TAPEMANAGER

Addendum

Cartridge Library Installation Guide

for

Unisys MCP Systems

Release 10.070

June 2023

Dynamic Solutions
INTERNATIONAL

Company Confidential

Copyright

Correspondence regarding this document should be addressed to:
Dynamic Solutions International, Inc.
Product Development Group
8744 Lucent Blvd. Suite 106 Highlands Ranch 80129
(800)641-5215 or (303)754-2000
Technical Support Hot-Line (800)332-9020
E-Mail: support@dynamicsolutions.com

Contents

Contents

Overview

This is an addendum to the Cartridge Library Installation Guide to document the additional interfaces that support various features available in Virtual Tape Libraries (VTL) accessed via DSI LibraryManager. This document is released under license or Custom Engineering Request (CER) only.

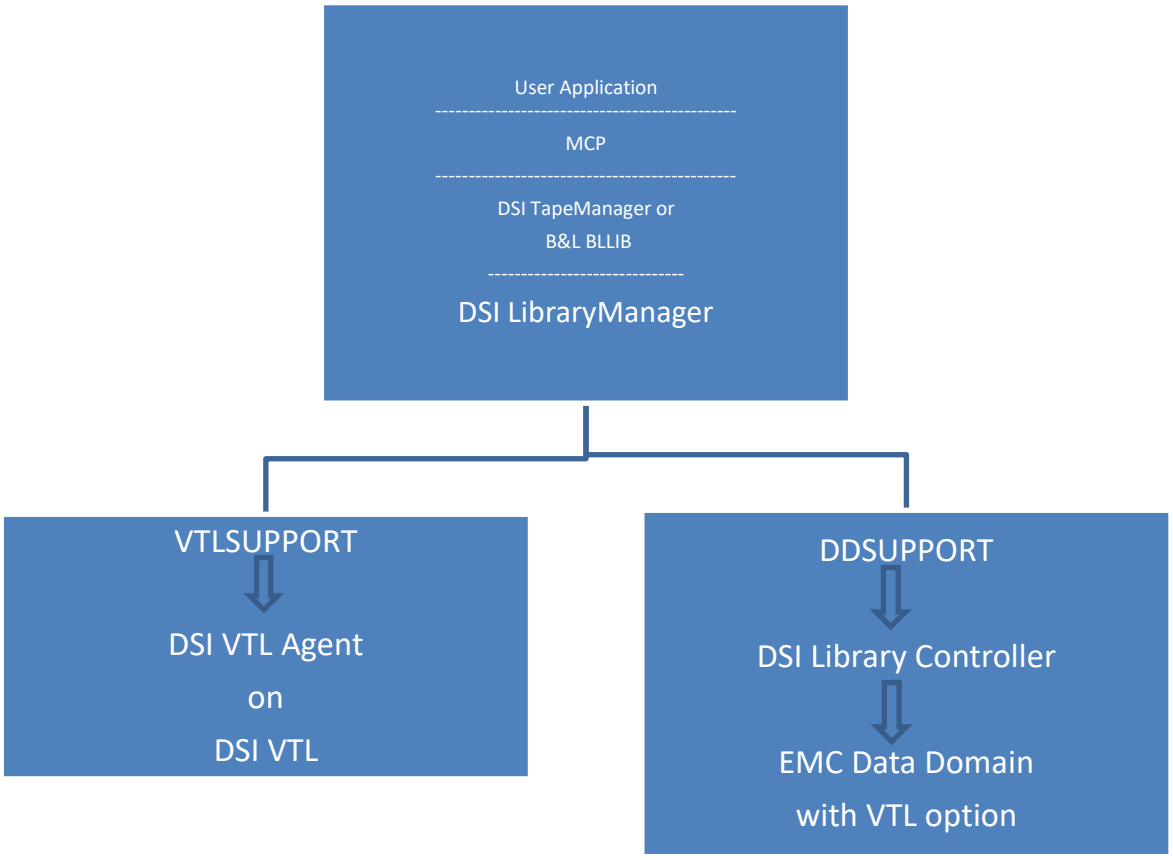
Requirements

- SYSTEM/VTLSUPPORT
  - DSI VTL Agent
  - DSI VTL

And/or

- SYSTEM/DDSUPPORT
  - DSI Library Controller - IPF
  - EMC Data Domain system with VTL option

Tape Library System Components



Appendix A - Addendum
Library Support Programming Interface

The following section describes additional information for VTL (VTL Agent) support for the programming interface for the Library Support software library.

Caution

DSI reserves the right to change this interface at any time. DSI will attempt to give ample warning of changes where practical. The user should review the release notes included with each release to determine if any changes have been or will be made to this interface.

LIBRARY_CONFIGURE Procedure

```
DEFINE % CINFO ARRAY OF LIBRARY_CONFIGURE
     LC_ACTION     = CINFO[0]#
         ,LC_SETV       = 1#
         ,LC_RESETV     = 2# % set to default values
         ,LC_INQV       = 3#
    ,LC_ITEM       = CINFO[1]#
         ,LC_LOGV       = 1#
         ,LC_DISKV      = 2#
         ,LC_MONITORV   = 3#
    % Following CINFO defines vary depending on LC_ITEM
    % LC_LOGV defines
    ,LC_LOG_SIZE   = CINFO[2]#
    ,LC_LOG_TIME   = CINFO[3]#
    ,LC_LOG_COPIES = CINFO[4]#
    % LC_DISKV defines
    ,LC_DISK_MIN   = CINFO[2]#
    ,LC_DISK_PCT   = CINFO[3]#
    ,LC_DISK_TIME  = CINFO[4]#
    % LC_MONITORV defines
    ,LC_MON_TIME   = CINFO[2]#
    ,LC_MON_LOG    = CINFO[3]#
    ,LC_MON_VAULT  = CINFO[4]#
    ,LC_MON_INVEN  = CINFO[5]#
    ,LC_CINFO_SZV  = 6# % WORDS
    ;
BOOLEAN PROCEDURE LIBRARY_CONFIGURE(LIB_ID, CINFO);
VALUE LIB_ID;
REAL
     LIB_ID             % LIBRARY ID (HANDLE) TO CONFIGURE
    ;
ARRAY
     CINFO[0]           % CONFIGURATION INFORMATION
          ;
      LIBRARY LIBRARYSUPPORT;
```

Function: Sets or retrieves various configuration values for VTL libraries.

Usage: This procedure is used to set or retrieve configuration parameters used to manage or monitor functions in a VTL library.

Parameters:

LIBRARY_ID	(Input) the library id of the library that is to process the command.	
CINFO	(Input/Output) the configuration parameters set or retrieved as follows:	
	CINFO[0]	Action word, values are 1 to set to supplied values, 2 to reset to defaults (parameters ignored), 3 to return the current values
	CINFO[1]	Item word, values are 1 to set/return logging parameters, 2 to set/return disk monitoring parameters, 3 to set/return performance monitoring
	CINFO[2-4]	For logging items word 2 is the size in MB the log can grow to before being transferred, word 3 is time in minutes before a log is transferred, word 4 is the number of logs to keep on disk
	CINFO[2-4]	For disk monitoring items word 2 is the size in MB to send an alert if available space falls below this amount, word 3 is the minimum percent of available disk below which an alert is sent, word 4 is the time in minutes that the disk space should be checked
	CINFO[2-5]	For performance monitoring items word 2 is the time in minutes to report performance statistics. Words 3-5 activate or deactivate the monitoring of the VTL log, VTL virtual vault, and/or the virtual library inventory by setting the field to a 1 or zero value. *NOTCE: the performance monitoring is not supported on VTL 8.x and later systems. Unisys Operations Sentinel should be used as a replacement.*

Results: The sets or returns the current parameters for the specified item.

