Video Server Toolkit
Installation and System Administration Guide

Document Number 007-3622-003

CONTRIBUTORS

Written by Carolyn Curtis and George Eckel
Illustrated by Dany Galgani
Production by Heather Hermstad
Engineering contributions by Rick Reed, Achuith Bhandarkar, Ashok Yerneni, Prince Kohli, Alan Saichek, Brad Thayer, Parkson Wong, Kurt Merriweather, Mike Moskowitz, Hilmi Ortadeveci, Manuel Ruiz, Aman Singla, and Ellen Evans
St. Peter's Basilica image courtesy of ENEL SpA and InfoByte SpA. Disk Thrower image courtesy of Xavier Berenguer, Animatica.

Video Server Toolkit Installation and System Administration Guide
Document Number 007-3622-003

Contents

List of Figures

List of Tables

About This Guide

Video Server Toolkit (VST) is the SGI broadcast-quality video playback, edit, and record engine that unifies SGI Origin servers and digital media components. VST allows application developers to create powerful, high-performance solutions for broadcast playout and editing. This tool enables (supported) automation systems and developer applications to trigger video for playout through (supported) video output devices on an SGI workstation or server.

This manual applies only to revision 1.2 of VST, which runs on Origin 200 GIGAchannel system running IRIX 6.5.4 or later. This revision can also run on the Silicon Graphics O2 workstation for development purposes only.

Note: The VST product was formerly called VCP-Recorder.

For more information about using VST, see the *Video Server Toolkit Developer's Guide*.

This manual describes how to set up the system hardware, how to configure filesystems for VST, how to install and configure the VST software, and how to monitor the VST system.

Audience

This document is written for VST system integrators and administrators responsible for installing and monitoring the running of VST. It is to be used in conjunction with the *Video Server Toolkit Administrator's Guide*. It presumes knowledge of SGI XFS filesystems.

Structure of This Document

This document contains three parts:

- Chapter 1, “Setting Up the Hardware and the System,” explains how to set up your system so that VST can run on it.
- Chapter 2, “Setting Up Filesystems for Video Server Toolkit,” describes how to create XFS filesystems for Video Server Toolkit and for the VST clip cache, which holds the VST media assets.
- Chapter 3, “Installing Video Server Toolkit,” provides a step-by-step description of installing the VST software.
- Chapter 4, “Configuring Video Server Toolkit,” describes the configuration files and settings for VST.
- Chapter 5, “Configuring for Redundancy,” describes how to install IRIS FailSafe 1.2, which is a monitoring system that switches system functions from a primary to a secondary server when the primary server fails, and how to set up one or more servers for clip mirroring to reflect the state of the primary server’s clip cache.
- Chapter 6, “Configuring and Using External Devices,” explains how to install and configure hardware devices and software protocols to work with VST, such as the Sony (P2) protocol, the Louth Automation Controller, and Diaquest software.
- Chapter 7, “Configuring and Using Vela and DVB-ASI Devices,” explains how to install and configure the Vela Research MPEG-2 Decoder, and the DVB-ASI adapter board to work with VST.
- Chapter 8, “Running VST,” explains starting and stopping VST, using log files and directories, and monitoring the system.
- Chapter 9, “Troubleshooting,” lists an assortment of common problems and their solutions.
- Appendix A, “Video Server Toolkit Control Summary,” describes the variables called controls that help configure the VST system.
- Appendix B, “RS-422 Pinouts,” describes the pinouts for the RS-422 mini-DIN8 to DB-9 adapters for SGI and Apple Macintosh cables.
- Appendix C, “Video Server Toolkit Man Pages,” summarizes VST man pages.

An index completes this guide.

HTML versions of the VST guides are installed at these URLs:

- http://*hostname.domain*/VST/VST_AG
- http://*hostname.domain*/VST/VST_DG

Related Documentation

The *Video Server Toolkit Developer's Guide* (part number 007-3620-*nnn*) contains information about using VST to play and record digital media data; and store the data in, and retrieve it from, an *archive system*. This manual is included on the VST CD.

