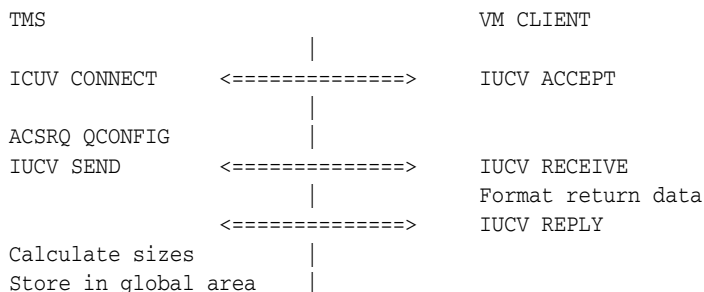
- An interface at allocation time to supply device type and scratch information
- An interface at message time to handle the message normally displayed to the operator
- A list of transport drive addresses and corresponding media and location information
- An IUCV interrupt handler

TMS to VM Client Initial Connection

At initial connection time it is useful to determine the size of the returned data areas for a few of the longer responses. These vary depending on the TapePlex configuration. These areas are then allocated before their required use.

Initial Connection Dialog

At connection time, a QCONFIG request should be issued to determine the size of the reply data areas that are needed for other requests. For example:

Example 9-1 Initial Connection Dialog

Drive Allocation

While the TMS is fully responsible for drive allocation, VM Client and the TapePlex assist in this process by presenting a list of drives in order of their suitability to satisfy mounts of specific and scratch volumes.

The following section describes the interaction between the TMS allocation interface and the VM Client.

Allocation Interaction

The TMS service machine receives a request from a virtual machine to mount a volume and invokes the allocation interface routine.

If an IUCV path to the VM Client service machine is not established, the tape management system attempts to establish one. If the attempt fails, no allocation assistance takes place, a return code indicates that condition, and a flag may be set to inform the message interface that operator message processing cannot take place since no special allocation has been done.

Allocation Dialog

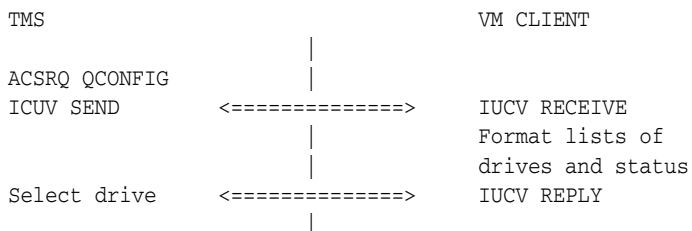
1. The TMS allocation interface sends a request, using an IUCV message, to the VM Client service machine:

```
ACSRQ QDRLIST,VOLSER=voladr
```

An alternative request is:

```
ACSRQ QDRLIST,VOLSER=voladr,COUNT=,LIST=
```

The request above includes the COUNT= and LIST= parameters. These two parameters describe a list of devices considered eligible by the TMS. This information is passed to the VM Client.

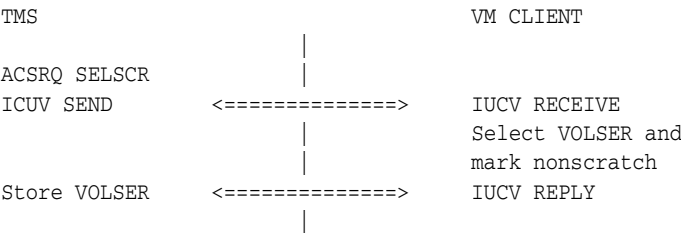


2. The allocation interface waits for an IUCV REPLY

- 3. A VM Client routine collects the data, formats the response, and issues an IUCV REPLY. The IUCV REPLY is mapped by the SLX macro.
- 4. The TMS IUCV support functions notify the waiting allocation process that a response has been received.
- 5. The allocation interface routine then reformats the reply into a drive preference list (TMS dependent format), comparing the reply to the TMS-managed available drives, and leaves the interface.
- 6. If scratch selection is needed, issue:

```
ACSRQ SELSCR,DRIVE=drivadr
```

This returns a VOLSER and marks the volume as nonscratch in the control data set, or indicates that no scratch volume is available.



Termination of allocation Interface

Control is returned to the TMS when the allocation interface routine completes processing.

Operation Message Processing

This section describes the interactions between the TMS message interface and the VM Client. The “Operator Message Dialog” section outlines the processing of a MOUNT request. Other message interface point requests (for example, DISMOUNT, SCRATCH), follow a similar sequence of events and are not separately described. The TMS may suppress or change the message based on the completion of the process.

A DISMOUNT request is issued in response to conditions detected by the TMS that require a volume dismount (for example, the mounted volume is not the one requested), or in the case where the TMS always dismounts volumes after use.

A SCRATCH request is issued by the TMS to return "work" volumes to scratch status. The HSC control data set on the server is updated to reflect these changes.

Operator Message Interaction

The TMS service machine receives a request from a virtual machine to mount a volume. The TMS service machine must have an IUCV path established to send commands to the VM Client. A drive has already been selected.

Operator Message Dialog

- 1. The message interface code determines that the request is for a drive.

2. The message interface may send a request, using an IUCV message, to the VM Client service machine to obtain location information.

`ACSRQ QVOLUME,VOLSER=voladr`
3. The message interface waits for an IUCV REPLY
4. The VM Client determines the volume status, adds the location data, and issues an IUCV REPLY. If volume status information is retained from the allocation routine, the four previous steps may be omitted.
5. If the volume is in the TapePlex, the operator message interface sends a request, using an IUCV message to the service machine, specifying the volume to be mounted and the drive to be used.

`ACSRQ MOUNT,VOLSER=volser,DRIVE=drivadr,PROTECT=`

6. The message interface waits for an IUCV REPLY
7. The VM Client MOUNT routine requests the server to perform the mount, formats a success/failure response, and issues a reply. The IUCV REPLY to the originating message is mapped by the SLX macro.

Note: A MOUNT request directed to a transport drive containing an unloaded volume causes a dismount of that volume followed by the requested mount.

8. TMS IUCV support routines notify the waiting message interface that a response has been received.
9. The message interface routine examines the reply to determine if the mount was successful, sets an appropriate return code, and leaves the interface.

Termination of Operator Message Interface

Control returns to the tape management system when the message interface routine completes processing.

PROP-Detected Dismount

When a StorageTek drive attached to a virtual machine is detached, or the virtual machine is logged off, a DETACH message is issued to the VM system operator. If a drive becomes detached while a library or virtual volume is mounted, the TMS may not be notified, and would not issue a normal DISMOUNT message to the VM Client service machine.

To properly handle the dismount, a VM PROP (PRogrammable OPerator) facility should intercept certain messages and process accordingly

Note: Like the TMS machine, the PROP machine must be authorized by the VM Client service machine for commands. The function called using the PROP RTABLE must have the name of the VM Client service machine available.

SMCPROP EXEC, LOGTAPE SAMPLE, and RTABLE SAMPLE are supplied as examples. The EXECs can be used unchanged or modified to suit the environment. These should be set up to execute similar to the following sequence:

1. PROP detects the DETACH message and invokes a routine (SMCPROP EXEC) to check if a library (or virtual) volume was previously mounted on the drive (saved using SMCPROP EXEC). If so, execute the following command:

CP SMSG vmclientuser DISMOUNT DRIVE cuu
2. SMCPROP EXEC issues the command using the CP SMSG interface to the VM Client service machine.
3. VM Client receives the dismount request.

Scenario A - Normal Dismount

If the volume is on the drive, the dismount is processed normally and the process is complete. For example:

Example 9-2 PROP-Detected Dismount Scenario A - Normal Dismount

PROP		VM CLIENT
Receive msg:		
.SLS#124I MOUNT OF vvvvv1 ON		
DRIVE cuu - COMPLETE		
Save volume and transport address		
Receive msg:		
TAPE cuu DETACHED ...		
Use drive address to get saved		
volume.		
If a volume was previously saved		
for the transport, issue dismount:		
CP SMSG smcuser DISMOUNT DRIVE cuu		====> Receive dismount request
		Process dismount
		DISMOUNT OF vvvvv1 FROM
		DRIVE cuu COMPLETE

Scenario B - Dismount Processed Automatically

If the drive has already been reallocated and a mount request is issued, VM Client finds the previous volume on the drive and automatically starts dismount processing for that volume. When this automatic dismount completes, the new volume is mounted. For example:

Example 9-3 PROP-Detected Dismount Scenario B - Dismount Processed Automatically

PROP		VM CLIENT
Receive msg:		
.SLS#124I MOUNT OF vvvvv1 ON		
DRIVE cuu - COMPLETE		
Save volume and transport address		
Receive msg:		
TAPE cuu DETACHED ...		
		Receive mount for vvvvv2
		from TMS

Use transport address to get save volume. Issue dismount: CP SMSG smcuser DISMOUNT DRIVE cuu Receive msg: .SLS0124I MOUNT OF vvvvv2 ON DRIVE cuu - COMPLETE Save volume and transport address.	Issue msg: .SLS##81I VOLUME vvvvv1 FOUND MOUNTED ON DRIVE cuu ATTEMPTING DISMOUNT. Process automatic dismount of vvvvvv1 Process mount of vvvvv2 =====> Receive dismount for vvvvv1 from PROP
---	---

ACSRQ Macro

The tape management system interface uses the ACSRQ invocation macro to prepare an IUCV message that contains a request for the VM Client. After the ACSRQ macro returns control to its caller, the TMS routine must issue an IUCV SEND.

ACSRQ Requests

The types of requests used to interact with the library include:

- query information
- set environment parameters
- volume processing

Invoke all VM Client requests through the ACSRQ macro instruction. In general, specify the name of the function to be performed, the address of the data area to be sent (ACSINT), and the other keyword parameters required.

The ACSRQ macro builds the ACS Interface Block (ACSINT) and optionally builds an IPARML for an IUCV SEND. An IUCV instruction referencing the IPARML which has been built should be coded after the ACSRQ macro. The receipt of the ACSINT invokes the proper routines in the VM Client and returns information to the sender using an IUCV REPLY.

The IUCV restrictions documented in the IBM publications VM/SP System Facilities for Programming and VM/XA CP Programming Services apply. Due to the data area sizes, PRMMSG is not supported. BUFLIST and ANSLIST are also not supported.

TMS		VM CLIENT
ASRQ xxxxxxxx		
IUCV SEND	<=====>	IUCV RECEIVE
WAIT		Process Request
	<=====>	IUCV REPLY
(External Interrupt)		

Check return code	
Process reply	

See "SLX Macro Mapping" for mapping of the reply area.

ACSRQ Macro Syntax

The following is the syntax for the ACSRQ macro:

Figure 9–1 ACSRQ macro syntax



Function is one of the following:

- DISMOUNT
- EJECT
- MOUNT
- MOVE
- QCAP
- QCONFIG
- QDRIVES
- QDRLIST
- QSCRATCH
- QVOLUME
- QVOLUME
- QVOLUME
- SCRATCH
- SELSCR

Parameter is one of the following:

- ,ACCT1=*acct1addr*
- ,ACCT2=*acct2taddr*
- ,CAP=*capidaddr*
- ,COL=*coladdr*
- ,COUNT=*countaddr*
- ,DRIVE=*driveaddr*
- ,HOSTID=*hostidaddr*
- ,IPARML=YES
- ,LIST=*listaddr*
- ,LSM=*lsmidaddr*
- ,MEDIA=*medaddr*
- ,MGMTCLS=*mgmtcls*

- ,NOTIFY=INSDEL/NOINSDEL
- ,PAN=*paneladdr*
- ,PATHID=*pathadr*
- ,PROTECT=YES
- ,RECTECH=*recaddr*
- ,ROW=*rowaddr*
- ,RSPADDR=*bufadr*
- ,RSPLEN=*buflen*
- ,SCRATCH=YES
- ,SUBPOOL=*subpooladdr*
- ,TEXT=*textaddr*
- ,TOLSM=*lsmidaddr*
- ,TOPAN=*paneladdr*
- ,USER=*useridaddr*
- ,VOLSER=*voladdr*

The tape management interface (TMI), which enables users to request query information, volume movement, and scratch volume control services from the VM Client, includes requests that allow media and recording technique to be specified.

The requests that can use media and recording technique information include:

- MOUNT
- QDRLIST
- QSCRATCH
- SELSCR

These requests are described on the following pages.

The TMI determines media and recording technique values for a request by using **MEDIA** and **RECTECH** parameters directly.

Note: If **DSECT=YES** is specified, no other functions or parameters are valid. An **ACSINT DSECT** is built.

DISMOUNT

The **DISMOUNT** request causes a cartridge to be removed from a specific drive. A library cartridge is moved to an LSM cell (selected by the HSC) and becomes available for future requests. A virtual volume remains resident in the VTSS and uses **MGMTCLAS** definitions to determine migration, replication, and buffer residency policies.

Considerations

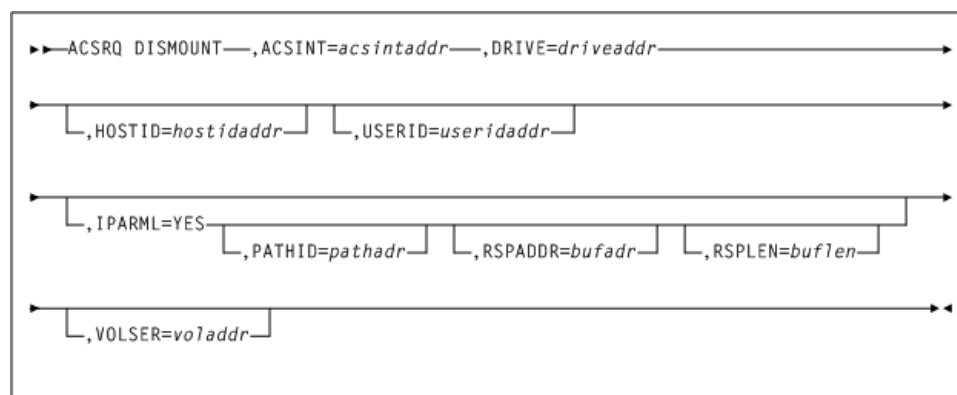
The success of a **DISMOUNT** request depends on whether the volume has received a **REWIND/UNLOAD CCW**. If the drive hasn't yet received a **REWIND/UNLOAD CCW**, the **DISMOUNT** request is aborted.

A **DISMOUNT** request may cancel a previous **MOUNT** request for the same drive.

Syntax

The following figure show syntax for the ACSRQ DISMOUNT request:

Figure 9–2 ACSRQ DISMOUNT request syntax



Parameters

ACSINT=acsintaddr

specifies the address of the data area being sent to the VM Client service machine. This parameter is required.

acsintaddr is an RX-type data address or the number of the register containing the data address. ACSRQ references the ACSINT at this address when filling in the data.

DRIVE=driveaddr

specifies the drive from which a volume should be dismounted. This parameter is required.

driveaddr is the address of the 2-byte drive specification (*ccua*). Specify an RX-type address of the data or the number of the register containing the data address.

HOSTID=hostidaddr

optionally, specifies the address of an eight character host ID, left justified and padded with blanks. If this parameter is not specified, the host ID executing the request is used.

hostidaddr is an RX-type host ID address or the register (2) - (12) containing the host ID address

IPARML=YES

optionally, specifies that the requester has established addressability to an IUCV IPARML, and that the IPARML is initialized.

PATHID=pathadr

optionally, specifies the address of a 2-byte IUCV path ID of the library service machine.

pathadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND PATHID statement.

PATHID is valid only if IPARML=YES is specified. If PATHID is not specified, the subsequent IUCV SEND must specify it.

RSPADDR=bufadr

optionally, specifies the address of the IUCV answer buffer.

bufaddr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND ANSBUF statement.

RSPADDR is valid only if IPARML=YES is specified. If RSPADDR is not specified, the subsequent IUCV SEND must specify it.

RSPLLEN=buflen

optionally, specifies the address of the length of the IUCV answer buffer.

buflen is an RX-type address of the 2-byte field or the number of the register containing the data address. This data is used in the IUCV SEND ANSLEN statement.

RSPLEN is valid only if IPARML=YES is specified. If RSPLEN is not specified, the subsequent IUCV SEND must specify it. The response length must be at least the value returned in SLXZDISM by QCONFIG.

USER=useridaddr

optionally, specifies an 8-byte user ID associated with a console ID for the request.

useridaddr is an RX-type address of the data or the number of the register containing the data address.

If USER is not specified, the user ID executing the request is used.

VOLSER=voladdr

optionally, specifies the VOLSER of the volume to be dismounted.

voladdr is the address of a 6-character volume label; either an RX-type address of the data or the number of the register containing the data address.

An error will occur if the mounted volume has a different VOLSER.

Request Response

The response to a DISMOUNT request is generated when all cartridge movement associated with the request has completed. The response contains a Reply Header and a Message Text Element. The reason code in the Reply Header (SLXSRC) is a binary message number that indicates which HSC message was issued when the DISMOUNT request completed. The Message Text Element contains the complete text of the message specified by the reason code.

See "SLX Macro Mapping" for information on the SLX macro.

EJECT

The EJECT request initiates the removal of one or more (up to 500) cartridges from the library. The cartridges are moved from LSM cells to the highest preference CAP or to a CAP specified in the request, so they can be retrieved by an operator

Considerations

A request to eject a virtual volume, or a volume not defined in the control data set is considered to be invalid.

The length of the response may vary considerably, depending on the number of volumes specified in the request. Several values are available in the response from a QCONFIG request for use in determining the appropriate answer buffer length for a particular EJECT request. These values include:

- SLXZEJC1 contains the length of an EJECT response for a single volume. Use this value for the answer buffer length when an EJECT request specifies either VOLSER= or COUNT=1.
- SLXXVOLL contains the length of a single Volume Information Element and SLXXMSGSL contains the length of a single Message Text Element. When an EJECT request specifies COUNT=n, then the answer buffer length is computed using the formula: $((n-1) * (SLXXVOLL + SLXXMSGSL)) + SLXZEJC1$.
- SLXZEJCT contains the length of an EJECT response when the maximum number of VOLSERs (500) is specified in the request list. Use this value for the answer buffer length when the above formula cannot be used, and when the requester can afford to commit a large amount of storage (approximately 78KB) to the request.

Syntax

The following figure show syntax for the ACSRQ EJECT request:

Figure 9–3 ACSRQ EJECT request syntax



Parameters

ACSINT=acsintaddr

specifies the address of the data area being sent to the VM Client service machine. This parameter is required.

acsintaddr is an RX-type data address or the number of the register containing the data address. ACSRQ references the ACSINT at this address when filling in the data.

CAP=capidaddr

optionally, specifies the address of the CAP used to satisfy the request.

capidaddr is the RX-type address of the data or the number of the register containing the data address.

The format is *AALLCC00*, where *AA* is the ACS number (00-99 decimal), *LL* is the LSM number (00-99 decimal), and *CC* is the CAP number in decimal. These identifiers are always followed by 00.

COUNT=*countaddr*

optionally, specifies the address of a 2-byte field containing the number of VOLSERS in the list designated by the **LIST** parameter.

countaddr is an RX-type address of the data or the number of the register that contains the address of the data.

COUNT is required with the **LIST** parameter and is mutually exclusive with the **VOLSER** parameter.

HOSTID=*hostidaddr*

optionally, specifies the address of an eight character host ID, left justified and padded with blanks. If this parameter is not specified, the host ID executing the request is used.

hostidaddr is an RX-type host ID address or the register (2) - (12) containing the host ID address

IPARML=YES

optionally, specifies that the requester has established addressability to an IUCV IPARML, and that the IPARML is initialized.

LIST=*listaddr*

optionally, specifies the address of the list of the elements.

listaddr is an RX-type address of the data or the number of a register that contains the data address. Each element in this list is a 6-byte VOLSER.

A special form of this parameter, **LIST=*** indicates to ACSRQ that the list is already appended to the ACSINT data area, and does not need to be moved.

LIST is required with the **COUNT** parameter and is mutually exclusive with the **VOLSER** parameter.

LSMID=*lsmidaddr*

optionally, specifies the address of the LSMid from which the volumes are ejected. If the CAP is not available in the specified LSM, the request fails. If the user does not specify LSM, the HSC chooses a single CAPid in the ACS of the first volume in the list. The format of an LSMid is *AALL*, where *AA* is the ACS number (decimal) and *LL* is the LSM number (decimal). For example, 0102 is ACS 01, LSM 02.

lsmidaddr is an RX address of the data or the number of the register containing the address of the LSMid.

PATHID=*pathadr*

optionally, specifies the address of a 2-byte IUCV path ID of the library service machine.

pathadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV **SEND PATHID** statement.

PATHID is valid only if **IPARML=YES** is specified. If PATHID is not specified, the subsequent IUCV **SEND** must specify it.

RSPADDR=*bufadr*

optionally, specifies the address of the IUCV answer buffer.

bufadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND ANSBUF statement.

RSPADDR is valid only if IPARML=YES is specified. If RSPADDR is not specified, the subsequent IUCV SEND must specify it.

RSPLN=buflen

optionally, specifies the address of the length of the IUCV answer buffer.

buflen is an RX-type address of the 2-byte field or the number of the register containing the data address. This data is used in the IUCV SEND ANSLEN statement.