Possible errors: 32, 33

LIBRARY_INFO Procedure

This procedure returns information about a specified tape library. If the tape library is a VTL the LIBRARY_DATA parameter returns a fifth row of information specific to VTLs defined as follows.

```
,VTL_ID           =  VTL_DATAW[0]#   %  1 WORD
,VTL_LIB_NAME     =P(VTL_DATAW[1])#  % 11 WORDS -64 BYTES
,VTL_VTL_NAME     =P(VTL_DATAW[12])# % 11 WORDS -64 BYTES
,VTL_ADDR         =P(VTL_DATAW[23])# %  8 WORDS -46 BYTES
,VTL_OPTIONS      =  VTL_DATAW[31] # %  1 WORD
     ,VTL_ADDR_IS_DOMAIN=B(VTL_OPTIONS).[47: 1]#
,VTL_MONITOR      =  VTL_DATAW[32] # %  1 WORD
     ,VTL_MON_IO      = B(VTL_MONITOR).[ 7: 1]#
     ,VTL_MON_MEM     = B(VTL_MONITOR).[ 6: 1]#
     ,VTL_MON_QUE     = B(VTL_MONITOR).[ 5: 1]#
     ,VTL_MON_CPU     = B(VTL_MONITOR).[ 4: 1]#
     ,VTL_MON_LOG     = B(VTL_MONITOR).[ 3: 1]#
     ,VTL_MON_VAULT   = B(VTL_MONITOR).[ 2: 1]#
     ,VTL_MON_TAPES   = B(VTL_MONITOR).[ 1: 1]#
     ,VTL_MON_STORAGE = B(VTL_MONITOR).[ 0: 1]#
,VTL_STATE        =  VTL_DATAW[33] # %  1 WORD
     %VTL_DEDUPED     = B(VTL_STATE).[10: 1]# % DEDUPE VTL
     %VTL_SSH         = B(VTL_STATE).[ 9: 1]# % USE SSH CONN
     %VTL_LICENSED    = B(VTL_STATE).[ 8: 1]# % AGENT LICENSED
     %VTL_FAILOVER    =   VTL_STATE .[ 7: 3]# % VTL FAILO STATE
     %VTL_INIT        = B(VTL_STATE).[ 4: 1]# % THIS DATA VALID
     %VTL_NEWINVEN    = B(VTL_STATE).[ 3: 1]# % INVEN UPDATED
     %VTL_TALKING     = B(VTL_STATE).[ 2: 1]# % PORT TO AGENT
     %VTL_ENABLED     = B(VTL_STATE).[ 1: 1]#
     %VTL_ACTIVE      = B(VTL_STATE).[ 0: 1]# % VTLDRIVER UP
,VTL_MSG_Q        =  VTL_DATAW[34] # %  1 WORD
,VTL_MIXNUM       =  VTL_DATAW[35] # %  1 WORD
,VTL_BUFFSIZE     =  VTL_DATAW[36] # %  1 WORD
,VTL_AGENT_VER    =P(VTL_DATAW[37])# %  6 WORDS
,VTL_VTL_VER      =P(VTL_DATAW[43])# % 11 WORDS
,VTL_OS_VER       =P(VTL_DATAW[54])# % 11 WORDS
,VTL_LIB_COUNT    =  VTL_DATAW[65] # %  1 WORD
,VTL_VID          =  VTL_DATAW[66] # %  1 WORD
,VTL_VENDOR       =P(VTL_DATAW[67])# %  3 WORDS -16 BYTES
,VTL_PRODID       =P(VTL_DATAW[70])# %  3 WORDS -16 BYTES
,VTL_REVISION     =P(VTL_DATAW[73])# %  3 WORDS -16 BYTES
,VTL_SERIALNO     =P(VTL_DATAW[76])# %  3 WORDS -16 BYTES
,VTL_DRIVES       =  VTL_DATAW[79] # %  1 WORD
,VTL_SLOTS        =  VTL_DATAW[80] # %  1 WORD
,VTL_TAPES        =  VTL_DATAW[81] # %  1 WORD
,VTL_BCSTART      =P(VTL_DATAW[82])# %  3 WORDS -16 BYTES
,VTL_BCEND        =P(VTL_DATAW[85])# %  3 WORDS -16 BYTES
,VTL_COD          =B(VTL_DATAW[88])# %  1 WORD
,VTL_CAPACITY     =  VTL_DATAW[89] # %  1 WORD
,VTL_MEDIA        =P(VTL_DATAW[90])# %  3 WORDS -16 BYTES
,VTL_ARCHIVE      =B(VTL_DATAW[93])# %  1 WORD
,VTL_CACHING      =B(VTL_DATAW[94])# %  1 WORD
,VTL_HOSTKEY      =  VTL_DATAW[95] # %  1 WORD
,VTL_DISK_SIZE    =  VTL_DATAW[96] # %  1 WORD
,VTL_DISK_AVAIL   =  VTL_DATAW[97] # %  1 WORD
,VTL_DISK_MIN     =  VTL_DATAW[98] # %  1 WORD
,VTL_DISK_PCT     =  VTL_DATAW[99] # %  1 WORD
,VTL_DISK_TIME    =  VTL_DATAW[100] # %  1 WORD
,VTL_LM_ID        =  VTL_DATAW[101]# %  1 WORD
,VTL_LOG_MAXAGE   =  VTL_DATAW[102]# %  1 WORD
,VTL_LOG_MAXSIZE  =  VTL_DATAW[103]# %  1 WORD
,VTL_LOG_COPIES   =  VTL_DATAW[104]# %  1 WORD
```

```
,VTL_CPU_CHECKTIME=  VTL_DATAW[105]# %  1 WORD
,VTL_QUE_CHECKTIME=  VTL_DATAW[106]# %  1 WORD
,VTL_MEM_CHECKTIME=  VTL_DATAW[107]# %  1 WORD
,VTL_IO_CHECKTIME =  VTL_DATAW[108]# %  1 WORD
,VTL_BELIB_ID     =  VTL_DATAW[109]# %  1 WORD
,VTL_VIA_LIBID    =  VTL_DATAW[109]# %  1 WORD - REDEFINE
,VTL_DOORS        =  VTL_DATAW[110]# %  1 WORD
,VTL_DDPOOL       =P(VTL_DATAW[111])#%  11 WORDS -64 BYTES
,VTL_DDUSER       =P(VTL_DATAW[122])#%  11 WORDS -64 BYTES
,VTL_DDPWRD       =P(VTL_DATAW[133])#%  11 WORDS -64 BYTES
,VTL_BE_LIBID(N)  =  VTL_DATAW[144+(N-1)]#% 16 WORDS
,VTL_POLICIES     =  VTL_DATAW[160]#%   1 WORD
,VTL_ROLES        =P(VTL_DATAW[161])#%  3 WORDS -16 BYTES
% Right now the values returned are:
%    "VTL"
%    "VTLS"
%    "SIR"
,VTL_INFO_SZV     = 164# % WORDS
;
ARRAY VTL_DATAW[0] = LIBRARY_DATA[4,*];
```

The data returned in this array is primarily configuration information for the VTL and may be a repeat of information in the other rows of the array. However, two items are important for using the VTL_COMMAND procedure. The VTL_VID is the VTL VID number for this library. VTL_BELIB_ID is the library ID (not VID) of a physical library attached to the VTL if one has been configured.