Refer to the latest versions of the following documents for supplementary information:

- *IRIX Admin: Software Installation and Licensing* (part number 007-1364-*nnn*) for information about installing software that runs under IRIX, the SGI implementation of the UNIX operating system
- *IRIX Admin: System Configuration and Operation* (007-2859-*nnn*) for information about IRIX system administration tasks
- *IRIX Admin: Disks and Filesystems* (007-2825-*nnn*) for information about general filesystem concepts and system administration procedures for SCSI disks, XFS and EFS filesystems, logical volumes, and guaranteed rate I/O
- *O2 Workstation Hardware Reference Guide* (007-3275-*nnn*) for information on the O2 workstation
- *Origin200 and Origin200 GIGAchannel Owner's Guide* (007-3708-*nnn*) and *Origin200 and Origin200 GIGAchannel Maintenance Guide* (part number 007-3709-*nnn*) for information about the SGI Origin 200 and Origin 200 GIGAchannel
- *DIVO Option and DIVO-DVC Option Owner's Guide* (007-3524-*nnn*) for information about the DIVO and DIVO-DVC XIO video option boards
- *O2 Digital Video Option Installation Guide* (007-3616-*nnn*), for information about the O2 digital video option module
- *StudioCentral Developer's Guide* (007-3246-*nnn*), for information regarding the StudioCentral archiving system
- *Digital Media Programmer's Guide* (part number 007-1799-*nnn*) for information about programming digital media

Conventions Used in This Document

The following type and symbol conventions are used in this document:

Italics Used for filenames, pathnames, directory names, emphasis, document titles, variable names, glossary terms, and command-line programs.

Bold Used for keywords.

`Fixed-width` Used for code examples and command syntax.

`Bold fixed-width`
Used for user input, including nonprinting keyboard keys.

Square brackets ([])
Surround syntax statement arguments that are optional.

Square bullets (■)
Indicate substeps within a multistep process.

Ellipsis (...) Indicates that the preceding is repeated or that material is omitted.

Angle brackets (>)
Indicate a path through menus to a menu option. For example, “File > Open” means “Under the File menu, choose the Open option.”

Setting Up the Hardware and the System

This chapter explains the steps you need to take to set up your system so that Video Server Toolkit (VST) can run on it, in these sections:

Supported Hardware

This section consists of the following subsections:

Platforms

VST works on the following platforms running IRIX version 6.5.4 or later:

- SGI Origin 200 GIGAchannel server with one of the following installed:
  - SGI DIVO or DIVO-DVC video option board (XIO)
  - Viewgraphics Dynamo MediaPump multiplexer adapter PCI board, compliant with the Digital Video Broadcast Asynchronous Serial Interface (DVB-ASI) protocol
  - Vela Research four-port SCSI MPEG-2 decoder
- Silicon Graphics O2 workstation with O2Video (mvp)

Port Configuration

The SGI Origin 200 GIGAchannel single-module system has six PCI slots and five XIO slots; one XIO slot can be used for the 100-Base-T XIO option board, which has six serial ports.

External Devices

VST can interact with a variety of external devices, as shown in Figure 1-1.

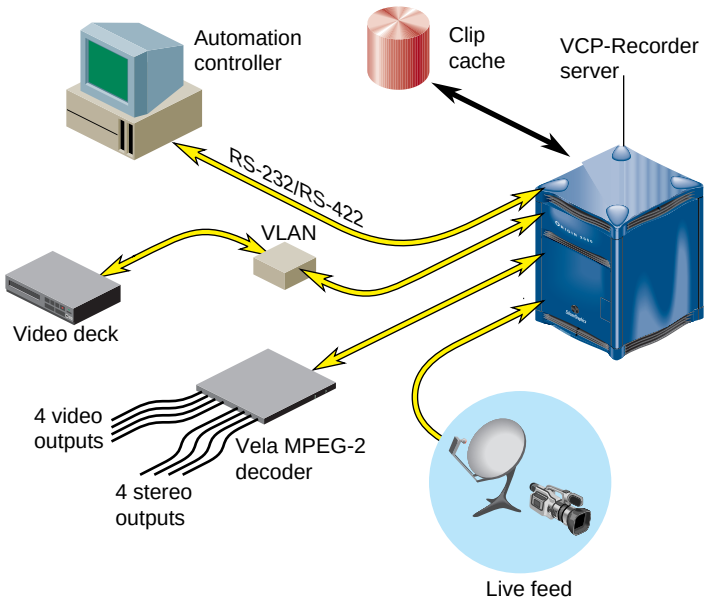

Figure 1-1 VST With External Devices

Installation of VST includes configuring it to work with external devices:

- Devices that control VST:
  - video tape recorder edit controller or other device for controlling VTRs, using a Sony-compatible (P2-compatible) RS-422 control protocol
  - automation controller using the Louth Video Disk Communications Protocol

- Devices that VST controls: most broadcast VTRs using the Sony-compatible RS-422 control protocol
- Devices through which media are played or recorded:
  - SGI DIVO video option board: 8- and 10-bit Rec. 601 digital video playback and recording with optional 2:1 lossless data reduction
  - SGI DIVO-DVC video option board: 8- and 10-bit Rec. 601 digital video playback and recording with optional 2:1 lossless data reduction or DVCPRO25 compression
  - Vela Research four-port SCSI MPEG-2 decoder (MPEG-2 transport streams (main profile at main level) playback)
  - Viewgraphics Dynamo MediaPump
  - SGI Digital Audio Option board for 24-bit uncompressed AES or ADAT digital audio
  - DVB-ASI devices
- Auxiliary devices: house time-code readers

Disk Storage Options

Origin servers have two storage options:

- SGI Origin 200 GIGAchannel internal drive bays (four 18-GB Ultra SCSI disk modules (32 MB/sec))
- Ciprico 7000 Fibre Channel RAID disk array (44 MB/sec per RAID)

Bandwidth Considerations

A stream is a video/audio program. The number of streams a server can record or play at once depends on the storage option and the compression type.

For example, if you assume that 8-bit 4:2:2 Rice variable compression is, at worst, 11 MB/sec, an Ultra SCSI system can handle two Rice-compressed video streams (32 MB/11 MB = 2 streams) or eight DVCPRO-compressed video streams (32 MB/3.6 MB = 8 streams).

A single Ciprico fibre channel RAID disk array can support four 8-bit 4:2:2 Rice-encoded video streams (44 / 11 = 4, assuming average 2:1 data reduction; some video streams may require more bandwidth) or 12 DVCPRO video streams (44 / 3.6 = 12).

Compression Formats

The bandwidth determines the number of streams of video that can be played or recorded at once. Each compression format requires a different bandwidth, as summarized in Table 1-1.

Table 1-1 Compression Formats and Bandwidths

Compression Format	Bandwidth
8-bit 4:2:2 uncompressed	21 MB / sec
8-bit 4:2:2 Rice-encoded	11 MB / sec (typical)
DVCPRO	3.6 MB / sec
MPEG-2 MP@ML	0.4 to 1.5 MB / sec (~3 to 12 Mb / sec)

Because the Rice-coded format is lossless, there are few reasons to use uncompressed video unless the video files or frames will be used by other programs, such as an editing application.

Running uncompressed video on an O2 workstation is not supported in this version of VST.

Hardware Installation

To set up hardware for use with VST, follow these steps:

1. Install the server, interface boards, and storage following instructions in the manual(s).
2. From the system administrator at your site, obtain a name and IP address for the system.
3. Connect the system to the network using a 10-Base-T Ethernet cable between the RJ-45 connector on the O2 workstation or Origin 200 GIGAchannel master module and a network port.

4. Connect the system console (RS-232/RS-422 serial) port on the Origin 200 GIGAchannel master module to a serial port of another system or an ASCII terminal using a serial cable.

   See instructions for using an SGI workstation as the system console or attaching the system console in the *Origin200 GIGAchannel Owner's Guide*.