RSPLN is valid only if IPARML=YES is specified. If RSPLN is not specified, the subsequent IUCV SEND must specify it. The response length must be at least the value returned in SLXZDISM by QCONFIG.

SEQ=NO|YES

optionally, specifies whether CAP eject processing fills the CAP cells sequentially or by home location distance.

- NO specifies that the EJECT process order the requested volumes by home location. EJECT fills the CAP or magazine (for the SL8500) according to the volume home location distance to the CAP; that is, volumes closest to the CAP are ejected first.
- YES specifies that the EJECT process place cartridges in the CAP beginning with the topmost available CAP cell and continuing sequentially.

Note: The SEQ parameter is effective for all LSM types but is used primarily for the SL8500 environment. If sequential order is desired for other LSM types, you must code SEQ=YES.

TEXT=textaddr

optionally, specifies a 32-character text string for association with the request.

textaddr is an RX-type address of the data or the number of the register containing the address of the data.

USERID=userisaddr

optionally, specifies an 8-byte user ID associated with a console ID for the request.

useridaddr is an RX-type address of the data or the number of the register containing the address of the data.

If USER is not specified, the user ID executing the request is used.

VOLSER=voladdr

optionally, specifies the address of a 6-character volume label.

voladdr is an RX-type address of the data or the number of the register containing the address of the data.

VOLSER is mutually exclusive with the LIST and COUNT parameters. Either VOLSER or LIST and COUNT must be specified.

Request Response

The response from an EJECT request consists of one Reply Header and one Volume Information Element, and one Message Text Element for each VOLSER that was specified in the request. Volume Information Elements and Message Text Elements appear in the same order as the VOLSERs in the request.

See "SLX Macro Mapping" for information on the SLX macro.

MOUNT

The MOUNT request causes a volume to be mounted on a specific drive.

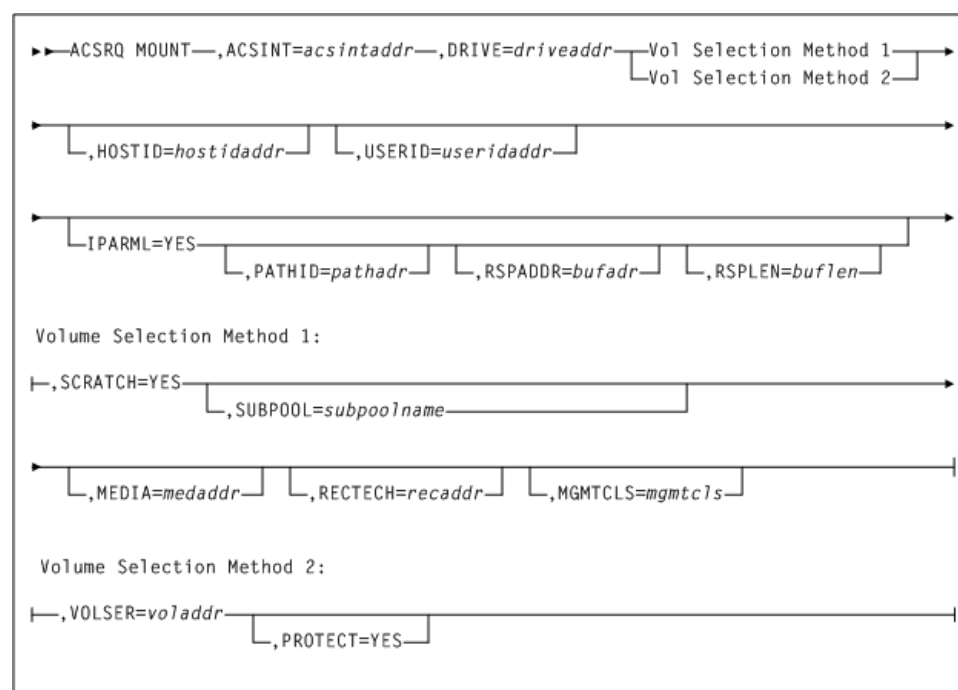
Considerations

An automatic dismount will occur when a MOUNT request is directed to a drive that contains an unloaded cartridge.

Syntax

The following figure show syntax for the ACSRQ MOUNT request:

Figure 9-4 ACSRQ MOUNT request syntax



Parameters

The ACSRQ MOUNT request includes the following parameters:

ACSINT=*acsintaddr*

specifies the address of the data area being sent to the VM Client service machine. This parameter is required.

acsintaddr is an RX-type data address or the number of the register containing the data address. ACSRQ references the ACSINT at this address when filling in the data.

DRIVE=*driveaddr*

specifies the drive on which the volume is to be mounted; the address of the 2-byte drive specification (*ccua*). This parameter is required.

driveaddr an RX-type address of the data or the number of the register containing the data address.

HOSTID=hostidaddr

optionally, specifies the address of an eight character host ID, left justified and padded with blanks. If this parameter is not specified, the host ID executing the request is used.

hostidaddr is an RX-type host ID address or the register (2) - (12) containing the host ID address

MEDIA=medaddr

optionally, specifies the address of an 8-byte character field containing the media type of the cartridge to be mounted.

If MEDIA is not specified, the next compatible scratch cartridge is mounted without regard to media type.

medaddr is an RX-type address of the data or the register (2) - (12) containing the address of the data.

MGMTCLAS=mgmtclas

optionally, specifies the address of an eight character field containing the management class left justified and padded with blanks.

mgmtclas is an RX-type address of the data or the register (2) - (12) containing the address of the data.

If MGMTCLS is not specified, but SUBPOOL is specified, then MGMTCLS may be set based upon the VM Client POOLmap command.

IPARM=YES

optionally, specifies that the requester has established addressability to an IUCV IPARML, and that the IPARML is initialized.

PATHID=pathadr

optionally, specifies the address of a 2-byte IUCV path ID of the library service machine.

pathadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND PATHID statement.

PATHID is valid only if IPARML=YES is specified. If PATHID is not specified, the subsequent IUCV SEND must specify it.

PROTECT=YES

optionally, specifies that the volume should be write protected. If PROTECT=YES is not specified, the physical position of the thumbwheel determines whether the volume is write protected.

PROTECT=YES is valid only with VOLSER.

RSPADDR=bufadr

optionally, specifies the address of the IUCV answer buffer.

bufadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND ANSBUF statement.

RSPADDR is valid only if IPARML=YES is specified. If RSPADDR is not specified, the subsequent IUCV SEND must specify it.

RSPLEN=buflen

optionally, specifies the address of the length of the IUCV answer buffer.

buflen is an RX-type address of the 2-byte field or the number of the register containing the data address. This data is used in the IUCV SEND ANSLEN statement.

RSPLLEN is valid only if IPARML=YES is specified. If RSPLLEN is not specified, the subsequent IUCV SEND must specify it. The response length must be at least the value returned in SLXZDISM by QCONFIG.

SCRATCH=YES

optionally, specifies that the request is for a nonspecific (scratch) volume. A scratch VOLSER is selected at this time and mounted on the specified transport.

Either SCRATCH=YES or VOLSER must be specified.

SUBPOOL=*subpoolname*

optionally, specifies the address of a 13-character field containing the name of the scratch subpool.

subpoolname is an RX-type address of the data or the number of the register containing the data address. SCRPOOL (subpool index) is no longer supported; you must use the SUBPOOL parameter to select a scratch pool.

SUBPOOL is valid only if SCRATCH=YES is specified. If SUBPOOL is specified, but MGMTCLS is not specified, then MGMTCLS may be set based upon the VM Client POOLmap command.

USER=*useridaddr*

optionally, specifies an 8-byte user ID associated with a console ID for the request.

useridaddr is an RX-type address of the data or the number of the register containing the data address.

If USER is not specified, the user ID executing the request is used.

VOLSER=*voladdr*

optionally, specifies the address of a 6-character volume label.

voladdr is an RX-type address of the data or the number of the register containing the data address.

Either VOLSER or SCRATCH=YES must be specified. This parameter specifies the VOLSER of the volume to be mounted. Either VOLSER or SCRATCH=YES must be specified.

Request Response

The response to a MOUNT request is generated when all cartridge movement associated with the request has completed. The response contains a Reply Header, a Message Text Element, and if the request specified SCRATCH=YES, a Volume Information Element. The reason code in the Reply Header (SLXSRC) is a binary message number that indicates which HSC message was issued when the MOUNT request completed. The Message Text Element contains the complete text of the message specified by the reason code. The Volume Information Element is present when the request specified SCRATCH=YES and describes the scratch volume that was mounted.

MOVE

The MOVE request causes a volume to be moved to a specific location in the ACS.

Considerations

The MOVE function enables movement of a single volume to another location within an ACS. The destination of moved volumes may be the same LSM or a different LSM.

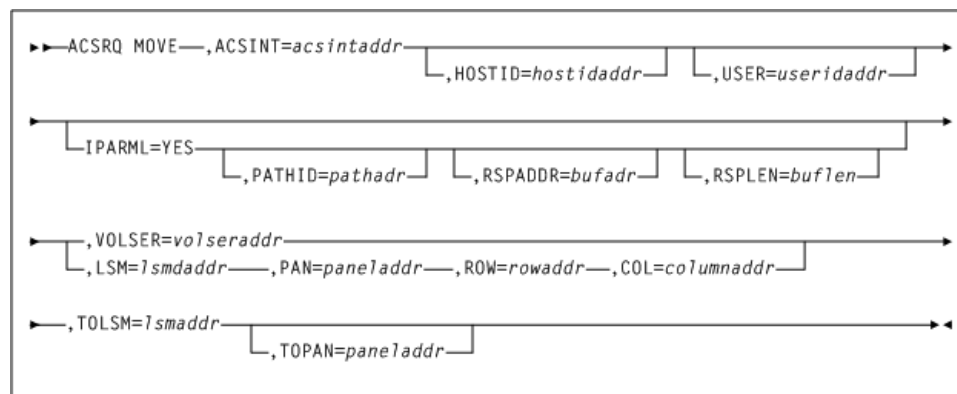
The MOVE functions provides volume movement and improved tape management control.

A request to move a virtual volume, or a volume not defined in the control data set is considered to be invalid.

Syntax

The following figure show syntax for the ACSRQ MOVE request:

Figure 9–5 ACSRQ MOVE request syntax



Parameters

The ACSRQ MOVE request includes the following parameters:

ACSINT=acsintaddr

specifies the address of the data area being sent to the VM Client service machine. This parameter is required.

acsintaddr is an RX-type data address or the number of the register containing the data address. ACSRQ references the ACSINT at this address when filling in the data.

HOSTID=hostidaddr

optionally, specifies the address of an eight character host ID, left justified and padded with blanks. If this parameter is not specified, the host ID executing the request is used.

hostidaddr is an RX-type host ID address or the register (2) - (12) containing the host ID address

USER=useridaddr

optionally, specifies an 8-byte user ID associated with a console ID for the request.

useridaddr is an RX-type address of the data or the number of the register containing the data address.

If USER is not specified, the user ID executing the request is used.

IPARML=YES

optionally, specifies that the requester has established addressability to an IUCV IPARML, and that the IPARML is initialized.

PATHID=pathaddr

optionally, specifies the address of a 2-byte IUCV path ID of the library service machine.

pathaddr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND PATHID statement.

PATHID is valid only if IPARML=YES is specified. If PATHID is not specified, the subsequent IUCV SEND must specify it.

RSPADDR=bufadr

optionally, specifies the address of the IUCV answer buffer.

bufadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND ANSBUF statement.

RSPADDR is valid only if IPARML=YES is specified. If RSPADDR is not specified, the subsequent IUCV SEND must specify it.

RSPLEN=buflen

optionally, specifies the address of the length of the IUCV answer buffer.

buflen is an RX-type address of the 2-byte field or the number of the register containing the data address. This data is used in the IUCV SEND ANSLEN statement.

RSPLEN is valid only if IPARML=YES is specified. If RSPLEN is not specified, the subsequent IUCV SEND must specify it. The response length must be at least the value returned in SLXZDISM by QCONFIG.

VOLSER=volseraddr

optionally, specifies the address of a 6-character volume label.

voladdr is an RX-type address of the data or the number of the register containing the data address.

Either VOLSER or SCRATCH=YES must be specified. This parameter specifies the VOLSER of the volume to be mounted. Either VOLSER or SCRATCH=YES must be specified.

LSM=lsmidr

optionally, specifies the address of an LSMid. The format of an LSMid is *AALL*, where *AA* is the ACS number (decimal) and *LL* is the LSM number (decimal). For example, 0110 is ACS 01, LSM 10.

lsmidr is an RX-type address of the LSMid or the register (2) - (12) containing the address of the LSMid.

The COL, PAN, and ROW parameters must accompany the LSM parameter. This parameter is required if VOL is not specified.

PAN=paneladdr

optionally, specifies the address of a panel number. Format of the panel number is *pp*, where *pp* is a decimal number.

paneladdr is an RX address of the panel or the register (2-12) containing the address of the panel number.

PAN is required if LSM is specified.

ROW=rowaddr

optionally, specifies the address of a row number. Format of the row number is *rr* where *rr* is a decimal number.

rowaddr is an RX address of the row or the register (2-12) containing the address of the row number.

ROW is required if LSM is specified.

COL=columnaddr

optionally, specifies the address of a column number. Format of the column number is *cc*, where *cc* is a decimal number.

coladdr is an RX address of the column or the register (2-12) containing the address of the column number.

COL is required if LSM is specified.

TOLSM=lsaddr

specifies the address of the LSMid where the volume is moved. The LSMid is two hexadecimal bytes in the format *AALL*, where *AA* is the ACS number (00-FF hexadecimal) and *LL* is the LSM number (*LL* is 00-17 hexadecimal). This parameter is required.

lsaddr is an RX-type address of the LSMid or the register (2) - (12) containing the address of the LSMid.

TOPAN=paneladdr

optionally, specifies the address of a panel number. This parameter is required.

paneladdr is an RX address of the panel or the register (2-12) containing the address of the panel number.

Request Response

The response to a MOVE request is generated when all cartridge movement associated with the request has completed. The response contains a Reply Header, a Message Text Element, and if the request was successful, one Volume Information Element. The reason code in the Reply Header (SLXSRC) is a binary message number that indicates which HSC message was issued when the MOVE request completed. The Message Text Element contains the complete text of the message specified by the reason code.

See ["SLX Macro Mapping"](#) for information on the SLX macro.

QCAP

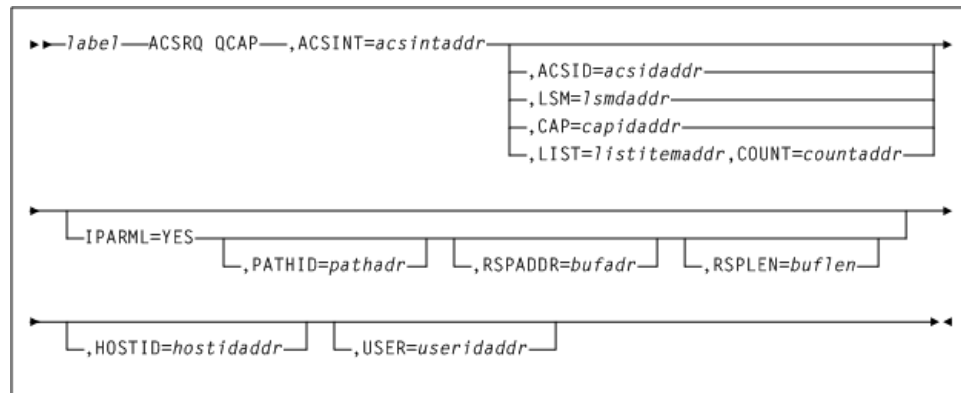
This request enables you to query the capacity and status of a CAP.

Considerations

If ACSID, LSM, CAP, or LIST and COUNT, are not specified, the data returned is for all CAPs.

Syntax

The following figure show syntax for the ACSRQ QCAP request:

Figure 9–6 ACSRQ QCAP request syntax

Parameters

The ACSRQ QCAP request includes the following parameters:

ACSID=acsidaddr

specifies the address of the data area being sent to the VM Client service machine. This parameter is required.

acsintaddr is an RX-type data address or the number of the register containing the data address. ACSRQ references the ACSINT at this address when filling in the data.

ACSINT=acsintaddr

specifies the address of the data area being sent to the VM Client service machine. This parameter is required.

acsintaddr is an RX-type data address or the number of the register containing the data address. ACSRQ references the ACSINT at this address when filling in the data.

CAP=capidaddr

optionally, specifies the address of the CAP used to satisfy the request.

capidaddr is the RX-type address of the data or the number of the register containing the data address.

The format of *capidaddr* is *AALLCC00*, where *AA* is the ACS number (decimal), *LL* is the LSM number (decimal), and *CC* is the CAP number. These identifiers are always followed by 00.

If CAP is specified, information about the specified CAP is returned.

COUNT=countaddr

optionally, specifies the address of a 2-byte field containing the number of CAPIDs in the list designated by the LIST parameter.

countaddr is an RX-type address of the data or the number of the register that contains the address of the data.

COUNT is required with the LIST parameter.

HOSTID=hostidaddr

optionally, specifies the address of an eight character host ID, left justified and padded with blanks. If this parameter is not specified, the host ID executing the request is used.

hostidaddr is an RX-type host ID address or the register (2) - (12) containing the host ID address

IPARML=YES

optionally, specifies that the requester has established addressability to an IUCV IPARML, and that the IPARML is initialized.

LIST=listitemaddr

optionally, specifies the address of the list of CAPs to be queried.

listitemaddr is an RX-type address of the data or the number of the register containing the address of the data.

If LIST is specified, information about all CAPs in the list is returned.

LSM=lsmidaddr

optionally, specifies the address of an LSMid. The format of an LSMid is *AALL*, where *AA* is the ACS number (00-FF hexadecimal) and *LL* is the LSM number (*LL* is 00-17 hexadecimal). For example, 0102 is ACS 01, LSM 02. All values are in hexadecimal format.

lsmidaddr is an RX-type address of the LSMid or the register (2) - (12) containing the address of the LSMid.

If LSM is specified, the returned data is for the specific LSM. If ACSID, LSM, CAP, or LIST and COUNT, are not specified, the data returned is for all CAPs.

PATHID=pathidaddr

optionally, specifies the address of a 2-byte IUCV path ID of the library service machine.

pathadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND PATHID statement.

PATHID is valid only if IPARML=YES is specified. If PATHID is not specified, the subsequent IUCV SEND must specify it.

RSPADDR=bufadr

optionally, specifies the address of the IUCV answer buffer.

bufadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND ANSBUF statement.

RSPADDR is valid only if IPARML=YES is specified. If RSPADDR is not specified, the subsequent IUCV SEND must specify it.

RSPLEN=buflen

optionally, specifies the address of the length of the IUCV answer buffer.

buflen is an RX-type address of the 2-byte field or the number of the register containing the data address. This data is used in the IUCV SEND ANSLEN statement.

RSPLEN is valid only if IPARML=YES is specified. If RSPLEN is not specified, the subsequent IUCV SEND must specify it. The response length must be at least the value returned in SLXZDISM by QCONFIG.

USER=useridaddr

optionally, specifies an 8-byte user ID associated with a console ID for the request.

useridaddr is an RX-type address of the data or the number of the register containing the data address.

If USER is not specified, the user ID executing the request is used.

Request Response

The response to the QCAP request contains a Reply Header and a CAP information element containing information about each CAP requested.

See "SLX Macro Mapping" for information on the SLX macro.

QCONFIG

The QCONFIG request enables you to obtain summary information about the TapePlex configuration and the recommended answer buffer lengths for other TMS interface requests.

Considerations

The QCONFIG request should be the first request issued after an IUCV connection has been established because its response contains a recommended answer buffer length (response length) for each type of TMS interface request.

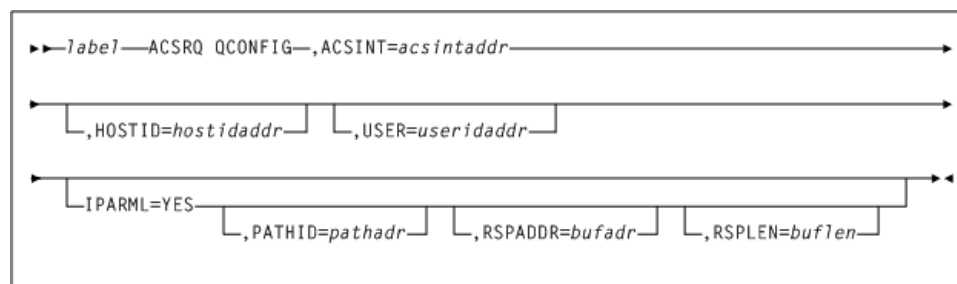
The length of the QCONFIG response may change from release to release. A TMS should use the following technique to obtain the recommended answer buffer length for a QCONFIG request:

1. Issue a QCONFIG request with answer buffer length of decimal 16. The response from this request consists of a Reply Header, truncated to 16 bytes. The return code in the header is 4, which indicates that the answer buffer was too small to contain the entire response. The word at offset decimal 12, SLXCRLN, contains the recommended answer buffer length for a QCONFIG request.
2. Reissue the QCONFIG request using the recommended answer buffer length.