The VTL_FAILOVER field has the following values.

0 = not configured for failover.
1 = Not failed over & fail over not suspended
2 = Not failed over BUT fail over suspended
3 = Fail over occurred and fail over not suspended.
4 = Fail over occurred and fail over IS suspended.
5 = Fail over to other system occurred, fail over not suspended
6= Fail over to other system occurred, fail over IS suspended
10= An Error occurred getting the fail over status

NOTE: Fail over status of 5 or 6 probably won't be seen as the client should be using the IP address that will get taken over by the "other system" which should be responding with the status of 3 or 4.

SLOT_STATUS Procedure

This procedure returns information about a specified tape library storage slot. If the tape library is a VTL and the SLOT_STAT_VTLINFO item is set to 1 then additional of information specific to VTLs is returned as follows.

```
BOOLEAN PROCEDURE SLOT_STATUS(LIBRARY_ID, SLOT, STATUS_DATA);
VALUE LIBRARY_ID, SLOT;
REAL
     LIBRARY_ID         % LIBRARY ID (HANDLE) THAT HAS SLOT
    ,SLOT               % SLOT TO RETURN STATUS FOR
    ;
EBCDIC ARRAY
     STATUS_DATA[0]     % STATUS INFO RETURNED FOR SLOT
    ;
     LIBRARY LIBRARYSUPPORT;

DEFINE % LAYOUT OF STATUS_DATA ARRAY RETURNED BY SLOT_STATUS
     SLOT_STAT_NUMBER   = STATUS_DATAW[0].[47:16]# % ELEMENT NUM
    ,SLOT_STAT_SLOTNUM  = STATUS_DATAW[0].[31:16]# % EXTERNAL NUM
    ,SLOT_STAT_DD       = STATUS_DATAW[0].[8: 1]# % SLOT IN DD
    ,SLOT_STAT_VIAVTL   = STATUS_DATAW[0].[7: 1]# % SLOT IN BACKEND
    ,SLOT_STAT_VTL      = STATUS_DATAW[0].[6: 1]# % SLOT IN VIRTUAL
    ,SLOT_STAT_LA       = STATUS_DATAW[0].[5: 1]# % LIBA CONTROLLED
    ,SLOT_STAT_CSCA     = STATUS_DATAW[0].[4: 1]# % CSC-A CONTROLL
    ,SLOT_STAT_ACCESS   = STATUS_DATAW[0].[3: 1]# % ACCESS ALLOWED
    ,SLOT_STAT_EXCEPT   = STATUS_DATAW[0].[2: 1]# % ABNORMAL STATE
    ,SLOT_STAT_SIDE2    = STATUS_DATAW[0].[1: 1]# % SIDE 2 LOADED
    ,SLOT_STAT_FULL     = STATUS_DATAW[0].[0: 1]# % TAPE IN SLOT
    ,SLOT_STAT_VOL_TAG  = STATUS_DATAW[1]#        % BAR CODE LABEL
    ,SLOT_STAT_VOL_TAG2 = STATUS_DATAW[2]#        % SN SIDE 2
    ,SLOT_STAT_UNIT     = STATUS_DATAW[3]#        % USING DRIVE
    ,SLOT_STAT_NOCONN   = STATUS_DATAW[4].[8: 1]# % CONNECTION=NONE
    ,SLOT_STAT_COMPRESS = STATUS_DATAW[4].[7: 1]# % COMPRESSION SET
    ,SLOT_STAT_CACHING  = STATUS_DATAW[4].[6: 1]# %
    ,SLOT_STAT_DUPL     = STATUS_DATAW[4].[5: 1]# %
    ,SLOT_STAT_REPL     = STATUS_DATAW[4].[4: 1]# %
    ,SLOT_STAT_ARCHIVE  = STATUS_DATAW[4].[3: 1]# %
    ,SLOT_STAT_READONLY = STATUS_DATAW[4].[2: 1]# % CART READ ONLY
    ,SLOT_STAT_COD      = STATUS_DATAW[4].[1: 1]# % COD IS SET
    ,SLOT_STAT_VTLINFO  = STATUS_DATAW[4].[0: 1]# % VTL INFO VALID
    ,SLOT_STAT_CAPACITY = STATUS_DATAW[5]#        % CART SIZE MB
    ,SLOT_STAT_CARTSIZE = STATUS_DATAW[6]#        % MB USED BY CART
    ,SLOT_STAT_DATASIZE = STATUS_DATAW[7]#        % MB USED BY DATA
    ,SLOT_STAT_VID      = STATUS_DATAW[8]#        % VTL VID OF CART
    ,SLOT_STAT_USEDSIZE = STATUS_DATAW[9]#        % MB USED BY USER
    ,SLOT_STAT_CTIME    = STATUS_DATAW[10]#       % VTL CREATE TIME
    ,SLOT_STAT_MTIME    = STATUS_DATAW[11]#       % VTL MODIFY TIME
    ,SLOT_STAT_REPVID   = STATUS_DATAW[12]#       % REPLICA VID
    ;

ARRAY STATUS_DATAW[0] = STATUS_DATA;
```

Function: Returns current status information about a library storage slot.

Usage: This procedure will return the current status information for a library storage slot. The procedure is used to diagnose errors and create information displays for the user.

Parameters: LIBRARY_ID (Input) the ID of the library that contains the storage slot.

SLOT — (Input) the storage slot number to return status for.

STATUS_DATA — (Output) the status information is returned as a set of fields. These fields are defined as follows:

Word 0

[47:16]	the element number of the slot as defined by the library hardware
[31:16]	the slot number
[08:01] = 1	if the slot is in Data Domain system
[07:01] = 1	if the slot is in back end library
[06:01] = 1	if the slot is in virtual library
[05:01] = 1	if the slot is controlled by StorageTek Library Attach.
[04:01] = 1	if the slot is controlled by Unisys CSC-A.
[03:01] = 1	if the slot is accessible (can be used).
[02:01] = 1	if the slot has an error reported against it.
[01:01] = 1	if the media has 2 sides and the second side is in use in the drive unit.
[00:01] = 1	if the slot has a cartridge assigned to it.

All other fields in this word are reserved for future use.

Word 1

If the slot has a cartridge assigned to it, this field contains the right most 6 characters of the bar code label of the cartridge. If the library does not have a barcode reader, the serial number of the cartridge is returned if known.

Word 2

If the slot has a cartridge assigned to it, and the media has 2 sides, this is the serial number of the second side otherwise it is zero.

Word 3

If the slot has a cartridge assigned to it and the cartridge is in use by a drive, the drives' unit number is stored here.

Word 4

[08:01] = 1	if library is CONNECTION=NONE
[07:01] = 1	if the tape is set for compression
[06:01] = 1	if caching is set for the logical library

[02:01] = 1 if the is read-only

[01:01] = 1 if the tape has Capacity-On-Demand set.

[00:01] = 1 (Input) request VTL tape info. (Output) VTL tape info valid.