5. Edit */etc/uucp/Devices* to remove the comment; change

   ```
   # --A direct line so `cu -lttyd2` will work
   # Direct ttyd2 - 9600 direct
   ```

   to

   ```
   # --A direct line so `cu -lttyd2` will work
   Direct ttyd2 - 9600 direct
   ```

6. Log in to the system console using the serial port on another system. Make sure the serial port on the other system is not being used for another purpose, such as its own system console.

   Use the following command if the other system is an SGI server.

   ```
   cu -s 9600 -l ttyd2
   ```

7. Install any hardware and associated driver software related to VST, such as DIVO/DIVO-DVC, Vela, or DVB-ASI boards, following instructions in the manuals for the products and in Chapter 6, "Configuring and Using External Devices."

Initial System Administration

Before turning to Chapter 2 to set up filesystems, complete the following steps to perform basic administration tasks:

1. Name the system.

   ```
   vi /etc/sys_id
   ```

2. Add system name and IP address to the hostname-address database.

   ```
   vi /etc/hosts
   ```

3. Set IP address in non-volatile RAM; also, set system up for automatic restart.

   ```
   nvram netaddr <IP Address>
   nvram AutoLoad Y
   ```

4. Set network address mask, as appropriate for the network.

   vi /etc/config/ifconfig-1.options

5. Set default system time zone.

   vi /etc/TIMEZONE

6. Set password for user root, if necessary.

   passwd

7. Reboot the system.

   reboot

Setting Up Filesystems for Video Server Toolkit

This chapter describes how to set up XFS filesystems for Video Server Toolkit and for the VST clip cache, which holds the VST media assets. The clip cache h is maintained on one or more filesystems across one or more disks or RAID arrays. You set up the filesystems on a system disk and a duplicate disk.

This chapter includes explicit steps for setting up the plexed filesystem. However, it is beyond the scope of this book to explain in detail the intricacies of the tools, including XFS, XLV, and Guaranteed Rate I/O (GRIO), that you use to set up your filesystem. For information about any of these tools, see the latest version of *IRIX Admin: Disks and Filesystems*.

This chapter contains these sections:

Preparing for Logical Volume Creation

A logical volume is a set of disks or disk arrays that are treated as one disk. Logical volumes (XLVs) allow for the creation of a real-time subvolume to be used within a real-time filesystem. XLVs are also used to stripe data across a set of disks or disk arrays to form a subvolume.

This section consists of the following:

For more information about XLV, see *IRIX Admin: Disks and Filesystems*.

Planning for Optimal Performance

For optimal performance, follow these recommendations:

- All RAID arrays should be RAID level 3.
- Use real-time filesystems whenever possible so that VST can use GRIO, which provides highly reliable video streaming.
- If multiple disks or RAID arrays are available, use XLV to stripe data across the disks.

  Note: Striping across multiple device types is not possible.

The creation of real-time filesystems requires the use of logical volumes. Logical volumes also provide the mechanisms for data striping. For information on configuring and using RAID arrays by creating logical units (LUNs), refer to the administration guide of the RAID storage system you are using.

Note: You can use a non-real-time filesystem, particularly if you are setting up a development system, but with some loss of performance. See "Configuring Non-Real-Time Filesystems" on page 29.

Using Real-Time Filesystems

An XFS filesystem can be created on an XLV disk volume that contains a real-time subvolume. The real-time subvolume is a separate data area, usually created on a separate set of disks or disk arrays, that is optimized for real-time recording and playback of digital media. When an XFS filesystem is part of the VST clip cache, the real-time subvolume, if available, is where the media data is stored. Only real-time filesystems have guaranteed real-time I/O rates (GRIO).

Data can be read normally from a real-time filesystem, but standard utilities, such as *ftp* and *cp*, cannot be used to write the data. However, the VST installation replaces the standard *ftp* daemon in the configuration file */etc/inetd.conf* with *vtrftpd*, an enhanced version that is installed from *vst_eoe.sw.ftpd*. Thus, invoking *ftp* automatically initiates *vtrftpd*, which supports a site marks facility that keeps track of in and out points in clips.

For more information on the use of real-time filesystems, see *Video Server Toolkit Developer's Guide*.