Syntax

The following figure show syntax for the ACSRQ QCONFIG request:

Figure 9–7 ACSRQ QCONFIG request syntax



Parameters

The ACSRQ QCONFIG request includes the following parameters:

ACSINT=acsintaddr

specifies the address of the data area being sent to the VM Client service machine. This parameter is required.

acsintaddr is an RX-type data address or the number of the register containing the data address. ACSRQ references the ACSINT at this address when filling in the data.

HOSTID=hostidaddr

optionally, specifies the address of an eight character host ID, left justified and padded with blanks. If this parameter is not specified, the host ID executing the request is used.

hostidaddr is an RX-type host ID address or the register (2) - (12) containing the host ID address

IPARML=YES

optionally, specifies that the requester has established addressability to an IUCV IPARML, and that the IPARML is initialized.

PATHID=pathadr

optionally, specifies the address of a 2-byte IUCV path ID of the library service machine.

pathadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND PATHID statement.

PATHID is valid only if IPARML=YES is specified. If PATHID is not specified, the subsequent IUCV SEND must specify it.

RSPADDR=bufaddr

optionally, specifies the address of the IUCV answer buffer.

bufaddr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND ANSBUF statement.

RSPADDR is valid only if IPARML=YES is specified. If RSPADDR is not specified, the subsequent IUCV SEND must specify it.

RSPLEN=buflen

optionally, specifies the address of the length of the IUCV answer buffer.

buflen is an RX-type address of the 2-byte field or the number of the register containing the data address. This data is used in the IUCV SEND ANSLEN statement.

RSPLEN is valid only if IPARML=YES is specified. If RSPLEN is not specified, the subsequent IUCV SEND must specify it. The response length must be at least the value returned in SLXZDISM by QCONFIG.

USER=useridaddr

optionally, specifies an 8-byte user ID associated with a console ID for the request.

useridaddr is an RX-type address of the data or the number of the register containing the data address.

If USER is not specified, the user ID executing the request is used.

Request Response

The QCONFIG response contains a Reply Header and a Configuration Summary Element. The length of each type of response element (for example, Volume Element) is returned in the Reply Header by QCONFIG.

See ["SLX Macro Mapping"](#) for information on the SLX macro.

QDRIVES

The QDRIVES request enables you to obtain detailed information about all transports and LSMs associated with the library, or with a particular ACS.

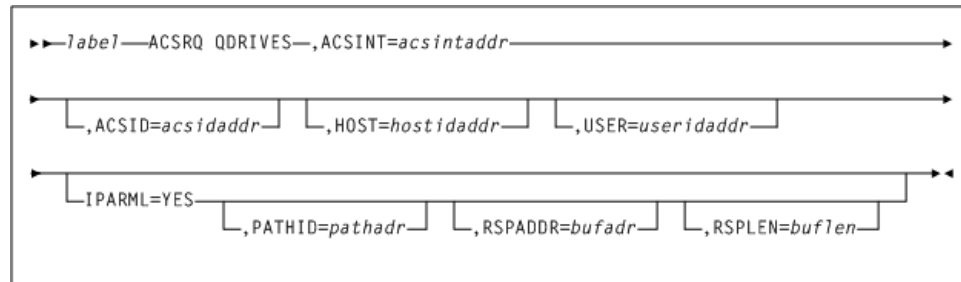
Considerations

None.

Syntax

The following figure show syntax for the ACSRQ QDRIVES request:

Figure 9–8 ACSRQ QDRIVES request syntax



Parameters

The ACSRQ QDRIVES request includes the following parameters:

ACSID=acsidaddr

specifies the address of the data area being sent to the VM Client service machine. This parameter is required.

acsintaddr is an RX-type data address or the number of the register containing the data address. ACSRQ references the ACSINT at this address when filling in the data.

ACSINT=acsintaddr

specifies the address of the data area being sent to the VM Client service machine. This parameter is required.

acsintaddr is an RX-type data address or the number of the register containing the data address. ACSRQ references the ACSINT at this address when filling in the data.

HOSTID=hostidaddr

optionally, specifies the address of an eight character host ID, left justified and padded with blanks. If this parameter is not specified, the host ID executing the request is used.

hostidaddr is an RX-type host ID address or the register (2) - (12) containing the host ID address

IPARML=YES

optionally, specifies that the requester has established addressability to an IUCV IPARML, and that the IPARML is initialized.

PATHID=pathadr

optionally, specifies the address of a 2-byte IUCV path ID of the library service machine.

pathadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND PATHID statement.

PATHID is valid only if IPARML=YES is specified. If PATHID is not specified, the subsequent IUCV SEND must specify it.

RSPADDR=bufadr

optionally, specifies the address of the IUCV answer buffer.

bufadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND ANSBUF statement.

RSPADDR is valid only if IPARML=YES is specified. If RSPADDR is not specified, the subsequent IUCV SEND must specify it.

RSPLEN=*buflen*

optionally, specifies the address of the length of the IUCV answer buffer.

buflen is an RX-type address of the 2-byte field or the number of the register containing the data address. This data is used in the IUCV SEND ANSLEN statement.

RSPLEN is valid only if IPARML=YES is specified. If RSPLEN is not specified, the subsequent IUCV SEND must specify it. The response length must be at least the value returned in SLXZDISM by QCONFIG.

USER=*useridaddr*

optionally, specifies an 8-byte user ID associated with a console ID for the request.

useridaddr is an RX-type address of the data or the number of the register containing the data address.

If USER is not specified, the user ID executing the request is used.

Request Response

The QDRIVES response consists of a Reply Header, a Drive Information section, and an LSM Information section. The Drive Information section contains one Drive Information Element for each transport in the library or ACS. The LSM Information section contains one LSM Information Element for each LSM in the library or ACS.

See ["SLX Macro Mapping"](#) for information on the SLX macro.

QDRLIST

The QDRLIST request enables you to obtain the TapePlex recommendation for a library transport to be specified for a subsequent MOUNT request.

Considerations

The VM Client makes its recommendation by returning a list of Drive Information Elements which are ordered so that the first element describes the best transport to use, the second element describes the second best transport to use, and so on.

When the QDRLIST request specifies a particular cartridge (that is, VOLSER is specified), the Drive Information Elements are arranged so that the first transport listed is in the LSM that is closest to (or the same as) the LSM containing the cartridge. The last transport listed is in the LSM that is farthest from the LSM containing the cartridge. Only transports in the same ACS as the cartridge are represented in the Drive Information section.

When the QDRLIST request specifies a scratch volume (that is, SCRATCH=YES is specified), the Drive Information Elements are arranged so the first transport listed is in the LSM containing the most scratch volumes. The last transport listed is in the LSM containing the fewest scratch volumes. All transports in all ACSs are represented in the Drive Information section.

For requests for a specific virtual volume, either the drives in the VTSS where the volume is resident or can be recalled are returned. For requests for a virtual scratch volume, drives in VTSSs that support the requested management class are returned.

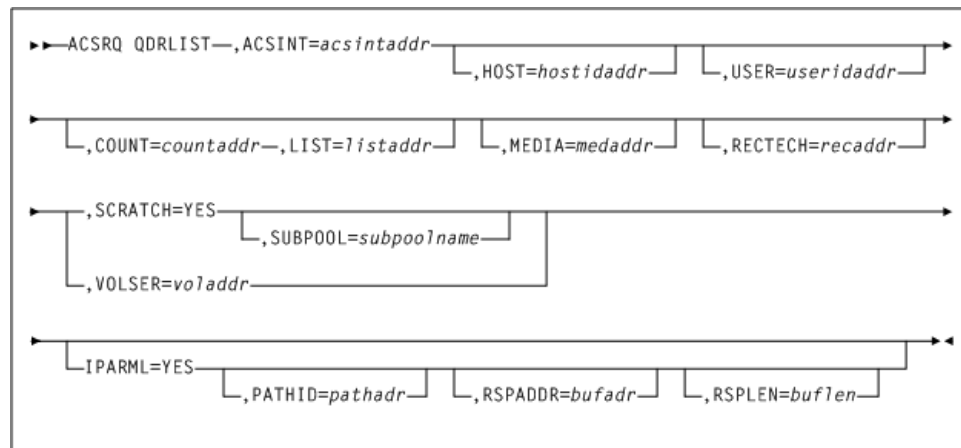
The VM Client ignores whether a transport already has a volume mounted, or is in an offline LSM or ACS when it orders the Drive Information Elements.

An optional list of transport addresses may be supplied with the QDRLIST request. If a list is provided, the VM Client uses it as a screen while building its response. A Drive Information Element is included in the response only when its transport address is present in the list.

Syntax

The following figure show syntax for the ACSRQ QDRLIST request:

Figure 9–9 ACSRQ QDRLIST request syntax



Parameters

The ACSRQ QDRLIST request includes the following parameters:

ACSINT=acsintaddr

specifies the address of the data area being sent to the VM Client service machine. This parameter is required.

acsintaddr is an RX-type data address or the number of the register containing the data address. ACSRQ references the ACSINT at this address when filling in the data.

COUNT=countaddr

optionally, specifies the address of a 2-byte field containing the number of CAPIDs in the list designated by the LIST parameter.

countaddr is an RX-type address of the data or the number of the register that contains the address of the data.

COUNT is required with the LIST parameter.

MEDIA=medaddr

optionally, specifies the address of an 8-byte character field containing the media type of the cartridge to be mounted.

If MEDIA is not specified, the next compatible scratch cartridge is mounted without regard to media type.

medaddr is an RX-type address of the data or the register (2) - (12) containing the address of the data.

RECTECH=*recaddr*

optionally, specifies the address of an 8-byte field containing the recording technique used to record data tracks on the tape surface.

If RECTECH is not specified, transports are selected depending on the MEDIA type that has been specified.

recaddr is an RX-type address of the data or the register (2) - (12) containing the address of the data.

See "Recording Technique (RECTECH)" for a list of valid recording-technique values.

HOSTID=*hostidaddr*

optionally, specifies the address of an eight character host ID, left justified and padded with blanks. If this parameter is not specified, the host ID executing the request is used.

hostidaddr is an RX-type host ID address or the register (2) - (12) containing the host ID address

IPARML=YES

optionally, specifies that the requester has established addressability to an IUCV IPARML, and that the IPARML is initialized.

LIST=*listaddr*

optionally, specifies the address of the list of the elements.

listaddr is an RX-type address of the data or the number of a register that contains the address of the data.

Each element in this list is a 2-byte drive address (*ccua*).

A special form of this parameter, LIST=* indicates to ACSRQ that the list is already appended to the ACSINT data area, and does not need to be moved.

PATHID=*pathadr*

optionally, specifies the address of a 2-byte IUCV path ID of the library service machine.

pathadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND PATHID statement.

PATHID is valid only if IPARML=YES is specified. If PATHID is not specified, the subsequent IUCV SEND must specify it.

RSPADDR=*bufadr*

optionally, specifies the address of the IUCV answer buffer.

bufadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND ANSBUF statement.

RSPADDR is valid only if IPARML=YES is specified. If RSPADDR is not specified, the subsequent IUCV SEND must specify it.

RSPLEN=*buflen*

optionally, specifies the address of the length of the IUCV answer buffer.

buflen is an RX-type address of the 2-byte field or the number of the register containing the data address. This data is used in the IUCV SEND ANSLEN statement.

RSPLEN is valid only if IPARML=YES is specified. If RSPLEN is not specified, the subsequent IUCV SEND must specify it. The response length must be at least the value returned in SLXZDISM by QCONFIG.

SCRATCH=YES

optionally, specifies that the request is for a nonspecific (scratch) volume. A scratch VOLSER is selected at this time and mounted on the specified transport.

Either SCRATCH=YES or VOLSER must be specified.

SUBPOOL=subpoolname

optionally, specifies the address of a 13-character field containing the name of the scratch subpool.

subpoolname is an RX-type address of the data or the number of the register containing the address of the data. SCRPOOL (subpool index) is no longer supported; you must use the SUBPOOL parameter to select a scratch pool.

SUBPOOL is valid only if SCRATCH=YES is specified.

USER=useridaddr

optionally, specifies an 8-byte user ID associated with a console ID for the request.

useridaddr is an RX-type address of the data or the number of the register containing the data address.

If USER is not specified, the user ID executing the request is used.

VOLSER=voladdr

optionally, specifies the address of a 6-character volume label.

voladdr is an RX-type address of the data or the number of the register containing the data address.

Either VOLSER or SCRATCH=YES must be specified. This parameter specifies the VOLSER of the volume to be mounted. Either VOLSER or SCRATCH=YES must be specified.

Request Response

The QDRLIST response always contains a Reply Header and a Drive Information section. If VOLSER was specified, the response also contains a Volume Information Element.

See "[SLX Macro Mapping](#)" for information on the SLX macro.

QSCRATCH

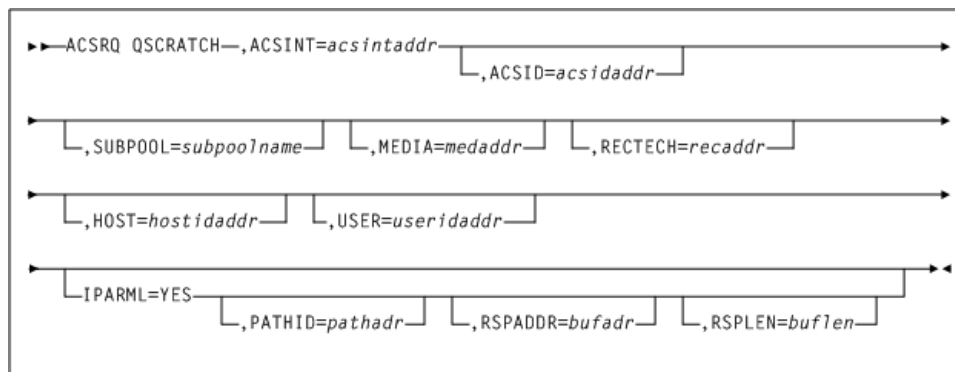
The QSCRATCH request enables you to obtain detailed information about all LSMs associated with the library, or with a particular ACS. This information includes the number of scratch volumes in each LSM.

Considerations

When the library and its associated control data set are shared by more than one HSC, then the reported scratch totals may differ from the true totals because they may not account for recent scratch volume activity on other processors. However, each HSC refreshes its scratch volume totals from the control data set every five minutes so the variance should be slight.

Syntax

The following figure show syntax for the ACSRQ QSCRATCH request:

Figure 9–10 ACSRQ QSCRATCH request syntax

Parameters

The ACSRQ QSCRATCH request includes the following parameters:

ACSID=acsidaddr

specifies the address of the data area being sent to the VM Client service machine. This parameter is required.

acsintaddr is an RX-type data address or the number of the register containing the data address. ACSRQ references the ACSINT at this address when filling in the data.

ACSINT=acsintaddr

specifies the address of the data area being sent to the VM Client service machine. This parameter is required.

acsintaddr is an RX-type data address or the number of the register containing the data address. ACSRQ references the ACSINT at this address when filling in the data.

MEDIA=medaddr

optionally, specifies the address of an 8-byte character field containing the media type of the requested scratch cartridge.

If MEDIA is not specified, scratch cartridges are selected without regard to media type.

medaddr is an RX-type address of the data or the register (2) - (12) containing the address of the data.

See ["Media Type \(MEDIA\)"](#) for a list of valid media-type values.

RECTECH=recaddr

optionally, specifies the address of an 8-byte field containing the recording technique used to record data tracks on the tape surface.

This parameter is optional. If RECTech is not specified, scratch cartridges are selected depending on the MEDIA type that has been specified.

recaddr is an RX-type address of the data or the register (2) - (12) containing the address of the data.

See ["Recording Technique \(RECTech\)"](#) for a list of valid recording-technique values.

HOSTID=hostidaddr

optionally, specifies the address of an eight character host ID, left justified and padded with blanks. If this parameter is not specified, the host ID executing the request is used.

hostidaddr is an RX-type host ID address or the register (2) - (12) containing the host ID address

IPARML=YES

optionally, specifies that the requester has established addressability to an IUCV IPARML, and that the IPARML is initialized.

PATHID=pathadr

optionally, specifies the address of a 2-byte IUCV path ID of the library service machine.

pathadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND PATHID statement.

PATHID is valid only if IPARML=YES is specified. If PATHID is not specified, the subsequent IUCV SEND must specify it.

RSPADDR=bufadr

optionally, specifies the address of the IUCV answer buffer.

bufadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND ANSBUF statement.

RSPADDR is valid only if IPARML=YES is specified. If RSPADDR is not specified, the subsequent IUCV SEND must specify it.

RSPLN=buflen

optionally, specifies the address of the length of the IUCV answer buffer.

buflen is an RX-type address of the 2-byte field or the number of the register containing the data address. This data is used in the IUCV SEND ANSLEN statement.

RSPLN is valid only if IPARML=YES is specified. If RSPLN is not specified, the subsequent IUCV SEND must specify it. The response length must be at least the value returned in SLXZDISM by QCONFIG.

SUBPOOL=subpoolname

optionally, specifies the address of a 13-character field containing the name of the scratch subpool.

subpoolname is Specify either an RX-type address of the data or the number of the register containing the address of the data. SCRPOOL (subpool index) is no longer supported; you must use the SUBPOOL parameter to select a scratch pool.

SUBPOOL is valid only if SCRATCH=YES is specified.

USER=useridaddr

optionally, specifies an 8-byte user ID associated with a console ID for the request.

useridaddr is an RX-type address of the data or the number of the register containing the data address.

If USER is not specified, the user ID executing the request is used.

Request Response

The QSCRATCH response contains a Reply Header and an LSM Information Element for each LSM.

See "[SLX Macro Mapping](#)" for information on the SLX macro.

QVOLUME

The QVOLUME request enables you to obtain the current library status of one or more (up to 500) cartridges.

Considerations

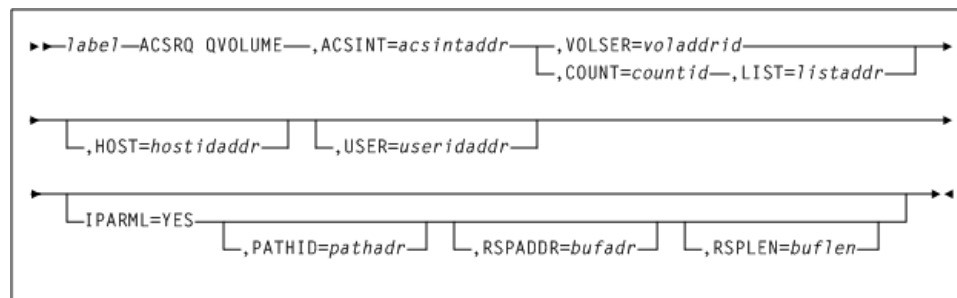
The length of the response may vary considerably, depending on the number of volumes specified in the request. Several values are available in the response from a QCONFIG request for use in determining the appropriate answer buffer length for a particular QVOLUME request. These values include:

- SLXZQVOL contains the length of a QVOLUME response for a single volume. Use this value for the answer buffer length when a QVOLUME request specifies either VOLSER or COUNT=1.
- SLXXVOLL contains the length of a single Volume Information Element. When a QVOLUME request specifies COUNT=n, then the answer buffer length is computed using the formula: $((n-1) * SLXXVOLL) + SLXZQVOL$.
- SLXZVOL contains the length of a QVOLUME response when the maximum number of VOLSERS (500) is specified in the request list. Use this value for the answer buffer length when the above formula cannot be used, and when the requester can afford to commit a large amount of storage (approximately 16KB) to the request.

Syntax

The following figure show syntax for the ACSRQ QVOLUME request:

Figure 9–11 ACSRQ QVOLUME request syntax



Parameters

The ACSRQ QVOLUME request includes the following parameters:

ACSINT=acsintaddr

specifies the address of the data area being sent to the VM Client service machine. This parameter is required.

acsintaddr is an RX-type data address or the number of the register containing the data address. ACSRQ references the ACSINT at this address when filling in the data.

COUNT=countid

optionally, specifies the address of a 2-byte field containing the number of CAPIDs in the list designated by the LIST parameter.

countaddr is an RX-type address of the data or the number of the register that contains the address of the data.

COUNT is required with the LIST parameter.

HOSTID=hostidaddr

optionally, specifies the address of an eight character host ID, left justified and padded with blanks. If this parameter is not specified, the host ID executing the request is used.

hostidaddr is an RX-type host ID address or the register (2) - (12) containing the host ID address

IPARML=YES

optionally, specifies that the requester has established addressability to an IUCV IPARML, and that the IPARML is initialized.

LIST=listaddr

optionally, specifies the address of the list of the elements.

listaddr is an RX-type address of the data or the number of a register that contains the address of the data.

Each element in this list is a 2-byte drive address (*ccua*).

A special form of this parameter, LIST=* indicates to ACSRQ that the list is already appended to the ACSINT data area, and does not need to be moved.

PATHID=pathadr

optionally, specifies the address of a 2-byte IUCV path ID of the library service machine.

pathadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND PATHID statement.