All other fields in this word are reserved for future use.

Word 5

The raw capacity of the virtual cartridge in megabytes.

Word 6

The amount of data on the virtual cartridge in megabytes.

Word 7

The amount of disk space used by this virtual cartridge in megabytes.

Word 8

The virtual ID assigned to the virtual cartridge by the VTL.

Word 9

The amount of data in compressed bytes on this virtual cartridge in megabytes.

Word 10

The timestamp in YYYY/MM/DD hh:mm:ss with hh being 00 to 23 format when the virtual cartridge was created.

Word 11

The timestamp in YYYY/MM/DD hh:mm:ss with hh being 00 to 23 format when the virtual cartridge was last modified.

Word 12

The virtual ID of the replica of this cartridge on a remote VTL if it has been replicated.

Results: Information showing the current status of the referenced drive unit is returned. The drive status is checked.

Possible errors: 15, 16

SN_STATUS Procedure

This procedure returns information about a specified tape library bar code in a VTL. This procedure is similar to SLOT_STATUS except that VTLINFO request is always assumed. If the library is offline or it is a CONNECTION= NONE library then words 0-3 will be zero. If the library is not a VTL or the connection to the Agent is unavailable then words 4-15 will be zero.

```
BOOLEAN PROCEDURE SN_STATUS(LIBRARY_ID, SN, STATUS_DATA);
VALUE LIBRARY_ID, SN;
REAL
     LIBRARY_ID         % LIBRARY ID (HANDLE) THAT HAS SN
    ,SN                 % BARCODE TO RETURN STATUS FOR
    ;
EBCDIC ARRAY
     STATUS_DATA[0]     % STATUS INFO RETURNED FOR SN
    ;
     LIBRARY LIBRARYSUPPORT;

DEFINE % LAYOUT OF STATUS_DATA ARRAY RETURNED BY SLOT_STATUS
     SN_STAT_NUMMBER  = STATUS_DATAW[0].[47:16]# % ELEMENT NUM
    ,SN_STAT_SLOTNUM  = STATUS_DATAW[0].[31:16]# % EXTERNAL NUM
    ,SN_STAT_DD       = STATUS_DATAW[0].[8: 1]# % SLOT IN DD
    ,SN_STAT_VIAVTL   = STATUS_DATAW[0].[7: 1]# % SLOT IN BACKEND
    ,SN_STAT_VTL      = STATUS_DATAW[0].[6: 1]# % SLOT IN VIRTUAL
    ,SN_STAT_LA       = STATUS_DATAW[0].[5: 1]# % LIBA CONTROLLED
    ,SN_STAT_CSCA     = STATUS_DATAW[0].[4: 1]# % CSC-A CONTROLL
    ,SN_STAT_ACCESS   = STATUS_DATAW[0].[3: 1]# % ACCESS ALLOWED
    ,SN_STAT_EXCEPT   = STATUS_DATAW[0].[2: 1]# % ABNORMAL STATE
    ,SN_STAT_SIDE2    = STATUS_DATAW[0].[1: 1]# % SIDE 2 LOADED
    ,SN_STAT_FULL     = STATUS_DATAW[0].[0: 1]# % TAPE IN SLOT
    ,SN_STAT_VOL_TAG  = STATUS_DATAW[1]#        % BAR CODE LABEL
    ,SN_STAT_VOL_TAG2 = STATUS_DATAW[2]#        % SN SIDE 2
    ,SN_STAT_UNIT     = STATUS_DATAW[3]#        % USING DRIVE
    ,SN_STAT_NOCONN   = STATUS_DATAW[4].[8: 1]# % CONNECTION=NONE
    ,SN_STAT_COMPRESS = STATUS_DATAW[4].[7: 1]# % COMPRESSION SET
    ,SN_STAT_CACHING  = STATUS_DATAW[4].[6: 1]# %
    ,SN_STAT_DUPL     = STATUS_DATAW[4].[5: 1]# %
    ,SN_STAT_REPL     = STATUS_DATAW[4].[4: 1]# %
    ,SN_STAT_ARCHIVE  = STATUS_DATAW[4].[3: 1]# %
    ,SN_STAT_READONLY = STATUS_DATAW[4].[2: 1]# % CART READ ONLY
    ,SN_STAT_COD      = STATUS_DATAW[4].[1: 1]# % COD IS SET
    ,SN_STAT_VTLINFO  = STATUS_DATAW[4].[0: 1]# % VTL INFO VALID
    ,SN_STAT_CAPACITY = STATUS_DATAW[5]#        % CART SIZE MB
    ,SN_STAT_CARTSIZE = STATUS_DATAW[6]#        % MB USED BY CART
    ,SN_STAT_DATASIZE = STATUS_DATAW[7]#        % MB USED BY DATA
    ,SN_STAT_VID      = STATUS_DATAW[8]#        % VTL VID OF CART
    ,SN_STAT_USEDSIZE = STATUS_DATAW[9]#        % MB USED BY USER
    ,SN_STAT_CTIME    = STATUS_DATAW[10]#       % VTL CREATE TIME
    ,SN_STAT_MTIME    = STATUS_DATAW[11]#       % VTL MODIFY TIME
    ,SN_STAT_REPVID   = STATUS_DATAW[12]#       % REPLICA VID
    ,SN_STAT_LOCTYPE  = STATUS_DATAW[13]#       % 1=SLT,2=DRV,3=VLT
    ,SN_STAT_LOCID    = STATUS_DATAW[14]#       % SLT # OR DRV ELEM
    ,SN_STAT_VERSION  = STATUS_DATAW[15]#       % VER OF WRDS 4-N
        ,SN_STAT_CURRENT_VERSION = 2# % LOCID FOR SLOT TO 1 REL
    ,SN_STAT_SZV      = 16# % WORDS
;

ARRAY STATUS_DATAW[0] = STATUS_DATA;
```

Function: Returns current status information about a library bar code.

Usage: This procedure will return the current status information for a library bar code. The procedure is used to diagnose errors and create information displays for the user.

Parameters:

LIBRARY_ID — (Input) the ID of the library that contains the bar code.

SN — (Input) the bar code to return status for.

STATUS_DATA — (Output) the status information is returned as a set of fields. These fields are defined as follows:

Word 0

[47:16]	the element number of the slot as defined by the library hardware
[31:16]	the slot number
[08:01] = 1	if the slot is in Data Domain system
[07:01] = 1	if the slot is in back-end library
[06:01] = 1	if the slot is in virtual library
[05:01] = 1	if the slot is controlled by StorageTek Library Attach.
[04:01] = 1	if the slot is controlled by Unisys CSC-A.
[03:01] = 1	if the slot is accessible (can be used).
[02:01] = 1	if the slot has an error reported against it.
[01:01] = 1	if the media has 2 sides and the second side is in use in the drive unit.
[00:01] = 1	if the slot has a cartridge assigned to it.

All other fields in this word are reserved for future use.

Word 1

If the slot has a cartridge assigned to it, this field contains the right most 6 characters of the bar code label of the cartridge. If the library does not have a barcode reader, the serial number of the cartridge is returned if known.

Word 2

If the slot has a cartridge assigned to it, and the media has 2 sides, this is the serial number of the second side otherwise it is zero.

Word 3

If the slot has a cartridge assigned to it and the cartridge is in use by a drive, the drives' unit number is stored here.