Determining Volume Size

Table 2-1 can help you determine the size of the volumes and therefore the amount of disk space you need according to different compression formats.

Table 2-1 Disk Space Requirements

Format	Rate (MB/sec)	Disk Space for One Minute	Disk Space (GB) for One Hour
Uncompressed	22	1.32 GB	79
Rice lossless	11	0.66 GB	39.6
DVCPRO	3.6	216 MB	13

Rice values are typically one half of uncompressed values. The bit rate for MPEG clips depends upon the settings of the MPEG encoder which recorded it.

Observe the following guidelines when you create a logical volume to be used as part of VST's clip cache:

- Set data subvolume size to at least 200 MB.
- Create a real-time subvolume using a stripe size based on the striping guidelines in "Determining Volume Striping Values" on page 10.
- If GRIO is used, put only real-time subvolumes on all partitions of any disks or disk arrays.
- If a real-time subvolume is placed on a RAID subsystem(s), plex the data subvolume on a pair of internal disks to improve reliability.

Subsections of "Setting Up Plexing" later in this chapter contain explicit instructions for setting up plexing.

Determining Volume Striping Values

Part of the job of creating a logical volume is to supply XLV with values for striping the disks. Striping is the process of distributing media data across several disks so that different parts of the media can be simultaneously accessed, which provides a higher overall bandwidth.

Striping is highly recommended for real-time performance. You can stripe disks for the real-time subvolume of real-time filesystems (recommended), or for non-real time filesystems (not recommended).

Follow these guidelines for striping volumes:

- Choose a larger stripe unit to improve performance, but do not exceed the optimal I/O size for the individual disk devices being striped. Exceeding the optimal size does not significantly improve performance, but increases system memory requirements.
  - For non-RAID disks and disk arrays, the stripe unit per disk should be at lest 256 KB ({512 disk blocks), preferably 612 KB (1024 disk blocks).
  - For a 4+1 RAID-3 disk array, the stripe unit should be at least 1 MB ({2048 disk blocks), preferably 2 MB (4096 disk blocks).
  - For an 8+1 RAID-3 disk array, the stripe unit should be at least 2 MB (4096 disk blocks), preferably 4 MB {(8192 disk blocks).
- If a real-time subvolume is being striped, the real-time extent size of the XFS filesystem must match the stripe size given by the following formula:

  num-striped-disks `* stripe-unit * 512bytes/block`

Uncompressed Media Striping

A stripe unit value that is a multiple of 552 disk blocks (226 KB) is optimal for 525-line/59.94 8-bit 4:2:2 uncompressed video use. This results in a stripe size of 2,260,992 bytes for an eight-disk volume, or 1,081,344 bytes for a four-disk volume. This stripe size holds, respectively, exactly six or three uncompressed 4:2:2 eight-bit fields (2 bytes/pixel) when the field is rounded up to the nearest 16 KB to match the recommended filesystem block size.

A stripe unit value that is a multiple of 624 disk blocks (312 KB) is optimal for 625-line / 50 8-bit 4:2:2 uncompressed video use. This results in a stripe size of 2,654,208, bytes for an eight-disk volume, or 1,327,104 bytes for a four-disk volume. This stripe size holds, respectively, exactly six or three uncompressed 4:2:2 eight-bit fields (2 bytes / pixel) when the field is rounded up to the nearest 16 KB to match the recommended filesystem block size.

Non-Real-Time, Non-Striped Filesystems

If you use a non-real-time filesystem with no striping, add an entry to */usr/vtr/config/vtrfsinfo.conf* to specify the optimal I/O size for the disk device. For example, to specify an optimal I/O size of 4 MB for a single RAID device, use:

```
/dev/dsk/dks2d1s7 4m
```

Choosing Parameters for Creating and Mounting an XFS Filesystem

When you create an XFS filesystem, follow these guidelines:

- Set filesystem block size to 16 KB only.
- Set the real-time extent size to the optimal I/O size for the real-time subvolume.