PATHID is valid only if IPARML=YES is specified. If PATHID is not specified, the subsequent IUCV SEND must specify it.

RSPADDR=bufadr

optionally, specifies the address of the IUCV answer buffer.

bufadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND ANSBUF statement.

RSPADDR is valid only if IPARML=YES is specified. If RSPADDR is not specified, the subsequent IUCV SEND must specify it.

RSPLEN=buflen

optionally, specifies the address of the length of the IUCV answer buffer.

buflen is an RX-type address of the 2-byte field or the number of the register containing the data address. This data is used in the IUCV SEND ANSLEN statement.

RSPLEN is valid only if IPARML=YES is specified. If RSPLEN is not specified, the subsequent IUCV SEND must specify it. The response length must be at least the value returned in SLXZDISM by QCONFIG.

USER=useridaddr

optionally, specifies an 8-byte user ID associated with a console ID for the request.

useridaddr is an RX-type address of the data or the number of the register containing the data address.

If USER is not specified, the user ID executing the request is used.

VOLSER=voladdr

optionally, specifies the address of a 6-character volume label.

Either VOLSER or SCRATCH=YES must be specified. This parameter specifies the VOLSER of the volume to be mounted. Either VOLSER or SCRATCH=YES must be specified.

The QVOLUME response consists of a Reply Header and a Volume Information Element for each VOLSER that was specified in the request. Volume Information Elements appear in the same order as the VOLSERS in the request.

See ["SLX Macro Mapping"](#) for information on the SLX macro.

The QVOLUSE request enables you to obtain the current status of mounted volumes.

The length of the response may vary considerably, depending on the number of volumes returned by the request.

The following figure show syntax for the ACSRQ QVOLUSE request:

```

graph LR
    label --> ACSQ
    ACSQ --> QVOLUSE
    QVOLUSE --> ACSINT
    ACSINT --> host
    host --> user
    user --> IPARML
    IPARML --> pathid
    pathid --> rspaddr
    rspaddr --> rsplen
    rsplen --> end
  
```

Diagram illustrating the structure of the ACSQ QVOLUSE command. The command is composed of several fields connected sequentially:

- `label` (start)
- `ACSQ`
- `QVOLUSE`
- `ACSINT=acsintaddr`
- `HOST=hostidaddr`
- `USER=useridaddr`
- `IPARML=YES`
- `PATHID=pathaddr`
- `RSPADDR=bufaddr`
- `RSPLEN=buflen`
- (end)

The ACSRQ QVOLUSE request includes the following parameters:

specifies the address of the data area being sent to the VM Client service machine. This parameter is required.

optionally, specifies the address of an eight character host ID, left justified and padded with blanks. If this parameter is not specified, the host ID executing the request is used.

VM Client Tape Management Interface 9-39

IPARML=YES

optionally, specifies that the requester has established addressability to an IUCV IPARML, and that the IPARML is initialized.

PATHID=pathadr

optionally, specifies the address of a 2-byte IUCV path ID of the library service machine.

pathadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND PATHID statement.

PATHID is valid only if IPARML=YES is specified. If PATHID is not specified, the subsequent IUCV SEND must specify it.

RSPADDR=bufadr

optionally, specifies the address of the IUCV answer buffer.

bufadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND ANSBUF statement.

RSPADDR is valid only if IPARML=YES is specified. If RSPADDR is not specified, the subsequent IUCV SEND must specify it.

RSPLLEN=buflen

optionally, specifies the address of the length of the IUCV answer buffer.

buflen is an RX-type address of the 2-byte field or the number of the register containing the data address. This data is used in the IUCV SEND ANSLEN statement.

RSPLLEN is valid only if IPARML=YES is specified. If RSPLLEN is not specified, the subsequent IUCV SEND must specify it. The response length must be at least the value returned in SLXZDISM by QCONFIG.

USER=useridaddr

optionally, specifies an 8-byte user ID associated with a console ID for the request.

useridaddr is an RX-type address of the data or the number of the register containing the data address.

If USER is not specified, the user ID executing the request is used.

Request Response

The QVOLUSE response consists of a Reply Header and a Volume Information Element for each VOLSER that was returned by the request. Volume Information Elements appear in the same order as the VOLSERS returned by the request.

See "SLX Macro Mapping" for information on the SLX macro.

SCRATCH

The SCRATCH request causes a volume to be placed in scratch status in the server control data set.

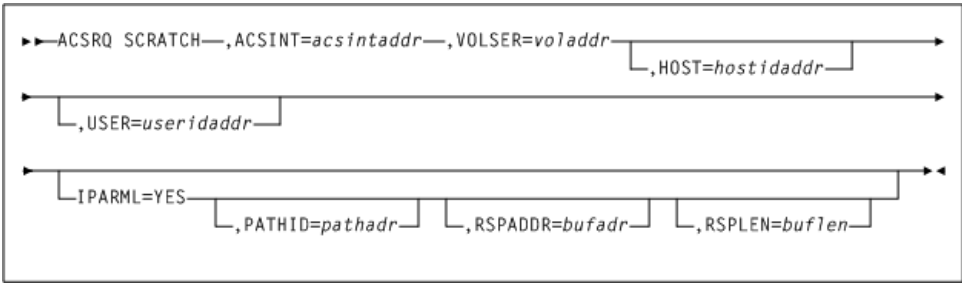
Considerations

The specified cartridge must already be in the library. No cartridge movement occurs as the result of a SCRATCH request. However, SCRATCH request processing must select the cartridge (that is, must acquire exclusive use of the cartridge) to change its state. As a result, a SCRATCH request will fail if the cartridge is mounted on a drive.

Syntax

The following figure show syntax for the ACSRQ SCRATCH request:

Figure 9–13 *ACSRQ SCRATCH request syntax*



Parameters

The ACSRQ_SCRATCH request includes the following parameters:

ACSINT=acsintaddr

specifies the address of the data area being sent to the VM Client service machine. This parameter is required.

acsintaddr is an RX-type data address or the number of the register containing the data address. ACSRQ references the ACSINT at this address when filling in the data.

HOSTID=*hostidaddr*

optionally, specifies the address of an eight character host ID, left justified and padded with blanks. If this parameter is not specified, the host ID executing the request is used.

hostidaddr is an RX-type host ID address or the register (2) - (12) containing the host ID address

IPARML=YES

optionally, specifies that the requester has established addressability to an IUCV IPARML, and that the IPARML is initialized.

PATHID=*pathadr*

optionally, specifies the address of a 2-byte IUCV path ID of the library service machine.

pathadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND PATHID statement.

PATHID is valid only if IPARML=YES is specified. If PATHID is not specified, the subsequent IUCV SEND must specify it.

RSPADDR=*bufadr*

optionally, specifies the address of the IUCV answer buffer.

bufadr is an RX-type data address or the number of the register containing the data address. This data is used in the **IUCV SEND ANSBUF** statement.

RSPADDR is valid only if IPARML=YES is specified. If RSPADDR is not specified, the subsequent IUCV SEND must specify it.

RSPLLEN=*buflen*

optionally, specifies the address of the length of the IUCV answer buffer.

buflen is an RX-type address of the 2-byte field or the number of the register containing the data address. This data is used in the IUCV SEND ANSLEN statement.

RSPLN is valid only if IPARML=YES is specified. If RSPLN is not specified, the subsequent IUCV SEND must specify it. The response length must be at least the value returned in SLXZDISM by QCONFIG.

USER=useridaddr

optionally, specifies an 8-byte user ID associated with a console ID for the request.

useridaddr is an RX-type address of the data or the number of the register containing the data address.

If USER is not specified, the user ID executing the request is used.

VOLSER=voladdr

optionally, specifies the address of a 6-character volume label.

voladdr is an RX-type address of the data or the number of the register containing the data address.

Either VOLSER or SCRATCH=YES must be specified. This parameter specifies the VOLSER of the volume to be mounted. Either VOLSER or SCRATCH=YES must be specified.

Request Response

The response to SCRATCH contains only a Reply Header.

See "[SLX Macro Mapping](#)" for information on the SLX macro.

SELSCR

The SELSCR request causes the server to select a library scratch volume and remove it from scratch status in the control data set.

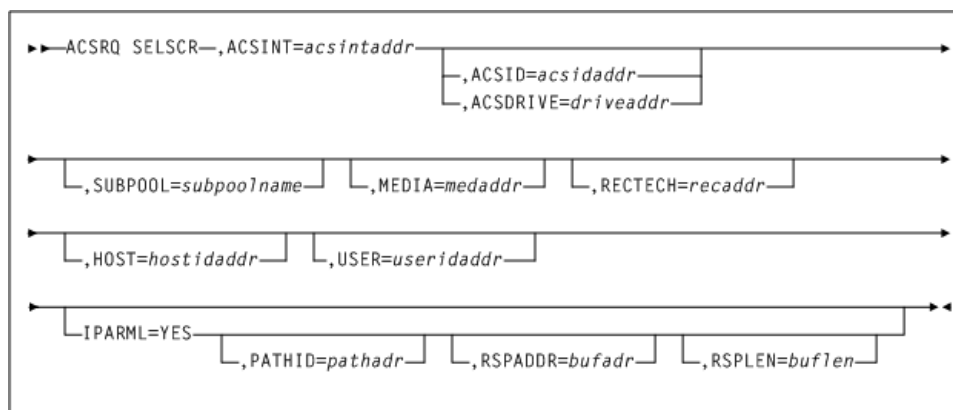
Considerations

No volume movement occurs.

When neither ACSID nor DRIVE is specified, the HSC searches through all library LSMs and chooses a scratch volume from the LSM containing the most scratch volumes. When ACSID is specified, the HSC chooses a scratch volume from the LSM in the specified ACS that holds the most cartridges. When DRIVE is specified the HSC chooses a scratch volume from the closest LSM, if the drive is in an automatic mode LSM.

Syntax

The following figure show syntax for the ACSRQ SELSCR request:

Figure 9–14 ACSRQ SELSCR request syntax

Parameters

The ACSRQ SELSCR request includes the following parameters:

ACSID=*acsidaddr*

specifies the address of the data area being sent to the VM Client service machine. This parameter is required.

acsintaddr is an RX-type data address or the number of the register containing the data address. ACSRQ references the ACSINT at this address when filling in the data.

ACSINT=*acsintaddr*

specifies the address of the data area being sent to the VM Client service machine. This parameter is required.

acsintaddr is an RX-type data address or the number of the register containing the data address. ACSRQ references the ACSINT at this address when filling in the data.

ACSDRIVE=*driveaddr*

optionally, specifies the address of the 2-byte drive specification (*ccua*).

driveaddr is an RX-type address of the data or the number of the register containing the address of the data.

DRIVE is mutually exclusive with ACSID. It specifies the drive which the scratch volume should be near.

MEDIA=*medaddr*

optionally, specifies the address of an 8-byte character field containing the media type of the selected scratch cartridge. If MEDIA is not specified, scratch cartridges are selected without regard to media type.

medaddr is an RX-type address of the data or the register (2) - (12) containing the address of the data.

See ["Media Type \(MEDIA\)"](#) for a list of valid media-type values.

RECTECH=*recaddr*

optionally, specifies the address of an 8-byte field containing the recording technique used to record data tracks on the tape surface.

If RECTECH is not specified, scratch cartridges are selected depending on the MEDIA type that has been specified.

recaddr is an RX-type address of the data or the register (2) - (12) containing the address of the data.

See "[Recording Technique \(RECTech\)](#)" for a list of valid recording-technique values.

HOSTID=hostidaddr

optionally, specifies the address of an eight character host ID, left justified and padded with blanks. If this parameter is not specified, the host ID executing the request is used.

hostidaddr is an RX-type host ID address or the register (2) - (12) containing the host ID address

IPARML=YES

optionally, specifies that the requester has established addressability to an IUCV IPARML, and that the IPARML is initialized.

PATHID=pathadr

optionally, specifies the address of a 2-byte IUCV path ID of the library service machine.

pathadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND PATHID statement.

PATHID is valid only if IPARML=YES is specified. If PATHID is not specified, the subsequent IUCV SEND must specify it.

RSPADDR=bufadr

optionally, specifies the address of the IUCV answer buffer.

bufadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND ANSBUF statement.

RSPADDR is valid only if IPARML=YES is specified. If RSPADDR is not specified, the subsequent IUCV SEND must specify it.

RSPLEN=buflen

optionally, specifies the address of the length of the IUCV answer buffer.

buflen is an RX-type address of the 2-byte field or the number of the register containing the data address. This data is used in the IUCV SEND ANSLEN statement.

RSPLEN is valid only if IPARML=YES is specified. If RSPLEN is not specified, the subsequent IUCV SEND must specify it. The response length must be at least the value returned in SLXZDISM by QCONFIG.

SUBPOOL=subpoolname

optionally, specifies the address of a 13-character field containing the name of the scratch subpool.

subpoolname is an RX-type address of the data or the number of the register containing the address of the data. SCRPOOL (subpool index) is no longer supported; you must use the SUBPOOL parameter to select a scratch pool.

SUBPOOL is valid only if SCRATCH=YES is specified.

USER=useridaddr

optionally, specifies an 8-byte user ID associated with a console ID for the request.

useridaddr is an RX-type address of the data or the number of the register containing the data address.

If USER is not specified, the user ID executing the request is used.

Request Response

The response to SELSCR contains a Reply Header and a Volume Information Element. The Reply Header contains a return code (SLXCMDRC) indicating the success of the operation. The Volume Information Element describes the selected volume.

See "SLX Macro Mapping" for information on the SLX macro.

UNSCRATCH

The UNSCRATCH request causes a volume to be removed from scratch status in the control data set.

Considerations

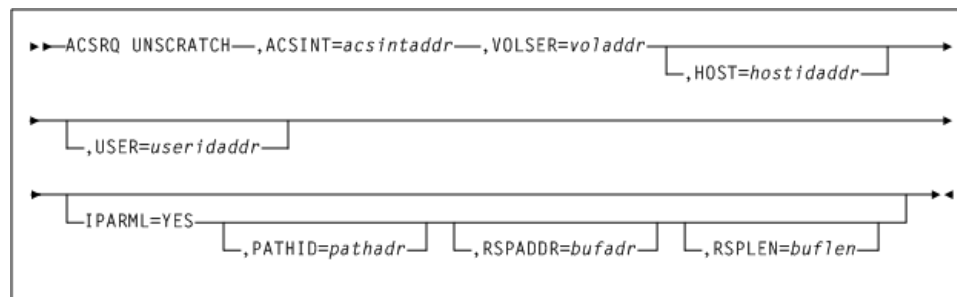
A request to unscratch of volume not defined in the control data set is considered to be invalid.

For real volumes, no cartridge movement occurs because of the UNSCRATCH request. However, UNSCRATCH request processing must select the volume (that is, must acquire exclusive use of the volume) to change its state. As a result, an UNSCRATCH request will fail if the volume is mounted on a real or virtual drive (see "Considerations").

Syntax

The following figure show syntax for the ACSRQ UNSCRATCH request:

Figure 9–15 ACSRQ UNSCRATCH request syntax



Parameters

The ACSRQ UNSCRATCH request includes the following parameters:

ACSINT=acsintaddr

specifies the address of the data area being sent to the VM Client service machine. This parameter is required.

acsintaddr is an RX-type data address or the number of the register containing the data address. ACSRQ references the ACSINT at this address when filling in the data.

HOSTID=hostidaddr

optionally, specifies the address of an eight character host ID, left justified and padded with blanks. If this parameter is not specified, the host ID executing the request is used.

hostidaddr is an RX-type host ID address or the register (2) - (12) containing the host ID address

IPARML=YES

optionally, specifies that the requester has established addressability to an IUCV IPARML, and that the IPARML is initialized.

PATHID=pathadr

optionally, specifies the address of a 2-byte IUCV path ID of the library service machine.

pathadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND PATHID statement.

PATHID is valid only if IPARML=YES is specified. If PATHID is not specified, the subsequent IUCV SEND must specify it.

RSPADDR=bufadr

optionally, specifies the address of the IUCV answer buffer.

bufadr is an RX-type data address or the number of the register containing the data address. This data is used in the IUCV SEND ANSBUF statement.

RSPADDR is valid only if IPARML=YES is specified. If RSPADDR is not specified, the subsequent IUCV SEND must specify it.

RSPLEN=buflen

optionally, specifies the address of the length of the IUCV answer buffer.

buflen is an RX-type address of the 2-byte field or the number of the register containing the data address. This data is used in the IUCV SEND ANSLEN statement.

RSPLEN is valid only if IPARML=YES is specified. If RSPLEN is not specified, the subsequent IUCV SEND must specify it. The response length must be at least the value returned in SLXZDISM by QCONFIG.

USER=useridaddr

optionally, specifies an 8-byte user ID associated with a console ID for the request.

useridaddr is an RX-type address of the data or the number of the register containing the data address.

If USER is not specified, the user ID executing the request is used.

VOLSER=voladdr

optionally, specifies the address of a 6-character volume label.

voladdr is an RX-type address of the data or the number of the register containing the data address.

Either VOLSER or SCRATCH=YES must be specified. This parameter specifies the VOLSER of the volume to be mounted. Either VOLSER or SCRATCH=YES must be specified.

Request Response

The response to UNSCRATCH contains only a Reply Header.

See "[SLX Macro Mapping](#)" for information on the SLX macro.

Interface Data Areas

This section describes interface data areas for the SLX macro.

SLX Macro

A reply always begins with a header. The header may be followed by one or more “sections”. Each section is a table of “elements” of a particular type (for example, Volume Information Element). If the Reply Header is aligned on a doubleword boundary, then all subsequent sections and elements are guaranteed to also begin on doubleword boundaries.

A section directory (number/offset/length) is defined in the Reply Header for each possible section type, even though no reply will ever contain all types of sections. The order in which the section directories appear within the header has no relationship to the order in which sections are physically arranged after the header.

A section directory’s number specifies how many elements of that type are actually present in the reply. If a section directory’s number is nonzero, then that section directory’s offset specifies the offset, from the start of the reply header, to the first (or only) element of that type. If a section directory’s number is greater than one, then that section directory’s length, which specifies the length of a single element of that type, must be used to access the second and subsequent elements of that type. For example, add the length to the offset to get the offset to the second element; add in the length again to get the offset to the third element; and so on. The number of elements in each section is variable. The following tables denote which sections of the reply will be returned for a given request.

The following table provides an SLS macro parameter matrix:

Table 9–1 SLS Macro - Parameter Matrix

Reply Section	Header	Config	CAP	Volume	Drive	LSM	Message Text
DISMOUNT	1	NA	NA	NA	NA	NA	1
EJECT	1	NA	NA	*	NA	NA	*
MOUNT	1	NA	NA	1 (if scratch)	NA	NA	1
MOVE	1	NA	NA	1 (if successful)	NA	NA	1
QCAP	1	NA	1+	NA	NA	NA	NA
QCONFIG	1	1	NA	NA	NA	NA	NA
QDRIVES	1	NA	NA	NA	1+	1+	NA
QDRLIST	1	NA	NA	1 (if not scratch)	1+	NA	NA
QEJECT	1	NA	NA	NA	NA	NA	NA
QREQUEST	1	NA	NA	NA	NA	NA	NA
QSCRATCH	1	NA	NA	NA	NA	NA	NA
QVOLUME	1	NA	NA	*	NA	NA	NA
QVOLUME	1	NA	NA	*	NA	NA	NA
SCRATCH	1	NA	NA	NA	NA	1+	NA
SELSCR	1	NA	NA	1	NA	NA	NA
UNSCRATCH	1	NA	NA	NA	NA	NA	NA

Note:

- 1 is equal to 1 and only 1.
 - * (EJECT and QVOLUME) are limited by the number of Volume Information Elements contained in the SLX reply area (maximum of 500).
 - 1+ indicates a number from 1 to n depending on the library configuration.
-

SLX Macro Mapping

The following example shows output for the SLX record format:

Example 9–4 SLX Record Format

SLX - VM CLIENT EXTERNAL INTERFACE REPLY

FUNCTION:

MAPS A REPLY AREA RETURNED BY ONE OF THE FOLLOWING VM CLIENT REQUESTS:

DISMOUNT - DISMOUNT A VOLUME

EJECT - EJECT A VOLUME FROM THE LIBRARY

MOUNT - MOUNT A VOLUME

MOVE - MOVE A VOLUME

QCAP - RETURN CAP SUMMARY

QCONFIG - RETURN CONFIGURATION SUMMARY

QDRIVES - RETURN DRIVE AND LSM INFORMATION

QDRLIST - RETURN DRIVE INFORMATION, ORDERED BY PREFERENCE

QSCRATCH - RETURN LSM INFORMATION, ORDERED BY PREFERENCE

QVOLUME - RETURN VOLUME INFORMATION

SCRATCH - CHANGE A VOLUME'S STATUS TO 'SCRATCH'

SELSCR - SELECT A SCRATCH VOLUME

SPECIAL CONSIDERATIONS:

A REPLY ALWAYS BEGINS WITH A HEADER. THE HEADER MAY BE FOLLOWED BY ONE OR MORE "SECTIONS". EACH SECTION IS A TABLE OF "ELEMENTS" OF A PARTICULAR TYPE (E.G. VOLUME INFORMATION ELEMENT). IF THE REPLY HEADER IS ALIGNED ON A DOUBLEWORD BOUNDARY, THEN ALL SUBSEQUENT SECTIONS AND ELEMENTS ARE GUARANTEED TO ALSO BEGIN ON DOUBLEWORD BOUNDARIES.