Word 4

[08:01] = 1 if library is CONNECTION=NONE

[07:01] = 1 if the tape is set for compression

[06:01] = 1 if tape is cached.

[05:01] = 1 if tape is duplicated.

[04:01] = 1 if tape is replicated (not Remote Copy).

[03:01] = 1 if tape is archived.

[02:01] = 1 if the is read-only

[01:01] = 1 if the tape has Capacity-On-Demand set.

[00:01] = 1 VTL tape info valid.

All other fields in this word are reserved for future use.

Word 5

The raw capacity of the virtual cartridge in megabytes.

Word 6

The amount of data on the virtual cartridge in megabytes.

Word 7

The amount of disk space used by this virtual cartridge in megabytes.

Word 8

The virtual ID assigned to the virtual cartridge by the VTL.

Word 9

The amount of data in compressed bytes on this virtual cartridge in megabytes.

Word 10

The timestamp in YYYY/MM/DD hh:mm:ss with hh being 00 to 23 format when the virtual cartridge was created.

Word 11

The timestamp in YYYY/MM/DD hh:mm:ss with hh being 00 to 23 format when the virtual cartridge was last modified.

Word 12

The virtual ID of the replica of this cartridge on a remote VTL if it has been replicated.

Word 13

The type of location in word 14 defined as follows; 0=no information, 1=slot number, 2=tape drive element number, 3=in vault (words 14 will be zero).

Word 14

The location value as specified by the location type in word 13.

Word 15

The version of the information in words 4 thru N. Current version is 2.

Results: Information showing the current status of the referenced bar code is returned. The slot status is checked.