A SECTION DIRECTORY (NUMBER/OFFSET/LENGTH) IS DEFINED IN THE REPLY HEADER FOR EACH POSSIBLE SECTION TYPE, EVEN THOUGH NO REPLY WILL EVER CONTAIN ALL TYPES OF SECTIONS. THE ORDER IN WHICH THE SECTION DIRECTORIES APPEAR WITHIN THE HEADER HAS NO RELATIONSHIP TO THE ORDER IN WHICH SECTIONS ARE PHYSICALLY ARRANGED AFTER THE HEADER.

A SECTION DIRECTORY'S NUMBER SPECIFIES HOW MANY ELEMENTS OF THAT TYPE ARE ACTUALLY PRESENT IN THE REPLY. IF A SECTION DIRECTORY'S NUMBER IS NONZERO, THEN THAT SECTION DIRECTORY'S OFFSET SPECIFIES THE OFFSET, FROM THE START OF THE REPLY HEADER, TO THE FIRST (OR ONLY) ELEMENT OF THAT TYPE. IF A SECTION DIRECTORY'S NUMBER IS GREATER THAN ONE, THEN THAT SECTION DIRECTORY'S LENGTH, WHICH SPECIFIES THE LENGTH OF A SINGLE ELEMENT OF THAT TYPE, MUST BE USED TO ACCESS THE SECOND AND SUBSEQUENT ELEMENTS OF THAT TYPE: ADD THE LENGTH TO THE OFFSET TO GET THE OFFSET TO THE SECOND ELEMENT; ADD IN THE LENGTH AGAIN TO GET THE OFFSET TO THE THIRD ELEMENT; AND SO ON.

HEADER

DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
0	(0)	STRUCTURE		SLX	
0	(0)	AREA	1	SLXRPLY	REPLY HEADER

0	(0)	CHARACTER	3	SLXHID	HEADER IDENTIFIER
3	(3)	A-ADDR	1	SLXCMDRC	RETURN CODE:
0	(00)	CONST		SLXRKOK	REQUEST PROCESSED SUCCESSFULLY
4	(04)	CONST		SLXRWARN	REQUEST SUCCESSFUL WITH WARNING SLXSRC WILL PROVIDE THE SPECIFIC REASON FOR THE WARNING
DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
8	(08)	CONST		SLXRBADP	REQUEST FAILED; THE REQUEST BLOCK (MAPPED BY ACSINT) CONTAINED INVALID DATA (E.G., INCOMPATIBLE OPTIONS); SLXSRC (REASON CODE) WILL PROVIDE THE OFFSET OF THE ACSINT FIELD FOUND TO BE IN ERROR.
DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
12	(0C)	CONST		SLXRIERR	REQUEST FAILED; AN UNRECOVERABLE INTERNAL ERROR OCCURRED WHILE PROCESSING THE REQUEST.
DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
16	(10)	CONST		SLXRFAIL	REQUEST FAILED; SLXSRC WILL PROVIDE THE SPECIFIC REASON FOR THE FAILURE.
20	(14)	CONST		SLXRNHSC	REQUEST FAILED - HSC NOT AVAILABLE
44	(2C)	CONST		SLXRBADL	REQUEST FAILED; REPLY AREA PROVIDED BY REQUESTOR WAS TOO SMALL TO CONTAIN ALL REPLY DATA ASSOCIATED WITH THE REQUEST. IF FIELD SLXCRLN IS NON-ZERO, IT CONTAINS THE LENGTH VALUE THAT SHOULD BE SPECIFIED FOR THE REPLY AREA FOR THIS REQUEST.
48	(30)	CONST		SLXRNVC I	VCIRQST AND VCIRESP NOT SUPPORTED. EITHER VTCS IS NOT INSTALLED - OR - IS NOT AT THE REQUIRED LEVEL TO SUPPORT THE PGMI VCI RESPONSES.
DEC	HEX	TYPE	LENGTH	LABEL	
4	(04)	A-ADDR	1	SLXVERS	REPLY VERSION CODE:
7	(07)	CONST	3	SLXVCODE	THIS IS VERSION 9 OF THE REPLY AREA.
5	(5)	HEXSTRING	4	-RESERVED-	RESERVED.
8	(8)	SIGNED-FWORD		SLXSRC	REASON CODE FOR FAILED OPERATION.
32818	(8032)	CONST		SLXTINTR	PGMI TASK INTERRUPTED.
32822	(8036)	CONST		SLXSANF	SEARCH ARGUMENT NOT FOUND.
32826	(803A)	CONST		SLXMSTT	MISMATCHED TOKEN TYPES.
32832	(8040)	CONST		SLXTRNF	TOKEN AREA NOT FOUND.
32848	(8050)	CONST		SLXSFUL	REPLY AREA FULL.

32849	(8051)	CONST		SLXDVM	MEDIA INCOMPATIBLE WITH DEVICE TYPE.
12	(C)	SIGNED-FWORD	4	SLXCRLN	IF RETURN CODE (SLXCMDRC) IS 2C (SLXRBADL), THEN THIS FIELD CONTAINS EITHER THE MINIMUM ACCEPTABLE REPLY AREA LENGTH FOR THE REQUEST, OR 0 IF THE MINIMUM LENGTH COULDN'T BE DETERMINED. OTHERWISE (I.E., RETURN CODE ISN'T 2C), THIS FIELD CONTAINS THE ACTUAL LENGTH OF THIS REPLY.
16	(10)	SIGNED-FWORD	4	SLXPEOFF	PARAMETER ERROR OFFSET IF SLSXRC <> 0 THEN THIS POINTS TO AN ELEMENT IN A LIST WHERE PROCESSING STOPPED WHEN THE REQUEST WAS "QCAP".
20	(14)	LENGTH		SLXHL	TO MAKE COMPATIBLE WITH MVS CODE

CONFIGURATION SUMMARY SECTION DIRECTORY

DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
20	(14)	SIGNED-FWORD	4	SLXXCFGN	NUMBER OF CONFIGURATION ELEMENTS PRESENT IN THIS REPLY.
24	(18)	SIGNED-FWORD	4	SLXXCFGO	OFFSET TO CONFIGURATION SECTION, FROM START OF REPLY, OR 0 IF REPLY DOESN'T CONTAIN ANY CONFIGURATION ELEMENTS.
28	(1C)	SIGNED-FWORD	4	SLXXCFGL	LENGTH OF A CONFIGURATION ELEMENT.

VOLUME INFORMATION SECTION DIRECTORY

DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
32	(20)	SIGNED-FWORD	4	SLXXVOLN	NUMBER OF VOLUME ELEMENTS PRESENT IN THIS REPLY.
36	(24)	SIGNED-FWORD	4	SLXXVOLO	OFFSET TO VOLUME SECTION, FROM START OF REPLY, OR 0 IF REPLY DOESN'T CONTAIN ANY VOLUME ELEMENTS.
40	(28)	SIGNED-FWORD	4	SLXXVOLL	LENGTH OF A VOLUME ELEMENT.

DRIVE INFORMATION SECTION DIRECTORY

DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
44	(2C)	SIGNED-FWORD	4	SLXXDRVN	NUMBER OF DRIVE ELEMENTS PRESENT IN THIS REPLY.
48	(30)	SIGNED-FWORD	4	SLXXDRVO	OFFSET TO DRIVE SECTION, FROM START OF REPLY, OR 0 IF REPLY DOESN'T CONTAIN ANY DRIVE ELEMENTS.
52	(34)	SIGNED-FWORD	4	SLXXDRVL	LENGTH OF A DRIVE ELEMENT.

LSM INFORMATION SECTION DIRECTORY

DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
-----	-----	------	--------	-------	-------------

56	(38)	SIGNED-FWORD	4	SLXXLSMN	NUMBER OF LSM ELEMENTS PRESENT IN THIS REPLY
60	(3C)	SIGNED-FWORD	4	SLXXLSMO	OFFSET TO LSM SECTION, FROM START OF REPLY, OR 0 IF REPLY DOESN'T CONTAIN ANY LSM ELEMENTS.
64	(40)	SIGNED-FWORD	4	SLXXLSML	LENGTH OF AN LSM ELEMENT.

MESSAGE TEXT SECTION DIRECTORY

WARNING: THIS DIRECTORY DOES NOT EXIST WHEN THE VALUE IN THE REPLY VERSION NUMBER
FIELD, SLXVERS, IS LESS THAN 2.

DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
68	(44)	SIGNED-FWORD	4	SLXXMSGN	NUMBER OF MESSAGE ELEMENTS PRESENT IN THIS REPLY.
72	(48)	SIGNED-FWORD	4	SLXXMSGO	OFFSET TO MESSAGE SECTION, FROM START OF REPLY, OR 0 IF REPLY DOESN'T CONTAIN ANY MESSAGE ELEMENTS.
76	(4C)	SIGNED-FWORD	4	SLXXMSGI	LENGTH OF A MESSAGE ELEMENT

QCAP INFORMATION SECTION DIRECTORY

DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
80	(50)	SIGNED-FWORD	4	SLXXCAPN	NUMBER OF CAP ELEMENTS PRESENT IN THIS REPLY.
84	(54)	SIGNED-FWORD	4	SLXXCAPO	OFFSET TO CAP SECTION FROM START OF REPLY.
88	(58)	SIGNED-FWORD	4	SLXXCAPL	LENGTH OF A CAP ELEMENT. CONFIGURATION SUMMARY ELEMENT THIS ELEMENT APPEARS IN THE REPLY TO A QCONFIG REQUEST AND SUPPLIES SUMMARY INFORMATION ABOUT THE LIBRARY AND ABOUT REPLY LENGTHS NECESSARY FOR OTHER TYPES OF REQUESTS.
0	(0)	STRUCTURE		SLXSCFG	CONFIGURATION SUMMARY ELEMENT.
0	(0)	CHARACTER	3	SLXLID	ELEMENT IDENTIFIER.
3	(3)	HEXSTRING	1	-RESERVED-	RESERVED.
4	(4)	CHARACTER	8	SLXLHNAM	HOST NAME.
12	(C)	SIGNED-FWORD	4	SLXLHHBT	HOST PULSE VALUE.
16	(10)	SIGNED-FWORD	4	SLXLRSTM	RESERVE TIMEOUT LIMIT.
20	(14)	CHARACTER	8	SLXLQNAM	ENQ MAJOR NAME.
28	(1C)	CHARACTER	8	SLXLEJPS	EJECT COMMAND PASSWORD (ENCRYPTED)
36	(24)	CHARACTER	1	SLXLCMPF	COMMAND PREFIX CHARACTER.
37	(25)	A-ADDR	1	SLXLSCLB	LIBRARY DEFAULT SCRATCH LABEL TYPE CODE:
1	(01)	CONST		SLXLLBSL	STANDARD (SL).
2	(02)	CONST		SLXLLBAL	ASCII (AL).
3	(03)	CONST		SLXLLBNL	NON-LABELED (NL).
4	(04)	CONST		SLXLLBNS	NON-STANDARD LABEL (NSL)
38	(26)	A-ADDR	1	SLXLSMF	SMF RECORD TYPE
39	(27)	HEXSTRING	1	-RESERVED-	RESERVED.
40	(28)	SIGNED-FWORD	4	SLXQMDR	LARGEST NUMBER OF DRIVES IN ANY ACS.
44	(2C)	SIGNED-FWORD	4	SLXQDRCT	NUMBER OF DRIVES IN THE LIBRARY.

48	(30)	SIGNED-FWORD	4	SLXQACNT	NUMBER OF ACSS IN THE LIBRARY.
52	(34)	SIGNED-FWORD	4	SLXQLCNT	NUMBER OF LSMS IN THE LIBRARY.
56	(38)	SIGNED-FWORD	4	SLXZVOL	MAXIMUM LENGTH OF REPLY DATA RETURNED IN RESPONSE TO A QVOLUME REQUEST THAT SPECIFIES THE LARGEST SUPPORTED NUMBER OF VOLUMES (500).
60	(3C)	SIGNED-FWORD	4	SLXZQDRV	MAXIMUM LENGTH OF REPLY DATA RETURNED IN RESPONSE TO A QDRIVES REQUEST.
64	(40)	SIGNED-FWORD	4	SLXZQDRL	MAXIMUM LENGTH OF REPLY DATA RETURNED IN RESPONSE TO A QDRLIST REQUEST.
68	(44)	SIGNED-FWORD	4	SLXQVOL	MAXIMUM LENGTH OF REPLY DATA RETURNED IN RESPONSE TO A QVOLUME REQUEST THAT SPECIFIES ONLY 1 VOLUME.
72	(48)	SIGNED-FWORD	4	SLXZGSCR	MAXIMUM LENGTH OF REPLY DATA RETURNED IN RESPONSE TO A SELSCR REQUEST.
76	(4C)	SIGNED-FWORD	4	SLXZMDM	MAXIMUM LENGTH OF REPLY DATA RETURNED IN RESPONSE TO A MOUNT REQUEST.
80	(50)	SIGNED-FWORD	4	SLXZQSCR	MAXIMUM LENGTH OF REPLY DATA RETURNED IN RESPONSE TO A QSCRATCH REQUEST.
84	(54)	SIGNED-FWORD	4	SLXZDISM	MAXIMUM LENGTH OF REPLY DATA RETURNED IN RESPONSE TO A DISMOUNT REQUEST.
88	(58)	SIGNED-FWORD	4	SLXZEJCT	MAXIMUM LENGTH OF REPLY DATA RETURNED IN RESPONSE TO AN EJECT REQUEST THAT SPECIFIES THE LARGEST SUPPORTED NUMBER OF VOLUMES (500).
92	(5C)	SIGNED-FWORD	4	SLXZSCR	MAXIMUM LENGTH OF REPLY DATA RETURNED IN RESPONSE TO A SCRATCH REQUEST.

WARNING: THE REMAINING FIELDS OF THIS ELEMENT ARE AVAILABLE FOR VERSION(S) 3 AND ABOVE.

DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
108	(6C)	SIGNED-FWORD	4	SLXZMOVE	MAXIMUM LENGTH OF REPLY DATA RETURNED IN RESPONSE TO AN MOVE REQUEST.
112	(70)	SIGNED-FWORD	4	SLXZEJC1	MAXIMUM LENGTH OF REPLY DATA RETURNED IN RESPONSE TO AN EJECT REQUEST FOR ONLY 1 VOLUME.

WARNING: THE REMAINING FIELDS OF THIS ELEMENT ARE AVAILABLE FOR VERSION(S) 6 AND ABOVE.

DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
116	(74)	A-ADDR	4	SLXQUCSA	MVS -- ADDRESS OF SLSUXCSA.
120	(78)	SIGNED-FWORD	4	-RESERVED-	RESERVED.
124	(7C)	SIGNED-FWORD	4	SLXQLCAP	NUMBER OF CAPS IN LIBRARY.
128	(80)	SIGNED-FWORD	4	SLXEXLM0	ExLM R15
132	(84)	SIGNED-FWORD	4	SLXEXLM1	ExLM R1
136	(88)	SIGNED-FWORD	4	SLXEXLM2	ExLM R2

140	(8C)	SIGNED-FWORD	4	SLXZQCAP	MAXIMUM LENGTH OF REPLY DATA RETURNED IN RESPONSE TO A QCAP REQUEST.
156	(9C)	SIGNED-FWORD	4	-RESERVED-	RESERVED FUTURE USE.
160	(A0)	SIGNED-HWORD	2	SLXHSCV	HSC VERSION NUMBER
162	(A2)	HEXSTRING	6	-RESERVED-	
168	(A8)	CONST		SLXSCFGL	LENGTH OF A CONFIGURATION ELEMENT.

QDSN INFORMATION ELEMENT

THIS ELEMENT APPEARS IN THE REPLY TO A QDSN REQUEST AND SUPPLIES SUMMARY INFORMATION ABOUT THE CURRENT REFERENCED DATASETS USED BY THE HSC.

DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
0	(0)	STRUCTURE		SLXDSNIM	DATASET INFORMATION MAP.
0	(0)	CHARACTER	3	SLXQDID	SECTION IDENTIFIER.
3	(03)	BITSTRING	1	SLXDSFLG	DATASET TYPE.
1	(01)	CONST		SLXDSPRM	CDS PRIMARY.
2	(02)	CONST		SLXDSSEC	CDS SECONDARY.
3	(03)	CONST		SLXDSSBY	CDS STANDBY.
4	(04)	CONST		SLXDSVAT	VOLUME ATTRIBUTES.
5	(05)	CONST		SLXDSUAT	UNIT ATTRIBUTES.
6	(06)	CONST		SLXDSTRQ	TAPEREQS.
7	(07)	CONST		SLXDSPLB	PARMLIB.
8	(08)	CONST		SLXDSJNP	PRIMARY JOURNAL.
9	(09)	CONST		SLXDSJNA	ALTERNATE JOURNAL.
9	(09)	CONST		SLXDSDMAX	MAX NUMBER OF QDS RETURNED.
4	(04)	CHARACTER	44	SLXDSNAM	DATASET NAME.
48	(30)	CHARACTER	8	SLXDMSMBR	MEMBER NAME.
56	(38)	CHARACTER	6	SLXDVSOL	VOLUME NAME.
62	(3E)	CHARACTER	8	SLXDSUNT	UNIT NAME.
70	(46)	CHARACTER	2	-RESERVED-	RESERVED.
72	(48)	CHARACTER	96	-RESERVED-	RESERVED.
168	(A8)	AREA	8	-RESERVED-	ALIGN
168	(A8)	LENGTH		SLXDSNEL	LENGTH OF ONE DATASET ENTRY.

CAP INFORMATION ELEMENT

THIS ELEMENT SUPPLIES INFORMATION ABOUT A SINGLE LIBRARY CAP

DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
0	(0)	STRUCTURE		SLXSCAP	
0	(0)	CHARACTER	4	SLXCID	SECTION IDENTIFIER.
4	(4)	HEXSTRING	1	SLXCACS	ACS ADDRESS.
5	(5)	HEXSTRING	1	SLXCLSM	LSM ADDRESS.
6	(6)	HEXSTRING	1	SLXCCAP	CAP NUMBER.
7	(7)	HEXSTRING	1	-RESERVED-	RESERVED.
8	(8)	AREA	2	SLXCSTAT	CAP STATUS.
8	(8)	BITSTRING	1	SLXCSTB1	CAP STATUS.
		1... X'80'		SLXCSTCA	CAP IS ACTIVE.
		.1.. X'40'		SLXCSTNR	CAP NEEDS RECOVERY.
		..1. X'20'		SLXCSTAM	CAP IS IN AUTOMATIC MODE.
		...1 X'10'		SLXCSTCL	CAP IS LINKED.
	 1... X'08'		SLXCSTCO	CAP IS ONLINE.
9	(9)	BITSTRING	1	SLXCSTB2	CAP MODE.
		1... X'80'		SLXCSTIE	CAP IS ENTERING.
		.1.. X'40'		SLXCSTID	CAP IS DRAINING.
		..1. X'20'		SLXCSTIJ	CAP IS EJECTING.
		...1 X'10'		SLXCSTIC	CAP IS CLEANING.
	 1... X'08'		SLXCSTII	CAP IS IDLE .
10	(A)	BITSTRING	1	SLXTYPE	TYPE OF CAP.
		1... X'80'		SLXCTPC	PRIORITY CAP
	1 X'01'		SLXCTCIM	CIMARRON

	1. X'02'		SLXCTCLP	CLIPPER.
	11 X'03'		SLXCTTWS	STANDARD CLIPPER
	1.. X'04'		SLXCTTWO	OPTIONAL CLIPPER
	1.1 X'05'		SLXCTTIM	(9740/TimberWolf)
12	(C)	SIGNED-HWORD	2	SLXCCELL	CELLS IN CAP.
14	(E)	HEXSTRING	1	SLXCNRW	ROWS.
15	(F)	HEXSTRING	1	SLXCNCOL	COLUMNS.
16	(10)	HEXSTRING	1	SLXCCMAG	MAGAZINES.
17	(11)	HEXSTRING	1	SLXCCMGC	CELLS IN MAGAZINE.
DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
18	(12)	CHARACTER	8	SLXCJOB	JOBNAME OF OWNER.
26	(1A)	HEXSTRING	6	-RESERVED-	ALIGN TO DOUBLE WORD
32	(20)	CONST		SLXSCAPL	LENGTH OF A CAP ELEMENT.

VOLUME INFORMATION ELEMENT

THIS ELEMENT SUPPLIES INFORMATION ABOUT A SINGLE VOLUME AND IS REPEATED FOR EACH VOLUME ASSOCIATED WITH A REQUEST. THIS ELEMENT MAY APPEAR IN REPLIES TO THE FOLLOWING REQUESTS:

QDRLIST - RETURN DRIVE INFORMATION, ORDERED BY PREFERENCE

QVOLUME - RETURN VOLUME INFORMATION

SELSCR - SELECT A SCRATCH VOLUME

EJECT - EJECT VOLUMES

DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
0	(0)	STRUCTURE		SLXSVOL	VOLUME INFORMATION ELEMENT.
0	(0)	CHARACTER	3	SLXVID	ELEMENT IDENTIFIER.
3	(3)	BITSTRING	1	SLXVSTA	VOLUME STATUS:
		1... X'80'		SLXVILB	VOLUME IS IN LIBRARY
		. .1.. X'40'		SLXVOHST	VOLUME IS IN USE BY ANOTHER HOST
		. ..1. X'20'		SLXVSCR	VOLUME IS CONSIDERED SCRATCH
		. ..1. X'10'		SLXVMAL	VOLUME IS IN MANUAL-MODE LSM.
	 1... X'08'		SLXVDSC	VOLUME IS IN DISCONNECTED ACS.
	1.. X'04'		SLXVMNT	VOLUME IS MOUNTED ON A DRIVE.
	1. X'02'		SLXVERR	VOLUME IS 'ERRANT' (I.E., ITS LOCATION WITHIN THE LIBRARY IS UNCERTAIN).
	1 X'01'		SLXVTV	VOLUME IS A VTCS VIRTUAL VOLUME
4	(4)	CHARACTER	6	SLXVSER	VOLUME SERIAL.
10	(A)	A-ADDR	1	SLXVLC	VOLUME LOCATION CODE:
0	(0)	CONST		SLXVUNK	LOCATION DATA UNAVAILABLE (SLXVLOC IS 0).
1	(1)	CONST		SLXVCEL	LOCATION DATA DESCRIBES A CELL.
2	(02)	CONST		SLXVDRV	LOCATION DATA DESCRIBES A DRIVE.
11	(B)	AREA	5	SLXVLOC	VOLUME LOCATION DATA:
11	(B)	A-ADDR	1	SLXVACS	ACS ID.
12	(C)	A-ADDR	1	SLXVLSM	LSM ID.
13	(D)	A-ADDR	3	SLXVPNL	CELL'S PANEL ID, ROW ID, COLUMN ID.
13	(D)	A-ADDR	2	SLXVDRIV	DRIVE ADDRESS (OCUU).
15	(F)	BITSTRING	1	SLXVSTA2	MORE VOLUME STATUS: EQU X'E0' RESERVED.
		...1 X'10'		SLXVLMU	VOLUME MEDIA TYPE CAME FROM LMU. EQU X'08' RESERVED.
	1.. X'04'		SLXVMUNR	VOLUME MEDIA TYPE UNREADABLE.
	1. X'02'		SLXVMVLA	VOLUME MEDIA TYPE CAME FROM VOLATTR.

	 1 X'01'		SLXVMDFL	VOLUME MEDIA TYPE DEFAULT ASSIGNED.
16	(10)	AREA	8	SLXVTSSN	VTSS NAME
16	(10)	SIGNED-FWORD	4	SLXVDATI	HI-WORD OF TOD AT INSERTION.
20	(14)	SIGNED-FWORD	4	SLXVDATL	HI-WORD OF TOD LAST SELECTION.
24	(18)	SIGNED-FWORD	4	SLXVSCNT	SELECTION COUNT.
28	(1C)	SIGNED-FWORD	4	SLXVDATD	HI-WORD OF TOD LAST MOUNT.
32	(20)	CHARACTER	8	SLXVMED	TYPE OF MEDIA.
DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
40	(28)	CONST		SLXSVOLN	LENGTH OF A VOLUME ELEMENT.
40	(28)	CONST		SLXSROLL	LENGTH OF A VOLUME ELEMENT.

DRIVE INFORMATION ELEMENT

THIS ELEMENT SUPPLIES INFORMATION ABOUT A SINGLE LIBRARY TAPE DRIVE AND IS REPEATED FOR EACH DRIVE ASSOCIATED WITH A REQUEST. THIS ELEMENT MAY APPEAR IN REPLIES TO THE FOLLOWING REQUESTS:

QDRIVES - RETURN DRIVE AND LSM INFORMATION

QDRLIST - RETURN DRIVE INFORMATION, ORDEREDBY PREFERENCE

DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
0	(0)	STRUCTURE		SLXSDRV	DRIVE INFORMATION ELEMENT.
0	(0)	CHARACTER	3	SLXDID	ELEMENT IDENTIFIER
3	(3)	BITSTRING	1	SLXDSTA	LIBRARY STATUS:
		...1 X'10'		SLXDMANU	LSM IS IN MANUAL MODE.
	 1... X'08'		SLXDDISC	ACS IS DISCONNECTED.
4	(4)	A-ADDR	1	SLXQDEAC	ACS ID.
5	(5)	A-ADDR	1	SLXQDELS	LSM ID.
6	(6)	A-ADDR	2	SLXQDECU	DRIVE ADDRESS (0CUU).
8	(8)	CHARACTER	8	SLXQDRT	RECORDING TECHNIQUE OF DRIVE
16	(10)	CONST		SLXSDRVL	LENGTH OF A DRIVE ELEMENT.

LSM INFORMATION ELEMENT

THIS ELEMENT SUPPLIES INFORMATION ABOUT A SINGLE LSM (LIBRARY STORAGE MODULE) AND IS REPEATED FOR EACH LSM ASSOCIATED WITH A REQUEST. THIS ELEMENT MAY APPEAR IN REPLIES TO THE FOLLOWING REQUESTS:

QDRIVES - RETURN DRIVE AND LSM INFORMATION

QSCRATCH - RETURN LSM INFORMATION, ORDERED BY PREFERENCE

DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
0	(0)	STRUCTURE		SLXSLSM	LSM INFORMATION ELEMENT
0	(0)	CHARACTER	3	SLXMID	ELEMENT IDENTIFIER.
3	(3)	BITSTRING	1	SLXMSTAT	LIBRARY STATUS:
		...1 X'10'		SLXMANUL	LSM IS IN MANUAL MODE
	 1... X'08'		SLXMDISC	ACS IS DISCONNECTED.
4	(4)	A-ADDR	1	SLXMACS	ACS ID.
5	(5)	A-ADDR	1	SLXMLSM	LSM ID.
6	(6)	SIGNED-FWORD	1	SLXMADJN	NUMBER OF ADJACENT LSMS.
7	(7)	A-ADDR	1	SLXMADJ(4)	LIST OF LSM IDS OF ADJACENT LSMS (ONLY THE FIRST N IDS ARE VALID, WHERE N IS THE VALUE IN SLXMADJN).
11	(B)	HEXSTRING	1	-RESERVED-	RESERVED.
12	(C)	SIGNED-FWORD	4	SLXMNSCR	NUMBER OF SCRATCH VOLUMES IN THIS LSM.
16	(10)	SIGNED-FWORD	4	SLXMTCEL	TOTAL CELLS IN LSM.
20	(14)	SIGNED-FWORD	4	SLXMFCEL	FREE CELLS IN LSM.
24	(18)	CONST		SLXSLSML	LENGTH OF AN LSM ELEMENT.

MESSAGE TEXT ELEMENT

THIS ELEMENT SUPPLIES THE COMPLETE TEXT OF THE MESSAGE WHOSE BINARY MESSAGE ID NUMBER IS REPORTED IN HEADER FIELD SLXSRC. THIS ELEMENT MAY APPEAR IN REPLIES TO

THE FOLLOWING REQUESTS:

DISMOUNT - DISMOUNT A VOLUME

MOUNT - MOUNT A VOLUME

MOVE - MOVE A VOLUME

EJECT - EJECT VOLUMES

DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
0	(0)	STRUCTURE		SLXSMG	MESSAGE TEXT ELEMENT.
0	(0)	CHARACTER	3	SLXGID	ELEMENT IDENTIFIER.
3	(3)	CHARACTER	125	SLXGTEXT	TEXT OF THE MESSAGE WHOSE NUMBER IS IN FIELD SLXSRC OF THE REPLY HEADER.
0	(0)	STRUCTURE		SLXSMG	
128	(80)	CONST		SLXSMGSL	LENGTH OF A MESSAGE ELEMENT.

Cross-Reference

The following table provides a cross-reference for the SLX macro:

Table 9–2 SLX Macro Cross-Reference

Name	Len	Offset Value
SLXCACS	000001	04
SLXCCAP	000001	06
SLXCCELL	000002	0C
SLXCCMAG	000001	10
SLXCCMGC	000001	11
SLXCID	000004	00
SLXCJOB	000008	12
SLXCLSM	000001	05
SLXCMDRC	000001	03
SLXCNCOL	000001	0F
SLXCNROW	000001	0E
SLXCRLN	000004	0C
SLXCSTAM	NA	20
SLXCSTAT	000002	08
SLXCSTB1	000001	08
SLXCSTB2	000001	09
SLXCSTCA	NA	80
SLXCSTCL	NA	10
SLXCSTCO	NA	08
SLXCSTIC	NA	10
SLXCSTID	NA	40
SLXCSTIE	NA	80
SLXCSTII	NA	08
SLXCSTIJ	NA	20
SLXCSTNR	NA	40

Table 9–2 (Cont.) SLX Macro Cross-Reference

Name	Len	Offset Value
SLXCTCIM	NA	01
SLXCTCLP	NA	02
SLXCTPC	NA	80
SLXCTTIM	NA	05
SLXCTTWO	NA	04
SLXCTTWS	NA	03
SLXCTYPE	000001	0A
SLXDDISC	NA	08
SLXDID	000003	00
SLXDMANU	NA	10
SLXDSFLG	000001	03
SLXDSJNA	NA	09
SLXDSJNP	NA	08
SLXDSDMAX	NA	09
SLXDSDMBR	000008	30
SLXDSDNAM	000044	04
SLXDSDNEL	NA	A8
SLXDSPLB	NA	07
SLXDSPRM	NA	01
SLXDSSBY	NA	03
SLXDSSEC	NA	02
SLXDSTA	000001	03
SLXDSTRQ	NA	06
SLXDSUAT	NA	05
SLXDSUNT	000008	3E
SLXDsvAT	NA	04
SLXDsvOL	000006	38
SLXDvMM	NA	8051
SLXEND	000008	B0
SLXEXLM0	000004	80
SLXEXLM1	000004	84
SLXEXLM2	000004	88
SLXGID	000003	00
SLXGTEXT	000125	03
SLXHID	000003	00
SLXHL	NA	14
SLXHSCV	000002	A0

Table 9–2 (Cont.) SLX Macro Cross-Reference

Name	Len	Offset Value
SLXL	NA	B0
SLXLCMPF	000001	24
SLXLEJPS	000008	1C
SLXLHHBT	000004	0C
SLXLHNAM	000008	04
SLXLID	000003	00
SLXLBAL	NA	02
SLXLLBNL	NA	03
SLXLLBNS	NA	04
SLXLLBSL	NA	01
SLXLOCKD	NA	20
SLXLQNAM	000008	14
SLXLRSTM	000004	10
SLXLSCLB	000001	25
SLXLSMF	000001	26
SLXMACS	000001	04
SLXMADJI	000001	07
SLXMADJN	000001	06
SLXMANUL	NA	10
SLXMDISC	NA	08
SLXMFCEL	000004	14
SLXMID	000003	00
SLXMLSM	000001	05
SLXMNSCR	000004	0C
SLXMSTAT	000001	03
SLXMSTT	NA	803A
SLXMTCEL	000004	10
SLXNORSP	NA	28
SLXNTCB	NA	1C
SLXPEOFF	000004	10
SLXQACNT	000004	30
SLXQDEAC	000001	04
SLXQDECU	000002	06
SLXQDELS	000001	05
SLXQDID	000003	00
SLXQDRCT	000004	2C
SLXQDRT	000008	08

Table 9–2 (Cont.) SLX Macro Cross-Reference

Name	Len	Offset Value
SLXQID	000004	00
SLXQJTC	000004	08
SLXQJTD	NA	00
SLXQJTL	NA	18
SLXQJTN	NA	FFFF
SLXQJTS	000002	0C
SLXQJTT	000004	04
SLXQJTV	000006	0E
SLXQLCAP	000004	7C
SLXQLCNT	000004	34
SLXQMDR	000004	28
SLXQUCSA	000004	74
SLXRBADL	NA	2C
SLXRBADP	NA	08
SLXRBTOK	NA	3C
SLXREOV	NA	34
SLXRFAIL	NA	10
SLXRIERR	NA	0C
SLXRNAUT	NA	18
SLXRNHSC	NA	14
SLXRNVC I	NA	30
SLXROK	NA	00
SLXRPLY	000001	00
SLXRVNV	NA	38
SLXRWARN	NA	04
SLXSANF	NA	8036
SLXSCAPL	NA	20
SLXSCFGL	NA	A8
SLXSDRVL	NA	10
SLXSFUL	NA	8050
SLXSID	000004	00
SLXSLSML	NA	18
SLXSMSG L	NA	80
SLXSRC	000004	08
SLXSTPE	NA	EE
SLXSTPK	NA	00
SLXSTPL	NA	10

Table 9–2 (Cont.) SLX Macro Cross-Reference

Name	Len	Offset Value
SLXSTPN	NA	FF
SLXSTPS	000001	09
SLXSTPT	000004	04
SLXSTPY	000001	08
SLXSVOLL	NA	28
SLXSVOLN	NA	28
SLXTINTR	NA	8032
SLXTPROT	NA	24
SLXTRNF	NA	8040
SLXVACS	000001	0B
SLXVCEL	NA	01
SLXVCODE	NA	07
SLXVDATD	000004	1C
SLXVDATI	000004	10
SLXVDATL	000004	14
SLXVDRIV	000002	0D
SLXVDRV	NA	02
SLXVDSC	NA	08
SLXVERR	NA	02
SLXVERS	000001	04
SLXVID	000003	00
SLXVILB	NA	80
SLXVLC	000001	0A
SLXVLOC	000005	0B
SLXVLSM	000001	0C
SLXVMAL	NA	10
SLXVMDFL	NA	01
SLXVMED	000008	20
SLXVMLMU	NA	10
SLXVMNT	NA	04
SLXVMUNR	NA	04
SLXVMVLA	NA	02
SLXVOHST	NA	40
SLXVPNL	000003	0D
SLXVSCNT	000004	18
SLXVSCR	NA	20
SLXVSER	000006	04

Table 9–2 (Cont.) SLX Macro Cross-Reference

Name	Len	Offset Value
SLXVSTA	000001	03
SLXVSTA2	000001	0F
SLXVTSSN	000008	10
SLXVTV	NA	01
SLXVUNK	NA	00
SLXXCAPL	000004	58
SLXXCAPN	000004	50
SLXXCAPO	000004	54
SLXXCFGL	000004	1C
SLXXCFGN	000004	14
SLXXCFG0	000004	18
SLXXDRV1	000004	34
SLXXDRVN	000004	2C
SLXXDRVO	000004	30
SLXXLSML	000004	40
SLXXLSMN	000004	38
SLXXLSMO	000004	3C
SLXXMSG1	000004	4C
SLXXMSGN	000004	44
SLXXMSG0	000004	48
SLXXQDSL	000004	7C
SLXXQDSN	000004	74
SLXXQDS0	000004	78
SLXXQJTL	000004	64
SLXXQJTN	000004	5C
SLXXQJTO	000004	60
SLXXSDL	NA	B0
SLXXSTPL	000004	70
SLXXSTPN	000004	68
SLXXSTPO	000004	6C
SLXXVCIL	000004	88
SLXXVCIN	000004	80
SLXXVCIO	000004	84
SLXXVOLL	000004	28
SLXXVOLN	000004	20
SLXXVOLO	000004	24
SLXZDEFP	000004	60

Table 9–2 (Cont.) SLX Macro Cross-Reference

Name	Len	Offset Value
SLXZDEFS	000004	64
SLXZDISM	000004	54
SLXZEJCT	000004	58
SLXZEJC1	000004	70
SLXZGSCR	000004	48
SLXZMDM	000004	4C
SLXZMOVE	000004	6C
SLXZQCAP	000004	8C
SLXZQDRL	000004	40
SLXZQDRV	000004	3C
SLXZQDSN	000004	98
SLXZQEJT	000004	90
SLXZQSCR	000004	50
SLXZQVOL	000004	44
SLXZSCR	000004	5C
SLXZSETO	000004	68
SLXZSTOP	000004	94
SLXZVOL	000004	38

ACSINT Request DSECT

The ACSINT enables information to be passed between the TMS and the ACS service machine. It is built by the ACSRQ macro when DSECT=YES is specified and is sent using IUCV.

Note: The following DSECT includes some parameters that are not supported by the VM Client. They are included for completeness and compatibility only.

The following example shows output for the ACSINT record format:

Example 9–5 ACSINT Record Format

```

ACSINT - TMS INTERFACE REQUEST PARAMETER LIST
FUNCTION: THIS DEFINES THE DATA PASSED TO THE TMS VIA IUCV IT DEFINES THE VARIOUS
FUNCTIONS THAT CAN BE REQUESTED AND THE STRUCTURE PASSED.
DEC  HEX      TYPE      LENGTH  LABEL      DESCRIPTION
0      (0)      STRUCTURE      ACSIHDR  TMS INTERFACE PARAMETER LIST:
0      (0)      CHARACTER      4      PARAMETER LIST IDENTIFIER.
'ACSI' (C1C3E2C9) CHAR CONST  ACSIID
4      (4)      SIGNED-FWORD  4      ACSILEN  PARAMETER LIST LENGTH.
8      (8)      A-ADDR      1      ACSIVER  PARAMETER LIST VERSION
NUMBER.
7      (07)      CONST      ACSIVN  CURRENT VERSION.
9      (9)      A-ADDR      1      ACSIRT  FUNCTION CODE:
0      (00)      CONST      ACSINOP NOOP - NO OPERATION.

```

1	(01)	CONST	ASCIRS01	RESERVED - MVS ONLY.
2	(02)	CONST	ASCIRS02	RESERVED - MVS ONLY.
3	(03)	CONST	ASCIRS03	RESERVED - MVS ONLY.
4	(04)	CONST	ASCISTOP	STOP - STOP AN INTERRUPTABLE PGMI TASK.
5	(05)	CONST	ACSIISOPR	SETOPER - SET OPERATOR INTERACTION MODE.
20	(14)	CONST	ACSIQCNF	QCONFIG - RETURN CONFIGURATION SUMMARY.
21	(15)	CONST	ACSIQDRV	QDRIVES - RETURN DRIVE AND LSM INFO.
22	(16)	CONST	ACSIQDRL	QDRLIST - RETURN DRIVE INFORMATION, X .
23	(17)	CONST	ACSIQSCR	QSCRATCH - RETURN SCRATCH COUNT INFO.
24	(18)	CONST	ACSIQVOL	QVOLUME - RETURN VOLUME INFORMATION.
25	(19)	CONST	ACSIQEJT	QEJECT - RETURN EJECT STATUS.
26	(1A)	CONST	ACSIQCAP	QCAP - QUERY CAP STATUS.
27	(1B)	CONST	ACSIQVLU	QVOLUME - RETURN MOUNTED VOLUMES.
28	(1C)	CONST	ACSIQRQS	QREQUEST - RETURN PENDING REQUESTS.
29	(1D)	CONST	ACSIQDSN	QDSN - QUERY DATASET.
40	(29)	CONST	ACSIMNT	MOUNT - MOUNT VOLUME.
41	(29)	CONST	ACSIMOVE	MOVE - MOVE A CARTRIDGE TO AN LSMID X.
42	(2A)	CONST	ACSIDSM	DISMOUNT - DISMOUNT VOLUME.
43	(2B)	CONST	ACSIJCT	EJECT - EJECT A VOL FROM THE LIBRARY.
60	(3C)	CONST	ACSISSCR	SELSCR - SELECT A SCRATCH VOLUME.
61	(3D)	CONST	ACSISCRA	SCRATCH - CHANGE VOL STATUS TO 'SCRATCH'.
62	(3E)	CONST	ACSIUNSC	UNSCRATCH- CHANGE VOLUME STATUS TO NOT X.
63	(3F)	CONST	ACSIDSCR	DEFSCR - SPECIFY NO. OF SCRATCH POOLS.
64	(40)	CONST	ACSIDPOL	DEFPOL - SPECIFY A SCRATCH POOL'S VOLSER RANGE.
10	(A)	BITSTRING 1	ACSIFLG1	FLAG BYTE 1: (PGMI CONTROL 1) 1
		1... X'80'	ACSIF180	RESERVED MVS OPTION=SYNC
		.1.. X'40'	ACSIVUSR	USER= SPECIFIED; ACSIUUSER CONTAINS NAME
		..1. X'20'	ACSIMANO	DIALOG=OFF SPECIFIED.
		...1 X'10'	ACSIWTOR	DIALOG=ON SPECIFIED.
	 1... X'08'	ACSINDEL	NOTIFY=INSDDEL SPECIFIED.
	1.. X'04'	ACSINNDL	NOTIFY=NOINSDDEL SPECIFIED.
	1. X'02'	ACSIACC1	ACCT1= SPECIFIED ACCOUNTING TOKEN.
	1 X'01'	ACSIACC2	ACCT2= SPECIFIED ACCOUNTING TOKEN.
11	(B)	HEXSTRING 1	ACSIFLG2	FLAG BYTE 2: (PGMI CONTROL 2).
12	(C)	BITSTRING 1	ACSIFLG3	FLAG BYTE 3: (MOVEMENT CONTROL 1).
		1... X'80'	ACSIHST	ACSIHOST CONTAINS HOST_ID.
		.1.. X'40'	ACSIVACS	ACSIACS CONTAINS AN ACSID.
		..1. X'20'	ACSIVLSM	ACSILSMI CONTAINS LSMID.