Possible errors: 15, 16

VTL_COMMAND Procedure

```
DEFINE % LAYOUT OF CMND WORDS AND CMND_RESP ARRAY
      SUBFUNCTION       =   CMND .[31:16]# % FUNCTION SUBTYPE
          % SUBFUNCTIONS OF VTL MOVE AND VTL BEMOVE
          ,MT_SLOTV     = 1# % MOVE FROM DRIVE TO SLOT (UNLOAD)
          ,SLOT_MTV     = 2# % MOVE FROM SLOT TO DRIVE (MOUNT)
          ,SLOT_SLOTV   = 3# % MOVE FROM SLOT TO SLOT
          ,VAULT_LIBV   = 4# % MOVE FROM VAULT TO LIBRARY
          ,LIB_VAULTV   = 5# % MOVE FROM LIBRARY TO VAULT (EXPORT)
          ,IE_LIBV      = 6# % MOVE FROM DOOR TO LIBRARY
          ,LIB_IEV      = 7# % MOVE FROM LIBRARY TO DOOR (EXPORT)
          % SUBFUNCTIONS OF VTL QUERY
          ,STACKED_DIRV = 1# % STACKED TAPE DIRECTORY
          ,VAULT_INVENV = 2# % UPDATE VAULT INVENTORY
          ,POLICY_LISTV = 3# % DEDUPE POLICY LIST
          % SUBFUNCTIONS OF VTL TAPE
          ,TAPE_ADDV    = 1# % ADD VIRTUAL TAPE
          ,TAPE_DELV    = 2# % DELETE VIRTUAL TAPE
          ,TAPE_MODV    = 3# % MODIFY VIRTUAL TAPE
          ,TAPE_RENAMEV = 4# % RENAME VIRTUAL TAPE
          % SUBFUNCTIONS OF VTL MSGSV
          ,GET_MSGSV    = 1#
          ,SET_MSGSV    = 2#
     ,FUNCTION          =   CMND .[15:16]# % WHAT IS WANTED DONE
          ,VTL_MOVEV    = 1# % MOVE A TAPE VIA AGENT
          ,VTL_STACKV   = 2# % STACK VTL TAPES
          ,VTL_UNSTACKV = 3# % UNSTACK VTL TAPES
          ,VTL_BEMOVEV  = 4# % MOVE A TAPE IN A BACK END LIBRARY
          ,VTL_EXPORTV  = 5# % COPY VIRTUAL TAPE TO PHYSICAL
          ,VTL_IMPORTV  = 6# % COPY PHYSICAL TAPE TO VIRTUAL
          ,VTL_REMCOPYV = 7# % REPLICATE VIRTUAL TAPE TO OTHER VTL
          ,VTL_QUERYV   = 8# % VARIOUS VTL QUERIES
          ,VTL_TAPEV    = 9# % VIRTUAL TAPE ADD/DELETE
          ,VTL_MSGSV    = 10# % AGENT/VTL MESSAGE FILTERING
    %% VTL_MOVEV/VTL_BEMOVEV CMND_RESP LAYOUT
     ,BARCODE           = CMND_RESP[0]# % TAPE TO MOVE
     ,SOURCELOC         = CMND_RESP[1]# % SOURCE MT #, SLOT #
     ,DESTLOC           = CMND_RESP[2]# % DESTINATION MT #, SLOT #
    %% VTL_STACKV CMND_RESP LAYOUT
     ,OPTIONS           = CMND_RESP[0]# % STACK OPTIONS
          ,MOVE_TAPE    = B(CMND_RESP[0]).[0:1]# % MOVE vs COPY
          ,FORCE_TAPE   = B(CMND_RESP[0]).[1:1]# % FORCE TBD TAPE
          ,EJECT_TAPE   = B(CMND_RESP[0]).[2:1]# % EJECT TAPE
          ,APPEND_TAPE  = B(CMND_RESP[0]).[3:1]# % APPEND TAPE
          ,DUP_TAPE     = B(CMND_RESP[0]).[4:1]# % N/A
          ,REMCOPY      = B(CMND_RESP[0]).[5:1]# % N/A
          ,REMTOVTL     = B(CMND_RESP[0]).[6:1]# % N/A
          ,REPLICATE    = B(CMND_RESP[0]).[7:1]# % N/A
          ,CHANGE_RO    = B(CMND_RESP[0]).[8:1]# % CHANGE READONLY
          ,READONLY     = B(CMND_RESP[0]).[9:1]# % READONLY STATE
          ,COD_OFF      = B(CMND_RESP[0]).[10:1]# % CHANGE COD OFF
          ,FORCE_BC     = B(CMND_RESP[0]).[11:1]# % FORCE NEWBC
          ,REP_OFF      = B(CMND_RESP[0]).[12:1]# % FORCE REPL OFF
          ,COMPRESS     = B(CMND_RESP[0]).[13:1]# % COMPRESS TAPE C
          ,ENCRYPT      = B(CMND_RESP[0]).[14:1]# % ENCRYPT TAPE CO
          ,DEDUPE       = CMND_RESP[0].[16:2]# % DEDUPE TAPE
             ,UNSPECIFIED=0#
             ,DEDUPE_ON = 1#
             ,DEDUPE_OFF= 2#
     ,DEST_SN           = CMND_RESP[1]# % BAR CODE OF DEST TAPE
     ,DELETE_DELAY      = CMND_RESP[2]# % DAYS TO DELAY AFTER MOVE
     ,KEY_NAME_SZ       = CMND_RESP[3].[47:8]# % KEY NAME BYTES
```

```
 ,KEY_NAME           = P(CMND_RESP[3])+1# % VTL ENCRYPT KEY NAME
 ,PASSWORD_SZ        = CMND_RESP[9].[47:8]# % PASSWORD BYTES
 ,PASSWORD           = P(CMND_RESP[9])+1# % ENCRYPT KEY PASSWORD
 ,SOURCE_COUNT       = CMND_RESP[12]# % NUMBER OF TAPES TO STACK
 ,SOURCE_SN(I)       = P(CMND_RESP[12+I])# % SNS TAPES TO STACK
%% VTL_UNSTACKV CMND_RESP LAYOUT
 %OPTIONS            = CMND_RESP[0]# % UNSTACK OPTIONS
 ,UCOUNT             = CMND_RESP[1]# % NUM OF TAPES TO UNSTACK
 %KEY_NAME_SZ        = CMND_RESP[3].[47:8]# % KEY NAME BYTES
 %KEY_NAME           = P(CMND_RESP[3])+1# % VTL ENCRYPT KEY NAME
 %PASSWORD_SZ        = CMND_RESP[9].[47:8]# % PASSWORD BYTES
 %PASSWORD           = P(CMND_RESP[9])+1# % ENCRYPT KEY PASSWORD
 ,PLIBVID            = CMND_RESP[12]# % VID OF PHYSICAL LIBRARY
 ,PBARCODE           = CMND_RESP[13]# % BAR CODE PHYSICAL TAPE
 ,UOLDBARCODE(I)     = CMND_RESP[11+(I*3)]# % BC TO BE UNSTACKED
 ,UNEWBARCODE(I)     = CMND_RESP[12+(I*3)]# % BC OF UNSTACK TAPE
 ,UNEWSLOT(I)        = CMND_RESP[13+(I*3)]# % SLOT TO PUT IT IN
%% VTL_EXPORTV CMND_RESP LAYOUT
 %OPTIONS            = CMND_RESP[0]# % EXPORT OPTIONS
      %MOVE_TAPE     = B(CMND_RESP[0]).[0:1]# % MOVE VS COPY
      %FORCE_TAPE    = B(CMND_RESP[0]).[1:1]# % FORCE TBD TAPE
      %EJECT_TAPE    = B(CMND_RESP[0]).[2:1]# % EJECT TAPE
      %DUP_TAPE      = B(CMND_RESP[0]).[4:1]# % DUP EXPORTED
 ,VBARCODE           = CMND_RESP[1]# % TAPE TO EXPORT
 %DELETE_DELAY       = CMND_RESP[2]# % DAYS TO DELAY AFTER MOVE
 %KEY_NAME_SZ        = CMND_RESP[3].[47:8]# % KEY NAME BYTES
 %KEY_NAME           = P(CMND_RESP[3])+1# % VTL ENCRYPT KEY NAME
 %PASSWORD_SZ        = CMND_RESP[9].[47:8]# % PASSWORD BYTES
 %PASSWORD           = P(CMND_RESP[9])+1# % ENCRYPT KEY PASSWORD
 %PLIBVID            = CMND_RESP[12]# % VID OF PHYSICAL LIBRARY
 %PBARCODE           = CMND_RESP[13]# % BAR CODE PHYSICAL TAPE
 ,DUP_COPIES         = CMND_RESP[14]# % IF DUP_TAPE, # OF COPIES
%% VTL_IMPORTV CMND_RESP LAYOUT
 %OPTIONS            = CMND_RESP[0]# % IMPORT OPTIONS
 %VBARCODE           = CMND_RESP[1]# % BAR CODE TO CREATE
 ,VSLOT              = CMND_RESP[2]# % SLOT TO CREATE TAPE IN
 %KEY_NAME_SZ        = CMND_RESP[3].[47:8]# % KEY NAME BYTES
 %KEY_NAME           = P(CMND_RESP[3])+1# % VTL ENCRYPT KEY NAME
 %PASSWORD_SZ        = CMND_RESP[9].[47:8]# % PASSWORD BYTES
 %PASSWORD           = P(CMND_RESP[9])+1# % ENCRYPT KEY PASSWORD
 %PLIBVID            = CMND_RESP[12]# % VID OF PHYSICAL LIBRARY
 %PBARCODE           = CMND_RESP[13]# % BAR CODE PHYSICAL TAPE
 ,IMPORT_MODE        = CMND_RESP[14]# % SEE BELOW
      ,COPY_MODEV    = 0# % NORMAL IMPORT (DEFAULT)
      ,RECYCLE_MODEV= 1# % RECYCLES PHYSICAL TAPE
      ,DIRECT_MODEV = 2# % LINKS PHYSICAL TO VIRTUAL
%% VTL_REMCOPYV CMND_RESP LAYOUT
 %OPTIONS            = CMND_RESP[0]# % REPLICATION OPTIONS
      %FORCE_TAPE    = B(CMND_RESP[0]).[1:1]# % FORCE TBD TAPE
      %REMCOPY       = B(CMND_RESP[0]).[5:1]# % FALSE=LOCAL COPY
      %REMTOVTL      = B(CMND_RESP[0]).[6:1]# % FALSE=IN VAULT
      %REP_OFF       = B(CMND_RESP[0]).[12:1]# % FORCE REPL OFF
      %COMPRESS      = B(CMND_RESP[0]).[13:1]# % COMP TAPE COPY
      %ENCRYPT       = B(CMND_RESP[0]).[14:1]# % ENCRY TAPE COPY
 %VBARCODE           = CMND_RESP[1]# % TAPE TO BE REPLICATED
 ,REMVID             = CMND_RESP[2]# % VTL VID IF REMTOVTL
 ,REMSVR_SZ          = CMND_RESP[3].[47:8]# % SERVER NAME <=39
 ,REMSVR             = P(CMND_RESP[3])+1# % SERVER NAME OR ADDR
 ,REMUSERID_SZ       = CMND_RESP[10].[47:8]# % USER ID <=64
 ,REMUSERID          = P(CMND_RESP[10])+1# % REMOTE VTL USER ID
 ,REMPASSWORD_SZ     = CMND_RESP[21].[47:8]# % PASSWORD <=64
 ,REMPASSWORD        = P(CMND_RESP[21])+1# % REMOTE VTL PASSWORD
 ,REMNAME_SZ         = CMND_RESP[32].[47:8]# % REMOTE NAME <=64
 ,REMNAME            = P(CMND_RESP[32])+1# % NAME OF REMOTE COPY
```

```
      %% VTL_QUERYV CMND_RESP LAYOUT
       %OPTIONS           = CMND_RESP[0]# % QUERY OPTIONS
       % STACK TAPE DIRECTORY QUERY
       %PLIBVID           = CMND_RESP[12]# % VID OF PHYSICAL LIBRARY
       %PBARCODE          = CMND_RESP[13]# % BAR CODE OF STACKED TAPE
       ,SCOUNT            = CMND_RESP[14]# % NUMBER OF ITEMS STACKED
       ,SBARCODE(I)       = CMND_RESP[14+I]# % BARCODE OF STACK ITEM
       ,STACK_DIR_SZV     = (15+1000)# % SIZE OF STACKED DIR RESPONSE
      %% VTL_TAPEV-TAPE_ADDV CMND_RESP LAYOUT
       %OPTIONS           = CMND_RESP[0]# % TAPE OPTIONS
            %EJECT_TAPE   = B(CMND_RESP[0]).[2:1]# % EJECT ON EXPORT
            %REPLICATE    = B(CMND_RESP[0]).[7:1]# % ENBL/DSBL REPL
            %DEDUPE       = CMND_RESP[0].[16:2]# % DEDUPE TAPE
       %VBARCODE          = CMND_RESP[1]# % BC TO ADD IF ADDCOUNT = 0
       ,ADDCOUNT          = CMND_RESP[2]# % # TAPES TO ADD
       ,INITSIZE          = CMND_RESP[3]# % INIT SIZE IN GB
       %PLIBVID           = CMND_RESP[12]# % VID OF PHYSICAL LIBRARY
       %PBARCODE          = CMND_RESP[13]# % BAR CODE OF PHYS TAPE
       %REMSVR_SZ         = CMND_RESP[3].[47:8]#%SRVR NAME BYTES <=39
       %REMSVR            = P(CMND_RESP[3])+1#%SRVR IP NAME OR ADDR
       ,POLICY_SZ         = CMND_RESP[10].[47:8]#%DDUP POL BYTES <=64
       ,POLICY            = P(CMND_RESP[10])+1# % DEDUPE POLICY NAME
      %% VTL_TAPEV-TAPE_DELV CMND_RESP LAYOUT
       %OPTIONS           = CMND_RESP[0]# % TAPE OPTIONS
            %FORCE_TAPE   = B(CMND_RESP[0]).[1:1]# % FORCE TBD TAPE
       %VBARCODE          = CMND_RESP[1]# % BC TO DELETE
       ,TAPEVID           = CMND_RESP[2]# % VID TO DELETE IF BC EMPTY
       %OPTIONS           = CMND_RESP[0]# % TAPE OPTIONS
            %CHANGE_RO    = B(CMND_RESP[0]).[8:1]# % CHANGE READONLY
            %READONLY     = B(CMND_RESP[0]).[9:1]# % READONLY STATE
            %COD_OFF      = B(CMND_RESP[0]).[10:1]# % CHANGE COD OFF
            %FORCE_BC     = B(CMND_RESP[0]).[11:1]# % FORCE NEWBC
       %VBARCODE          = CMND_RESP[1]# % TAPE TO MODIFY
       ,NEWBARCODE        = CMND_RESP[13]# % BARCODE TO CHANGE TO
      %% VTL_TAPEV-TAPE_RENAMEV CMND_RESP LAYOUT
       %OPTIONS           = CMND_RESP[0]# % TAPE OPTIONS
       %VBARCODE          = CMND_RESP[1]# % TAPE TO MODIFY
       ,TNAME_SZ          = CMND_RESP[32].[47:8]# %TAPE NAME SZ <=32
       ,TNAME             = P(CMND_RESP[32])+1# % TAPE NAME TO SET
      %% VTL_MSGSV CMND_RESP LAYOUT
       ,MSG_CONTROL       = CMND_RESP[0]#
            ,MSG_COUNTF   = [19:20]# % NUMBER OF MSG WORDS IN ARRAY
            ,MSG_ALL_TYPEF= [23:4]# % MSG TYPE FOR ALL MSG SETTING
            ,MSG_ALL_SETF = [24:1]# % ALL MSG HAS BEEN SPECIFIED
       ,MSG_WORD(X)       = CMND_RESP[X]#% WRD 4 EACH VTL MSG DEFINED
            ,MSG_NUMBERF  = [19:20]# % VTL MSG NUMBER BEING REDEFINED
            ,MSG_TYPEF    = [23:4]# % MSG TYPE MSG BEING REDEFINED TO
            % MSG TYPES
           ,MSG_DEFAULTV = 0#
           ,MSG_INFOV    = 1#
           ,MSG_WARNV    = 2#
           ,MSG_ERRORV   = 3#
           ,MSG_CRITICALV= 4#
           ,MSG_IGNOREV  = 5#
       ,CMND_RESP_MIN_SZV = 43# % WORDS
       ;

BOOLEAN PROCEDURE VTL_COMMAND(LIBRARY_ID, CMND, CMND_RESP);
VALUE LIBRARY_ID, CMND;
REAL
     LIBRARY_ID         % ID OF VTL LIBRARY TO SEND AGENT COMMAND
    ,CMND               % VTL/AGENT COMMAND
    ;
ARRAY CMND_RESP[0];     % COMMAND/RESPONSE ARRAY, VARIES BY CMND
```

```
LIBRARY LIBRARYSUPPORT;
```

Function: This procedure passes a request to the VTL Agent via the VTLSUPPORT library.

Usage: This procedure is used to control various functions of VTL virtual and backend physical libraries that are not available via the normal library interface. Any unused or undefined bits or words in the CMND and CMND_RESP parameters are reserved and must be set to zero. The details of these functions can be found in the Virtual Tape Library User Guide.

Parameters:

LIBRARY_ID	(Input) the library id of the virtual library that is to process the command.
CMND	(Input) the type and subtype of the command to be performed: [15:16] command type (FUNCTION) that is to be performed [31:16] command subtype (SUBFUNCTION) that is to be performed
CMND_RESP	(Input/Output) input parameters and output information for VTL commands as defined below using the above defines

FUNCTION 1 (VTL_MOVEV)

Moves a virtual cartridge within a virtual library. Only SUBFUNCTON 4 (VAULT_LIBV) should be used. The VAULT_LIBV sub function moves a virtual cartridge from the virtual vault to a virtual library. All other movement should be done with the normal LIBRARY_MOVE interface.

BARCODE – barcode of tape to be moved from vault to library defined by LIBRARY_ID
SOURCELOC – not used
DESTLOC – not used

FUNCTION 2 (VTL_STACKV)[1]

Copies one or more virtual cartridges onto a single physical tape using the VTL tape stacking feature.

MOVE_TAPE – if TRUE the virtual cartridge is deleted from the VTL after the stack/append operation has completed (see DELETE_DELAY)
FORCE_TAPE – must be true if stacking/appending a virtual cartridge that is scheduled to be deleted
EJECT_TAPE – if TRUE eject physical tape from library when operation complete
APPEND_TAPE – if TRUE operation is appended to a previously created stacked tape
SOURCE_COUNT – number of tapes to be stacked or appended (maximum 1000)
SOURCE_SN(I) – bar codes of tapes to be stacked or appended
DEST_SN – bar code of physical tape for stack/append operation
DELETE_DELAY – time on days to delay cartridge delete if MOVE_TAPE = TRUE (default 365)
KEY_NAME_SZ – length in bytes of the VTL encryption key name (maximum 32)

KEY_NAME – name of the VTL encryption key to be used if the output tape is to be encrypted. Must be predefined at VTL
PASSWORD_SZ – length in bytes of password for encryption key (maximum 16)
PASSWORD – password to access encryption key

FUNCTION 3 (VTL_UNSTACKV)[1]

Copies one or more stacked volumes from a physical stacked tape to virtual volumes.

PLIBVID – virtual ID of physical library where stacked tape resides
PBARCODE – bar code of the physical stacked tape
UCOUNT – number of tapes to be unstacked (maximum 1000)
UOLDBARCODE(I) – bar codes of tapes to be unstacked
UNEWBARCODE(I) – bar codes to create unstacked tapes as (must not already exist)
UNEWSLOT(I) – virtual library slot to place unstacked tape in (must be empty)
KEY_NAME_SZ – length in bytes of the VTL encryption key name (maximum 32)
KEY_NAME – name of the VTL encryption key to be used if the input tape was encrypted. Must be predefined at VTL
PASSWORD_SZ – length in bytes of password for encryption key (maximum 16)
PASSWORD – password to access encryption key

FUNCTION 4 (VTL_BEMOVEV)[1]

Moves a physical cartridge in a VTL backend library. This function should not be accessed directly but rather these functions are available through the LIBRARY_MOVE procedure.