		...1 X'10'		ACSIVCAP	ACSICAP CONTAINS CAP_ID.
	 1... X'08'		ACSIF308	RESERVED FUTURE USE.
	1.. X'04'		ACSIVVOL	ACSIVOLS CONTAINS A VOLSER.
	1. X'02'		ACSIVLST	ACSILOFF CONTAINS LIST OFFSET.
13	(D)1 X'01'		ACSIVCNT	ACSICNT CONTAINS LIST COUNT.
		BITSTRING 1		ACSIFLG4	FLAG BYTE 4: (MOVEMENT CONTROL 2).
		1... X'80'		ACSIVTLM	ACSITLSM CONTAINS TO LSM_ID.
		.1.. X'40'		ACSIVTPN	ACSITPAN CONTAINS TO PANEL.
		..1. X'20'		ACSIF420	RESERVED FUTURE USE.
		...1 X'10'		ACSIF410	RESERVED FUTURE USE.
	 1... X'08'		ACSIVPAN	ACSIPAN CONTAINS PANEL NUMBER
	1.. X'04'		ACSIVROW	ACSIROW CONTAINS ROW NUMBER.
	1. X'02'		ACSIVCOL	ACSICOL CONTAINS COLUMN NUMBER.
	1 X'01'		ACSIVDRV	ACSIDRIV CONTAINS DRIVE DEVICE NUMBER.
14	(E)	HEXSTRING 1		ACSIFLG5	FLAG BYTE 5: (MOVEMENT CONTROL 3).
15	(F)	BITSTRING 1		ACSIFLG6	FLAG BYTE 6: (MISCELLANOUS CONTROL 1).
		1... X'80'		ACSIPROT	PROTECT=YES SPECIFIED.
		.1.. X'40'		ACSISCR	SCRATCH=YES SPECIFIED.
		..1. X'20'		ACSIVSCP	ACSIPOOL CONTAINS SCRATCH POOL NUMBER.
		...1 X'10'		ACSISUBN	ACSISUBP CONTAINS SUBPOOL NAME.
	 1... X'08'		ACSIVTKN	ACSITKNO CONTAINS TOKEN NUMBER.
	1.. X'04'		ACSIVTXT	ACSITEXT CONTAINS TEXT STRING.
	1. X'02'		ACSIF602	RESERVED FUTURE USE.
	1 X'01'		ACSIF601	RESERVED FUTURE USE.
16	(10)	HEXSTRING 1		ACSIFLG7	FLAG BYTE 7: (MISCELLANOUS CONTROL 2).
17	(11)	A-ADDR 1		ACSILABT	SCRATCH LABEL TYPE LTYPE= PARAMETER.
	 X'00'		ACSILLDT	LDT (LIBRARY DEFAULT TYPE).
	1 X'01'		ACSILSL	SL (STANDARD LABEL).
	1. X'02'		ACSILAL	AL (ANSI LABEL).
	11 X'03'		ACSILNL	NL (NON-LABELED).
	1.. X'04'		ACSILNS	NSL (NON-STANDARD LABEL).
18	(12)	CHARACTER 8		ACSIUSER	USER NAME USED TO ASSOCIATE CONSOLE ID.
26	(1A)	CHARACTER 8		ACSIACT1	ACCOUNTING TOKEN 1.
34	(22)	CHARACTER 8		ACSIACT2	ACCOUNTING TOKEN 2.
42	(2A)	HEXSTRING 2		-RESERVED-	RESERVE SLACK BYTES.
44	(2C)	SIGNED-FWORD 4		-RESERVED-	RESERVED MVS.
48	(30)	HEXSTRING 4		ACSITKNO	PASS THROUGH TOKEN NUMBER.
52	(34)	A-ADDR 4		-RESERVED-	RESERVED MVS.
56	(38)	SIGNED-HWORD 2		ACSICNT	COUNT FROM COUNT= PARAMETER.
58	(3A)	SIGNED-HWORD 2		ACSILOFF	OFFSET, FROM START OF PARAMETER LIST, TO START OF THE ELEMENT LIST AREA.
60	(3C)	CHARACTER			CHARACTER 6 ACSIVOLS VOLSER FROM VOLSER= PARAMETER.
66	(42)	CHARACTER 8		ACSIHOST	ASSOCIATED HOST FROM HOSTID= PARAMETER.
74	(4A)	AREA 4		ACSIALC	ACS / LSM / CAP

					IDENTIFICATION.
74	(4A)	HEXSTRING	1	ACSIACS	ACS ID NUMBER (AA).
74	(4A)	HEXSTRING	2	ACSILSMI	LSM ID NUMBER (AA0L).
74	(4A)	HEXSTRING	4	ACSICAP	CAP ID NUMBER (AA0LCC00).
74	(4A)	HEXSTRING	3	-RESERVED-	
77	(4D)	HEXSTRING	1	-RESERVED-	NOT IMPLEMENTED (ALWAYS X'00').
78	(4E)	HEXSTRING	1	ACSIPAN	PANEL FROM PAN= PARAMETER.
79	(4F)	HEXSTRING	1	ACSIROW	ROW FROM ROW= PARAMETER.
80	(50)	HEXSTRING	1	ACSICOL	COLUMN FROM COL= PARAMETER.
81	(51)	HEXSTRING	1	ACSITPAN	TO PANEL FROM TOPAN= PARAMETER.
82	(52)	HEXSTRING	2	ACSITLSM	TO LSM FROM TOLSM= PARAMETER.
84	(54)	A-ADDR	2	ACSIDRIV	DRIVE DEVICE NUMBER FROM DRIVE= PARAMETER.
86	(56)	A-ADDR	1	ACSIPOOL	SCRATCH POOL NUMBER, SCRPOOL= PARAMETER.
87	(57)	CHARACTER	32	ACSITEXT	TEXT ASSOCIATED WITH REQUEST.
119	(77)	CHARACTER	13	ACSISUBP	NAME FROM SUBPOOL= KEYWORD.

TAPEREQ INPUT KEY VALUES.

THE FOLLOWING VALUES ARE USED AS THE KEY IN SEARCHING THE CURRENT TAPEREQ PARAMETER FILE.

DEC	HEX	TYPE	LENGTH	LABEL	DESCRIPTION
132	(84)	BITSTRING	1	ACSIFLG8	FLAG BYTE 8: (TAPEREQ CONTROL 1).
		1... X'80'		ACSIFJOB	ACSIJOBN PRESENT.
		.1... X'40'		ACSIFSTP	ACSISTEP PRESENT.
		..1. X'20'		ACSIFPGM	ACSIPGMN PRESENT.
		...1 X'10'		ACSIFDSN	ACSIDSN PRESENT.
	 1... X'08'		ACSIFEXP	ACSIEXPD PRESENT.
	1.. X'04'		ACSIFRET	ACSIRETP PRESENT.
	1. X'02'		ACSIFVOL	ACSIVOLT PRESENT.
133	(85)	OFFSET		ACSITRI	TAPEREQ INPUT VALUES.
133	(85)	CHARACTER	8	ACSIJOBN	STRING TO MATCH TAPEREQ JOBNAME VALUE.
141	(8D)	CHARACTER	8	ACSISTEP	STRING TO MATCH TAPEREQ STEPNAME VALUE.
149	(95)	CHARACTER	8	ACSIPGMN	STRING TO MATCH TAPEREQ PROGNAME VALUE.
157	(9D)	CHARACTER	44	ACSIDSN	STRING TO MATCH TAPEREQ DSN VALUE.
201	(C0)	HEXSTRING	3	ACSIEXPD	VALUE TO MATCH TAPEREQ EXPDTP VALUE
204	(CC)	HEXSTRING	2	ACSIRETP	VALUE TO MATCH TAPEREQ RETPD VALUE.
206	(CE)	CHARACTER	1	ACSIVOLT	STRING TO MATCH TAPEREQ VOLTYPE VALUE.
74	(4A)	LENGTH		ACSITRIL	TAPEREQ OUTPUT VALUES.

THE FOLLOWING VALUES ARE USED AS OVERRIDE (OR SPECIFIC) VALUES TO THE VALUES FOUND IN THE CURRENT TAPEREQ PARAMETER FILE.

207	(CF)	BITSTRING	1	ACSIFLG9	FLAG BYTE 9: (TAPEREQ CONTROL 2).
		1... X'80'		ACSIFREC	ACSIRECT PRESENT.
		.1... X'40'		ACSIFMED	ACSIMED PRESENT.
208	(D0)	CHARACTER	8	ACSIRECT	RECORDING TECHNIQUE.
216	(D8)	CHARACTER	8	ACSIMED	MEDIA.
224	(E0)	HEXSTRING	256	-RESERVED-	RESERVED FOR FUTURE PARM EXPANSION.

480	(1E0)	AREA	8	-RESERVED-	ALIGNMENT.
480	(1E0)	LENGTH		ACSIHLN	LENGTH OF FIXED AREA.
480	(1E0)	AREA	1	ACSILIST	ELEMENT LIST DESIGNATED BY LIST= PARAMETER BEGINS HERE (FIELD ACSICNT CONTAINS THE NUMBER OF ELEMENTS IN THIS LIST).
2	(02)	CONST		ACSILDRL	LENGTH OF EACH ELEMENT (DRIVE DEVICE NUMBER) IN THE LIST USED BY THE QDRLIST FUNCTION.
1500	(5DC)	CONST		ACSIMDRL	MAXIMUM NUMBER OF ELEMENTS ALLOWED IN THE LIST USED BY THE QDRLIST FUNCTION.
6	(06)	CONST		ACSILVSL	LENGTH OF EACH ELEMENT (VOLSER) IN THE LIST USED BY THE QVOLUME FUNCTION.
500	(1F4)	CONST		ACSIMVSL	MAXIMUM NUMBER OF ELEMENTS ALLOWED IN THE LIST USED BY THE QVOLUME AND EJECT.
12	(0C)	CONST		ACSILPOL	LENGTH OF EACH ELEMENT (VOLSER RANGE PAIR) IN THE LIST USED BY THE DEFPOOL FUNCTION.
250	(FA)	CONST		ACSIMPOL	MAXIMUM NUMBER OF ELEMENTS ALLOWED IN THE LIST USED BY THE DEFPOOL FUNCTION.
4	(04)	CONST		ACSILCAP	LENGTH OF EACH ELEMENT (CAP IDENTIFIER) IN THE LIST USED BY THE QCAP FUNCTION.
500	(1F4)	CONST		ACSIMCAP	MAXIMUM NUMBER OF ELEMENTSALLOWED IN THE LIST USED BY THE QCAP FUNCTION.
4	(04)	CONST		ACSILTOK	LENGTH OF EACH ELEMENT (UNIQUE TOKEN) IN THE LIST USED BY THE QCAP FUNCTION.
500	(1F4)	CONST		ACSIMTOK	MAXIMUM NUMBER OF ELEMENTS ALLOWED IN THE LIST USED BY THE QEJECT/STOP FUNCTIONS.

Cross-Reference

The following table provides a cross-reference for the ACSINT macro:

Table 9–3 ACSINT Macro Cross-Reference

Name	Len	Offset Value
ACSIACC1	NA	02
ACSIACC2	NA	01
ACSIACS	000001	4A
ACSIACT1	000008	1A
ACSIACT2	000008	22
ACSIALC	000004	4A
ACSICAP	000004	4A
ACSICNT	000002	38

Table 9–3 (Cont.) ACSINT Macro Cross-Reference

Name	Len	Offset Value
ACSICOL	000001	50
ACSIDPOL	NA	40
ACSIDRIV	000002	54
ACSIDSCR	NA	3F
ACSIDSM	NA	2A
ACSIDSN	000044	9D
ACSIEJCT	NA	2B
ACSIEXPD	000003	C9
ACSIFDSN	NA	10
ACSIFEXP	NA	08
ACSIFJOB	NA	80
ACSIFLG1	000001	0A
ACSIFLG2	000001	0B
ACSIFLG3	000001	0C
ACSIFLG4	000001	0D
ACSIFLG5	000001	0E
ACSIFLG6	000001	0F
ACSIFLG7	000001	10
ACSIFLG8	000001	84
ACSIFLG9	000001	CF
ACSIFMED	NA	40
ACSIFMED	NA	40
ACSIFREC	NA	80
ACSIMED	000008	D8
ACSIMNT	NA	28
ACSIMOVE	NA	29
ACSIMPOL	NA	FA
ACSIMTOK	NA	1F4
ACSIFRET	NA	04
ACSIFSTP	NA	40
ACSIFVOL	NA	02
ACSIF180	NA	80
ACSIF308	NA	08
ACSIF410	NA	10
ACSIF420	NA	20
ACSIF601	NA	01
ACSIF602	NA	02

Table 9–3 (Cont.) ACSINT Macro Cross-Reference

Name	Len	Offset Value
ACSIHDR	000004	00
ACSIHLN	NA	1E0
ACSIHOST	000008	42
ACSID	NA	'CVAL'
ACSIJOB	000008	85
ACSILABT	000001	11
ACSILAL	NA	02
ACSILCAP	NA	04
ACSILDRL	NA	02
ACSILIST	000001	1E0
ACSILLDT	NA	00
ACSILNL	NA	03
ACSILNS	NA	04
ACSILOFF	000002	3A
ACSILPOL	NA	0C
ACSILSL	NA	01
ACSILSMI	000002	4A
ACSILTOK	NA	04
ACSILVSL	NA	06
ACSIMANO	NA	20
ACSIMCAP	NA	1F4
ACSIMDRL	NA	5DC
ACSIVTPN	NA	40
ACSIVTXT	NA	04
ACSIVUSR	NA	40
ACSIVVOL	NA	04
ACSIWTOR	NA	10
ACSIMVSL	NA	1F4
ACSINDEL	NA	08
ACSINNDL	NA	04
ACSINOOP	NA	00
ACSIPAN	000001	4E
ACSIPGMN	000008	95
ACSIPOOL	000001	56
ACSIPROT	NA	80
ACSIQCAP	NA	1A
ACSIQCNF	NA	14

Table 9–3 (Cont.) ACSINT Macro Cross-Reference

Name	Len	Offset Value
ACSIQDRL	NA	16
ACSIQDRV	NA	15
ACSIQDSN	NA	1D
ACSIQEJT	NA	19
ACSIQRQS	NA	1C
ACSIVDRV	NA	01
ACSIVER	000001	08
ACSIVHST	NA	80
ACSIVLSM	NA	20
ACSIVLST	NA	02
ACSIVN	NA	07
ACSIVOLS	000006	3C
ACSIVOLT	000001	CE
ACSIVPAN	NA	08
ACSIVROW	NA	04
ACSIVSCP	NA	20
ACSIVTKN	NA	08
ACSIVTLM	NA	80
ACSIROW	000001	4F
ACSIRS01	NA	01
ACSIRS02	NA	02
ACSIRS03	NA	03
ACSIRT	000001	09
ACSISCR	NA	40
ACSISCRA	NA	3D
ACSISOPR	NA	05
ACSISSCR	NA	3C
ACSISTEP	000008	8D
ACSISTOP	NA	04
ACSISUBN	NA	10
ACSISUBP	000013	77
ACSITEXT	000032	57
ACSITKNO	000004	30
ACSITLSM	000002	52
ACSITPAN	000001	51
ACSITRI	NA	85
ACSITRIL	NA	4A

Table 9–3 (Cont.) ACSINT Macro Cross-Reference

Name	Len	Offset Value
ACSIUNSC	NA	3E
ACSIUSER	000008	12
ACSIVACS	NA	40
ACSIVCAP	NA	10
ACSIVCNT	NA	01
ACSIVCOL	NA	02

IUB Record Format

IUB - IUCV Request Block The IUB describes an outstanding IUCV request resulting from an IUCV operation. Both the request and the final status are contained in the IUB data structure. The IUB is an IUCV counterpart to the 'IOBLOK' structure used by device management.

The following table provides a cross-reference for the IUB IUCV request block:

Table 9–4 IUB Request Block Reference

Name	Len	Offset Value
ACSIACC1	NA	02
ACSIACC2	NA	01
ACSIACS	000001	4A
ACSIACT1	000008	1A
ACSIACT2	000008	22
ACSIALC	000004	4A
ACSICAP	000004	4A
ACSICNT	000002	38
ACSICOL	000001	50
ACSIDPOL	NA	40
ACSIDRIV	000002	54
ACSIDSCR	NA	3F
ACSIDSM	NA	2A
ACSIDSN	000044	9D
ACSIEJCT	NA	2B
ACSIEXPD	000003	C9
ACSIFDSN	NA	10
ACSIFEXP	NA	08
ACSIFJOB	NA	80
ACSIFLG1	000001	0A
ACSIFLG2	000001	0B

Table 9–4 (Cont.) IUB Request Block Reference

Name	Len	Offset Value
ACSIFLG3	000001	0C
ACSIFLG4	000001	0D
ACSIFLG5	000001	0E
ACSIFLG6	000001	0F
ACSIFLG7	000001	10
ACSIFLG8	000001	84
ACSIFLG9	000001	CF
ACSIFMED	NA	40
ACSIFPGM	NA	20
ACSIFRET	NA	04
ACSIFSTP	NA	40
ACSIFVOL	NA	02
ACSIF180	NA	80
ACSIF308	NA	08
ACSIF410	NA	10
ACSIF420	NA	20
ACSIF601	NA	01
ACSIF602	NA	02
ACSIHDR	000004	00
ACSIHLN	NA	1E0
ACSIHOST	000008	42
ACSIID	NA	'CVAL'
ACSIJOB	000008	85
ACSILABT	000001	11
ACSILAL	NA	02
ACSILCAP	NA	04
ACSILDRL	NA	02
ACSILIST	000001	1E0
ACSILLDT	NA	00
ACSILNL	NA	03
ACSILNS	NA	04
ACSILOFF	000002	3A
ACSILPOL	NA	0C
ACSILSL	NA	01
ACSILSMI	000002	4A
ACSILTOK	NA	04
ACSILVSL	NA	06

Table 9–4 (Cont.) IUB Request Block Reference

Name	Len	Offset Value
ACSIMANO	NA	20
ACSIMCAP	NA	1F4
ACSIFREC	NA	80
ACSIMED	000008	D8
ACSIMNT	NA	28
ACSIMOVE	NA	29
ACSIMPOL	NA	FA
ACSIMTOK	NA	1F4
ACSIMVSL	NA	1F4
ACSINDEL	NA	08
ACSINNDL	NA	04
ACSINOOP	NA	00
ACSIPAN	000001	4E
ACSIPGMN	000008	95
ACSIPOOL	000001	56
ACSIPROT	NA	80
ACSIQCAP	NA	1A
ACSIQCNF	NA	14
ACSIQDRL	NA	16
ACSIQDRV	NA	15
ACSIQDSN	NA	1D
ACSIQEJT	NA	19
ACSIQRQS	NA	1C
ACSIVDRV	NA	01
ACSIVER	000001	08
ACSIVHST	NA	80
ACSIVLSM	NA	20
ACSIVLST	NA	02
ACSIVN	NA	07
ACSIVOLS	000006	3C
ACSIVOLT	000001	CE
ACSIVPAN	NA	08
ACSIVROW	NA	04
ACSIVSCP	NA	20
ACSIVTKN	NA	08
ACSIVTLM	NA	80
ACSIMDRL	NA	5DC

Table 9–4 (Cont.) IUB Request Block Reference

Name	Len	Offset Value
ACSIVTPN	NA	40
ACSIVTXT	NA	04
ACSIVUSR	NA	40
ACSIVVOL	NA	04
ACSIWTOR	NA	10
ACSIROW	000001	4F
ACSIRS01	NA	01
ACSIRS02	NA	02
ACSIRS03	NA	03
ACSIRT	000001	09
ACSISCR	NA	40
ACSISCRA	NA	3D
ACSISOPR	NA	05
ACSISSCR	NA	3C
ACSISTEP	000008	8D
ACSISTOP	NA	04
ACSISUBN	NA	10
ACSISUBP	000013	77
ACSITEXT	000032	57
ACSITKNO	000004	30
ACSITLSM	000002	52
ACSITPAN	000001	51
ACSITRI	NA	85
ACSITRIL	NA	4A
ACSIUNSC	NA	3E
ACSIUSER	000008	12
ACSIVACS	NA	40
ACSIVCAP	NA	10
ACSIVCNT	NA	01
ACSIVCOL	NA	02

Cross-Reference

The following table provides a cross-reference for the IUB:

Table 9–5 IUB Cross-Reference

Name	Len	Offset Value
IUBCC	000001	31

Table 9–5 (Cont.) IUB Cross-Reference

Name	Len	Offset Value
IUBCONN	NA	10
IUBDABQ	000016	10
IUBDAVL	000001	20
IUBDCBPT	000004	44
IUBECKBY	000001	21
IUBBECBT	000004	34
IUBEXT	000040	78
IUBEXT1	000008	78
IUBEXT2	000008	80
IUBEXT3	000008	88
IUBEXT4	000008	90
IUBEXT5	000008	98
IUBEYE	000004	0C
IUBFLG1	000001	30
IUBFLG2	000001	32
IUBHCOMM	NA	20
IUBIRT	000004	2C
IUBIUBPT	NA	24
IUBLEN	NA	A0
IUBLOK	NA	08
IUBNPOST	NA	40
IUBORGID	000004	3C
IUBPARML	000040	50
IUBPARM1	000008	50
IUBPARM2	000008	58
IUBPARM3	000008	60
IUBPARM4	000008	68
IUBPARM5	000008	70
IUBQ	000016	00
IUBREAD	NA	80
IUBREJCT	NA	08
IUBREPLY	NA	20
IUBSEND	NA	40
IUBSENT	NA	80
IUBTASK	000004	28

MEDia, RECtech, and MODel Values

This appendix provides values for MEDia, RECtech, and MODel parameters. These parameters enable you to specify transport and media characteristics. They are specified in various VM Client commands and TMI requests.

Note:

- The SL8500 library supports only the T9840A/B/C/D, T9940B, LTO, SDLT, and T10000A/B/C media types and recording techniques.
 - The SL3000 library supports only the T9840C/D, LTO, SDLT, and T10000A/B/C media types and recording techniques.
 - LTO and SDLT drives are not supported in an MVS environment. These drives are recognized by the HSC but are accessible only to open systems clients using LibraryStation.
-
-

Media Type (MEDia)

The media type, or MEDia, enables you to specify the desired type of media to be used for a data set. It is specified in the following VM Client commands and TMI requests:

- MOUNT command
- QDRLIST TMI request
- QSCRATCH TMI request
- SELSCR TMI request

The following table describes valid Media types:

Table A-1 Media Types

Media Type	Description
LONGitud	Indicates any standard or enhanced (ECART) capacity cartridges.
ZLONGI	Indicates standard, enhanced (ECART), or extended-enhanced (ZCART) capacity cartridges.
Standard	Indicates a standard length, 3480 cartridge. It can be read on any longitudinal drive (4480, 4490, 9490, or 9490EE). Data can be written in 36-track mode on a 4490, 9490, or 9490EE drive but cannot be read on an 18-track (4480) drive. Synonyms include CST, MEDIA1, STD, 1, and 3480.

Table A–1 (Cont.) Media Types

Media Type	Description
ECART	Indicates a 3490E, extended capacity cartridge. It can be used only on a 36-track drive (4490, 9490, or 9490EE). Synonyms include E, ECCST, ETAPE, Long, MEDIA2, and 3490E.
ZCART	Indicates a 3490E, extended-enhanced capacity cartridge. It can be used only on a 9490EE drive. ZCART can be abbreviated as Z.
Virtual	Indicates a VTV (Virtual Tape Volume) mounted on a VTD (Virtual Tape Drive).
HELical	<p>Indicates a helical cartridge. A helical cartridge can be used only on RedWood drives. The following subtypes and abbreviations specify a helical cartridge:</p> <ul style="list-style-type: none"> ■ DD3 indicates any DD3A, DD3B, or DD3C helical cartridge. ■ DD3A or A indicates a helical cartridge with a 10GB media capacity. ■ DD3B or B indicates a helical cartridge with a 25GB media capacity. ■ DD3C or C indicates a helical cartridge with a 50GB media capacity. <p>The seventh position in the external label is encoded with the cartridge type (that is, A, B, or C).</p>
STK1	Indicates any T9840 cartridge.
STK1R	<p>Indicates a T9840 data cartridge. The media indicator in the external label is encoded with the cartridge type (R). STK1R can be abbreviated as R.</p> <p>T9840 cartridge media capacities are 20GB (T9840A and T9840B), 40GB (T9840C), or 75GB (T9840D).</p>
STK1U	Indicates a T9840A, T9840B, or T9840C cleaning cartridge. STK1U can be abbreviated as U.
STK1Y	Indicates a T9840D cleaning cartridge. STK1Y can be abbreviated to Y.
STK2	Indicates any T9940 cartridge.
STK2P	<p>Indicates a T9940 data cartridge. STK2P can be abbreviated as P.</p> <p>T9940 cartridge media capacities are 60GB (T9940A) or 200GB (T9940B).</p>
STK2W	Indicates a T9940 cleaning cartridge. STK2W can be abbreviated as W.
T10000T1	Indicates a full-capacity 500GB T10000A or 1TB T10000B cartridge. T10000T1 can be abbreviated as T1.
T10000TS	Indicates a smaller-capacity 120GB T10000A or 240GB T10000B cartridge. T10000TS can be abbreviated as TS.
T10000CT	Indicates a T10000A or T10000B cleaning cartridge. T10000CT can be abbreviated as CT.
T10000T2	Indicates a full-capacity 5TB T10000C cartridge. T10000T2 can be abbreviated as T2.
T10000TT	Indicates a smaller-capacity 1TB T10000C cartridge. T10000TT can be abbreviated as TT.
T10000CL	Indicates a T10000A, T10000B, or T10000C cleaning cartridge. T10000CL can be abbreviated as CL.

Note:

- T10000C drives can read T10000T1 or T10000TS media but cannot write to that media.
- T10000C drives can only write to T10000T2 or T10000TT media.

When the MEDia parameter is not specified, a default is chosen based on the value of the RECTech parameter. The following table shows default values used if MEDia is omitted:

Table A–2 Media Type Defaults

RECTech Entered	MEDia Default
18track	Standard
36track, 36Atrack, 36Btrack	LONGitud
36Ctrack	ZLONGI
LONGitud	LONGitud
DD3, Helical	DD3A
STK1R, STK1R34, STK1R35, STK1RA, STK1RA34, STK1RA35, STK1RB, STK1RB34, STK1RB35, STK1RAB, STK1RAB34, STK1RAB35, STK1RC, STK1RC34, STK1RC35, STK1RD, STK1RDE, STK1RDN, STK1RD34, STK1RD35, STK1RDE4, STK1RDE5	STK1R
STK2P, STK2P34, STK2P35, STK2PA, STK2PA34, STK2PA35, STK2PB, STK2PB34, STK2PB35	STK2P
T10K, T10KN, T10KE, T10KA, T10KAN, T1A34, T1A35, T10KAE, T1AE34, T1AE35, T10KC, T10KCN, T1C34, T1C35, T10KCE, T1CE34, T1CE35	T10000T1

Note:

- T10000C drives can read T10000T1 or T10000TS media but cannot write to that media.
- T10000C drives can only write to T10000T2 or T10000TT media.

Recording Technique (RECTech)

The recording technique, or RECTech, enables you to specify the method used to record data tracks on the tape surface for the desired data set. It is specified in the following VM Client TMI requests:

- QDRLIST
- QSCRATCH
- SELSCR

The following table describes valid recording techniques:

Table A–3 Recording Techniques

Recording Technique	Description
LONGitud	Indicates any device that uses longitudinal recording. Devices include 4480, 4490, 9490, and 9490EE drives.
18track	Indicates a 4480 drive.
36track	Indicates a 4490, 9490, or 9490EE drive (any device that records in 36-track mode).
36Atrack	Indicates a 4490 (Silverton) drive.
36Btrack	Indicates a 9490 (Timberline) drive.
36Ctrack	Indicates a 9490EE drive.
HELical	Indicates a device using helical recording.
DD3	Indicates a device using helical recording.
STK1R	Indicates any T9840 drive.
STK1R34	Indicates a 3490E-image T9840 drive.
STK1R35	Indicates a 3590-image T9840 drive.
STK1RA	Indicates any T9840A drive.
STK1RA34	Indicates a 3490E-image T9840A drive.
STK1RA35	Indicates a 3590-image T9840A drive.
STK1RB	Indicates any T9840B drive.
STK1RB34	Indicates a 3490E-image T9840B drive.
STK1RB35	Indicates a 3590-image T9840B drive.
STK1RAB	Indicates any T9840A or T9840B drive.
STK1RAB4	Indicates a 3490E-image T9840A or T9840B drive.
STK1RAB5	Indicates a 3590-image T9840A or T9840B drive.
STK1RC	Indicates any T9840C drive.
STK1RC34	Indicates a 3490E-image T9840C drive.
STK1RC35	Indicates a 3590-image T9840C drive.
STK1RD	Indicates any T9840D drive.
STK1RDE	Indicates an encryption-enabled T9840D drive.
STK1RDN	Indicates a non-encryption-enabled T9840D drive.
STK1RD34	Indicates a non-encryption-enabled 3490E-image T9840D drive.
STK1RD35	Indicates a non-encryption-enabled 3590-image T9840D drive.
STK1RDE4	Indicates an encryption-enabled 3490E-image T9840D drive.
STK1RDE5	Indicates an encryption-enabled 3590-image T9840D drive.
STK2P	Indicates any T9940 drive.

Table A–3 (Cont.) Recording Techniques

Recording Technique	Description
STK2P34	Indicates a 3490E-image T9940 drive.
STK2P35	Indicates a 3590-image T9940 drive.
STK2PA	Indicates any T9940A drive.
STK2PA34	Indicates a 3490E-image T9940A drive.
STK2PA35	Indicates a 3590-image T9940A drive.
STK2PB	Indicates any T9940B drive.
STK2PB34	Indicates a 3490E-image T9940B drive.
STK2PB35	Indicates a 3590-image T9940B drive.
T10K	Indicates any T10000 drives.
T10KN	Indicates all non-encrypted T10000 drives.
T10KE	Indicates all encrypted T10000 drives.
T10KA	Indicates any T10000A drive.
T10KAN	Indicates a non-encryption enabled 3490E- or 3590-image T10000A drive.
T1A34	Indicates a non-encryption enabled 3490E-image T10000A drive.
T1A35	Indicates a non-encryption enabled 3590-image T10000A drive.
T10KAE	Indicates an encryption-enabled 3490E- or 3590-image T10000A drive.
T1AE34	Indicates an encryption-enabled 3490E-image T10000A drive.
T1AE35	Indicates an encryption-enabled 3590-image T10000A drive.
T10KB	Indicates any T10000B drive.
T10KBN	Indicates a non-encryption enabled 3490E- or 3590-image T10000B drive.
T1B34	Indicates a non-encryption enabled 3490E-image T10000B drive.
T1B35	Indicates a non-encryption enabled 3590-image T10000B drive.
T10KBE	Indicates an encryption-enabled 3490E- or 3590-image T10000B drive.
T1BE34	Indicates an encryption-enabled 3490E-image T10000B drive.
T1BE35	Indicates an encryption-enabled 3590-image T10000B drive.
T10KC	Indicates any T10000C drive.
T10KCN	Indicates a non-encrypted 3490E- or 3590-image T10000C drive.
T1C34	Indicates a non-encrypted 3490E-image T10000C drive.
T1C35	Indicates a non-encrypted 3590-image T10000C drive.
T10KCE	Indicates an encryption-enabled 3490E- or 3590-image T10000C drive.
T1CE34	Indicates an encryption-enabled 3490E-image T10000C drive.
T1CE35	Indicates an encryption-enabled 3590-image T10000C drive.
Virtual	Indicates a VTV (Virtual Tape Volume) mounted on a VTD (Virtual Tape Drive).

When the RECtech parameter is not specified, a default is chosen based on the value of the MEDIA parameter. The following table shows default values used if RECtech is omitted.

Table A–4 Recording Technique Defaults

MEDIA Entered	RECtech Default
LONGitud	LONGitud
ZLONGI	LONGitud
Standard	LONGitud
ECART	36track
ZCART	36Ctrack
DD3A, DD3B, DD3C, DD3D	DD3
STKR, STK1U, STKY	STK1R
STK2P, STK2W	STK2P
T10000T1, T10000TS, T10000CL	T10K
T10000CT	T10KA and T10KB
T10000T2, T10000TT	T10KC
Virtual	Virtual

Model Type (MODE1)

The model type, or MODE1, enables you to specify the model number of a transport (drive), or drive. MODE1 provides the same type of information as RECtech, however, a user may find it more convenient to specify a transport model rather than a recording technique.

Note:

- MODE1 and RECtech are mutually exclusive.
 - The SL8500 library supports only model types associated with T9840, T9940, and T10000 series drives.
 - You can specify multiple values for this parameter; separate each value with a comma.
-
-

The following table describes valid model types:

Table A–5 Model Types

Model Type	Description
4480	Indicates a 4480 (18-track) drive.
4490	Indicates a 4490 (36-track Silverton) drive.
9490	Indicates a 9490 (36-track Timberline) drive.
9490EE	Indicates a 9490EE (36-track Timberline EE) drive.
SD3	Indicates an SD-3 (RedWood) drive.
9840	Indicates a 3490E-image T9840A drive.

Table A–5 (Cont.) Model Types

Model Type	Description
984035	Indicates a 3590-image T9840A drive.
T9840B	Indicates a 3490E-image T9840B drive.
T9840B35	Indicates a 3590-image T9840B drive.
T9840C	Indicates a 3490E-image T9840C drive.
T9840C35	Indicates a 3590-image T9840C drive.
T9840D	Indicates a non encryption-enabled 3490E-image T9840D drive.
T9840D35	Indicates a non encryption-enabled 3590E-image T9840D drive.
T9840DE	Indicates an encryption-enabled 3490E-image T9840D drive.
T9840DE5	Indicates an encryption-enabled 3590E-image T9840D drive.
T9940A	Indicates a 3490E-image T9940A drive.
T9940A35	Indicates a 3590-image T9940A drive.
T9940B	Indicates a 3490E-image T9940B drive.
T9940B35	Indicates a 3590-image T9940B drive.
T1A34	Indicates a non-encryption enabled 3490E-image T10000A drive.
T1A35	Indicates a non-encryption enabled 3590-image T10000A drive.
T1AE34	Indicates an encryption-enabled 3490E-image T10000A drive.
T1AE35	Indicates an encryption-enabled 3590-image T10000A drive.
T1B34	Indicates a non-encryption enabled 3490E-image T10000B drive.
T1B35	Indicates a non-encryption enabled 3590E-image T10000B drive.
T1BE34	Indicates an encryption-enabled 3490E-image T10000B drive.
T1BE35	Indicates an encryption-enabled 3590-image T10000B drive.
T1C34	Indicates a non-encryption enabled 3490E-image T10000C drive.
T1C35	Indicates a non-encryption enabled 3590-image T10000C drive.
T1CE34	Indicates an encryption-enabled 3490E-image T10000C drive.
T1CE35	Indicates an encryption-enabled 3590-image T10000C drive.
Virtual	Indicates a VTV (Virtual Tape Volume) mounted on a VTD (Virtual Tape Drive).

Diagnostics

This appendix describes diagnostic information you may be asked to provide when you contact Oracle for VM Client support.

This information includes the following:

- SMCCMDS and SMCPARMS data set files
- VM Client console log
- TRACE files
- Set MSGDEF LVL=28
- System DUMPS
- Display of VM Client maintenance (VMFINFO)
- VM level (Q CPLEVEL)
- CMS level (Q CMSLEVEL)
- VM TMS maintenance level

See [Preface](#) for information about contacting Oracle for support.

Index

A

ACSINT DSECT, 9-62
ACSLs, XAPI client interface, 1-2
ACSRQ Macro
 overview, 9-12
 requests
 DISMOUNT, 9-14
 EJECT, 9-16
 MOUNT, 9-19
 MOVE, 9-22
 QCAP, 9-25
 QCONFIG, 9-27
 QDRIVES, 9-29
 QSCRATCH, 9-34
 QVOLUME, 9-37
 QVOLUME, 9-39
 SCRATCH, 9-40
 SELSCR, 9-42
 UNSCRATCH, 9-45
 syntax, 9-13
AUTHorize command, 6-1

C

CD-ROM contents, 2-1
CMS command, 6-3
command files, 5-2
commands
 AUTHorize, 6-1
 CMS, 6-3
 COMMtest, 6-3
 CP, 6-5
 DISMount, 6-5
 Display DRive, 6-6
 Display RC, 6-8
 DRIVemap, 6-9
 DUMP, 6-11
 DUMPOpts, 6-11
 EXIT, 6-12
 Help, 6-12
 issuing, 6-1
 LIst, 6-13
 LOGdisk, 6-14
 MOunt, 6-15
 MSGDef, 6-17

OPERator, 6-18
POOLmap, 6-19
READ, 6-20
RESYNChronize, 6-21
Route, 6-22
SERVer, 6-23
TAPEPlex, 6-26
TCPip, 6-28
TRace, 6-30
COMMtest command, 6-3
CP command, 6-5
CP DETACH support, 5-3

D

DASD requirements, 2-3
data flow, VM Client, 1-2
determining VM Client resource requirements, 3-2
diagnostics, B-1
DISMount command, 6-5
DISMOUNT request, 9-14
Display DRive command, 6-6
Display RC command, 6-8
DRIVemap command, 6-9
DUMP command, 6-11
DUMPOpts command, 6-11

E

EJECT request, 9-16
ELS, 2-2
EXIT command, 6-12

F

features, VM Client, 1-1

H

hardware requirements, 2-2
Help command, 6-12
HTTP server, 7-1

I

installation

- allocating VM Client resources, 3-3
- building VM Client executable code, 3-4
- creating a PPF override file, 3-3
- creating VM Client service machine, 3-5
- customizing VM Client machine files, 3-5
- DASD requirements, 2-3
- IBM VSES/E, 2-1, 3-1, 4-1
- installing VM Client product files, 3-3
- MVS requirements, 2-2
- placing VM Client into production, 3-5
- software and hardware requirements, 2-2
- summary of steps, 3-1
- testing VM Client, 3-5
- installation contents, 2-1
- interface data areas, 9-46
- introduction, 1-1
- issuing commands, 6-1
- IUB record format, 9-70
- IUCV considerations, 9-5

L

- LlSt command, 6-13
- LOGdisk command, 6-14

M

- maintenance, installing, 4-1
- MAXRC keyword value pair, 5-2
- Media Type (MEDIA) values, A-1
- messages, 8-1
- MOdel Type (MOdel) values, A-6
- MOunt command, 6-15
- MOUNT request, 9-19
- MOVE request, 9-22
- MSGDef command, 6-17
- MVS requirements, 2-2

O

- OPERator command, 6-18
- OPERATOR keyword value pair, 5-2
- overview, 1-1

P

- POOLmap command, 6-19
- PPF override file, creating, 3-3

Q

- QCAP request, 9-25
- QCONFIG request, 9-27
- QDRIVES request, 9-29
- QSCRATCH request, 9-34
- QVOLUME request, 9-37
- QVOLUME request, 9-39

R

- READ command, 6-20

- Recording Technique (RECtech) values, A-3
- RESYNChronize command, 6-21
- Route command, 6-22

S

- SCRATCH request, 9-40
- scratch subpools, 7-1
- SELSCR request, 9-42
- SERVer command, 6-23
- server considerations, 7-1
- service machine, creating, 3-5
- SLX macro, 9-47
- SMC HTTP server, 7-1
- SMCBINT module parameters, 5-1
- SMCCMDS command file, 5-2
- SMCPARMS command file, 5-2
- software requirements, 2-2
- starting VM Client, 5-1
- syntax
 - ACSRQ macro, 9-13
 - AUTHorize command, 6-2
 - CMS command, 6-3
 - COMMtest command, 6-3
 - CP command, 6-5
 - DISMOUNT command, 6-5
 - DISMOUNT request, 9-14
 - Display DRive command, 6-6
 - Display RC command, 6-8, 6-9
 - Display Volume command, 6-9
 - DRIVemap command, 6-9
 - DUMP command, 6-11
 - DUMPOpts command, 6-11
 - EJECT request, 9-17
 - EXIT command, 6-12
 - Help command, 6-13
 - LlSt command, 6-13
 - LOGdisk command, 6-14
 - MOunt command, 6-15
 - MOUNT request, 9-20
 - MOVE request, 9-22
 - MSGDef command, 6-17
 - OPERator command, 6-18
 - POOLmap command, 6-19
 - QCAP request, 9-25
 - QCONFIG request, 9-28
 - QDRIVES request, 9-29
 - QDRLIST request, 9-31
 - QSCRATCH request, 9-34
 - QVOLUME request, 9-37
 - QVOLUME request, 9-39
 - READ command, 6-20
 - RESYNChronize command, 6-21
 - Route command, 6-22
 - SCRATCH request, 9-41
 - SELSCR request, 9-42
 - SERVer command, 6-23
 - TAPEplex command, 6-26
 - TCPip command, 6-28
 - TRace command, 6-30

UNSCRATCH request, 9-45

T

tape management interface (TMI), 9-1

TAPEPlex command, 6-26

TCPIP command, 6-28

testing VM Client, 3-5

TMS

 decision points, 9-3

 interaction with VM Client, 9-7

 overview, 9-2

TRace command, 6-30

TRACE keyword value pair, 5-1

U

UNSCRATCH request, 9-45

V

VM Client overview, 1-1

VMSES/E, 2-1, 3-1, 4-1

VTCS management classes, 7-2

X

XAPI client interface to ACSLS server, 1-2

Z

ZIP file contents, 2-1