FUNCTION 5 (VTL_EXPORTV)[1]

Copies a single virtual cartridge onto a single physical cartridge.

MOVE_TAPE – if TRUE the virtual cartridge is deleted from the VTL after the copy operation has completed (see DELETE_DELAY)
FORCE_TAPE – must be true if copying a virtual cartridge that is scheduled to be deleted
EJECT_TAPE – if TRUE eject physical tape from library when operation complete
DUP_TAPE – if TRUE make duplicate copies of tape (see DUP_COPIES)
VBARCODE – bar code of virtual tape to be copied
PLIBVID – virtual ID of physical library where output tape resides
PBARCODE – bar code of physical tape to be written
DELETE_DELAY – time on days to delay cartridge delete if MOVE_TAPE = TRUE (default 365)
DUP_COPIES – number of copies of tape to make if DUP_TAPE is TRUE
KEY_NAME_SZ – length in bytes of the VTL encryption key name (maximum 32)
KEY_NAME – name of the VTL encryption key to be used if the output tape is to be encrypted. Must be predefined at VTL
PASSWORD_SZ – length in bytes of password for encryption key (maximum 16)
PASSWORD – password to access encryption key

FUNCTION 6 (VTL_IMPORTV)[1]

Copies a single physical cartridge to a single virtual cartridge.

PLIBVID – virtual ID of physical library where physical tape resides
PBARCODE – bar code of the physical tape to be copied

VBARCODE – bar code of virtual tape to be created (must not exist)
VSLOT – virtual library slot to place created tape in (must be empty)
KEY_NAME_SZ – length in bytes of the VTL encryption key name (maximum 32)
KEY_NAME – name of the VTL encryption key to be used if the input tape was encrypted. Must be predefined at VTL
PASSWORD_SZ – length in bytes of password for encryption key (maximum 16)
PASSWORD – password to access encryption key

FUNCTION 7 (VTL_REMCOPYV)[2]

Copies a single virtual cartridge to another VTL using the Remote Copy feature.

FORCE_TAPE – must be true if copying a virtual cartridge that is scheduled to be deleted
REMCOPY – if TRUE copy is to remote VTL otherwise a local copy
REMTOVTL – if TRUE copy is placed in a virtual library otherwise put in vault
REP_OFF – if TRUE auto replication is turned off before Remote Copy is started
COMPRESS – if TRUE compression is turned on for Remote Copy request – if neither COMPRESS or ENCRYPT are specified compression on and encryption off is the default
ENCRYPT – if TRUE encryption is turned on for Remote Copy request – Note that turning encryption on may degrade the performance of the Remote Copy
VBARCODE – bar code of virtual tape to be copied
REMVID – virtual ID of virtual library on remote system (if REMTOVTL is TRUE)
REMSVR_SZ – length in bytes of remote VTL name (maximum 39)
REMSVR – name of remote VTL system (network name or IP address)
REMNAME_SZ – length in bytes of copied tape name (maximum 64) (optional)
REMNAME – name of copy on remote system (optional)
REMUSERID_SZ – length in bytes of the user ID for the remote VTL (maximum 64)
REMUSERID – user ID to access remote VTL
REMPASSWORD_SZ – length in bytes of password for remote VTL (maximum 64)
REMPASSWORD – password to access remote VTL

FUNCTION 8 (VTL_QUERYV)[1]

Requests information from a VTL. Currently only the SUBFUNCTION of STACKED_DIRV is defined that returns a list of tapes on a stacked tape.

SUBFUNCTION – STACKED_DIRV – return list of stacked tapes on volume
PLIBVID – virtual ID of physical library where physical tape resides
PBARCODE – bar code of the physical stacked tape
SCOUNT – number of tapes stacked on designated stacked tape
SBARCODE(I) – bar codes of tapes on stacked tape

FUNCTION 9 (VTL_TAPEV)[1]

Adds, deletes or modifies virtual cartridges in a virtual library. Created cartridges will inherit the attributes defined for the library the cartridges are created in.

SUBFUNCTION – TAPE_ADDV – creates virtual cartridges in a virtual library
VBARCODE – bar code of cartridge to be created, must be empty if ADDCOUNT used
ADDCOUNT – number of tapes to create, bar codes selected by VTL based on library attributes

SUBFUNCTION – TAPE_DELV – removes a virtual cartridge in a virtual library
FORCE_TAPE – must be true if replication is set for cartridge to be deleted
VBARCODE – bar code of cartridge to be deleted
TAPEVID – if VBARCODE is null then this is the VID of the tape to delete

SUBFUNCTION – TAPE_MODV – changes a virtual cartridge attributes
CHANGE_RO – if TRUE then the READONLY has a valid read-only state
READONLY – if TRUE then set the read-only attribute otherwise reset read-only
COD_OFF – if TRUE then turns the Capacity On Demand (COD) feature off for the cartridge. WARING: cannot not be set back on!
FORCE_BC – Used with NEWBARCODE, if TRUE allows a new bar code that does not match the attributes specified for the library
VBARCODE – bar code of cartridge to be modified
NEWBARCODE – new bar code to be applied to a virtual cartridge. The bar code must match the attributes defined for the library unless FORCE_BC is set.

SUBFUNCTION – TAPE_RENAMEV – changes the name of the virtual tape displayed at the VTL GUI console
VBARCODE – bar code of the virtual cartridge to have its name changed
TNAME_SZ – number of characters in the new name string
TNAME – the new name string

Results: Results will vary by VTL request. If the procedure returns TRUE, an error value can be found in bits [26:10] of the result. If the error is Command error (8) then the CMND_RESP array will contain a text message from the VTL that is terminated by a null. NOTE: most of these functions start an independent process on the VTL. The results returned are only the results of starting (or queuing) that process. Messages from the VTL need to be monitored to determine the completion and results of the actual process.

Possible errors: All

[1]This function is not supported by Data Domain VTL systems.

[2]The Remote Copy feature is not supported by Data Domain systems but is simulated for compatibility with DSI VTLs. A request for a Remote Copy causes DDSUPPORT to monitor the replication status of the DD VTL pool and send a completion event when the replication status shows that there is nothing replicating in that pool.

Document Evaluation Form

DSI is interested in receiving your comments and suggestions regarding this document. Comments will be utilized in subsequent revisions to improve this document.

Manual Title:

(Addendum)
Cartridge Library Installation Guide
for
Unisys MCP Systems

Version No: **10.070**
Date: **June 2023**

Please check type of suggestion:

☐ Addition ☐ Deletion ☐ Revision ☐ Error

Comments / Suggestion:

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

From:

Name: _____

Title: _____

Company:_____

Address: _____

City/ST/ZIP:_____

Phone: (_____)_____-_____ Date: _____/_____/_____

Remove form and mail preaddressed overleaf, or FAX to:

Dynamic Solutions International (DSI)
Product Development Group

FAX Number: (303)754-2066

FOLD HERE AND TAPE

PUT
STAMP
HERE

Dynamic Solutions International (DSI)
8744 Lucent Blvd. Suite 106
Highlands Ranch 80129
U.S.A.

Attn: Product Development Group