  If the subvolume is striped, it must match the stripe size. If the subvolume is not striped (single disk or RAID array), it should match the optimal I/O size for the disk system.

  For a single RAID, use 2 to 4 MB for the extent size. For more information about determining the optimal I/O size for a disk, see the striping guidelines in "Using GRIO" on page 30.

For more information about XFS, see *IRIX Admin: Disks and Filesystems*.

Using Redundancy

VST works with IRIS FailSafe 1.2 to provide failover capability. You can also set up one or more servers for clip mirroring, to reflect the state of the primary server's clip cache. Chapter 5, "Configuring for Redundancy," describes how to configure for these options.

Setting Up Plexing

Setting up plexing consists of several procedures, explained in the following subsections:

Caution: Partitioning the disk destroys all data on it.

Partitioning the System Disk

Table 2-2 summarizes the desired setup for the system disk in a redundant system.

Table 2-2 System Disk Partitions

Partition	Contents	Size
0	root, type xlv	256 MB
1	swap, type raw	512 MB
6	usr, type xlv	rest of disk
7	data subvolume, type xlv	1024 MB
8	volhdr	2 MB

Note: Have ready the following CDs: IRIX 6.5.4 and the IRIX 6.5 CDs: Installation Tools, Foundation 1, Foundation 2, Applications, and NFS.

This subsection gives instructions for partitioning the system disk and breaking the current IRIX installation. Follow these steps:

1. Perform initial system administration, as explained in “Initial System Administration” on page 5 in Chapter 1.
2. Back up data on the disk as necessary.
3. Enter

   # **fx -x**

   The system responds:

   ```
   fx version 6.5, Apr 13, 1999
   fx: "device-name" = (dksc)
   ```

4. Press **Enter**. The system responds:

   ```
   fx: ctlr# = (0)
   ```

5. Press **Enter**. The system responds:

   ```
   fx: drive# = (1)
   ```

6. Press **Enter**. The system responds:

   ```
   fx: lun# = (0)
   ```

7. Press **Enter**. The system responds:

   ```
   ...opening dksc(0,1,0)
   ...drive selftest...OK
   Scsi drive type == SGI     IBM  DGHS18Y    0190

   ----- please choose one (? for help, .. to quit this menu)-----
   [exi]t             [d]ebug/           [l]abel/           [a]uto
   [b]adblock/        [exe]rcise/        [r]epartition/
   fx>
   ```

8. Enter **repartition/expert**. The system responds:

   ```
   Warning: you will need to re-install all software and restore user
   data from backups after changing the partition layout.  Changing
   partitions will cause all data on the drive to be lost.  Be sure you
   have the drive backed up if it contains any user data.  Continue?
   ```

9. Enter **yes**. The system responds:

   ```
   Enter .. when done
   fx/repartition/expert: change partition = (0)
   ```

10. Press **Enter**. The system responds:

```
before:  type xfs        block       0,         0 MB
                         len:        0 blks,    0 MB
fx/repartition/expert: partition type = (xfs)
```

11. Enter **xlv**. The system responds:

```
fx/repartition/expert: base in megabytes = (0)
```

12. Enter **2**. The system responds:

```
fx/repartition/expert: size in megabytes (max 17363) = (0)
```

13. Enter **256**. The system responds:

```
 after:  type xlv        block    4096,         2 MB
                         len:   524288 blks,  256 MB
fx/repartition/expert: change partition = (1)
```

14. Enter **1**. The system responds:

```
before:  type xfs        block       0,         0 MB
                         len:        0 blks,    0 MB
fx/repartition/expert: partition type = (xfs)
```

15. Enter **raw**. The system responds:

```
fx/repartition/expert: base in megabytes = (0)
```

16. Enter **258**. The system responds:

```
fx/repartition/expert: size in megabytes (max 17107) = (0)
```

17. Enter **512**. The system responds:

```
after:  type raw        block  528384,       258 MB
                         len:  1048576 blks,  512 MB
fx/repartition/expert: change partition = (6)
```

18. Enter **7**. The system responds:

```
before:  type xfs        block    4096,         2 MB
                         len:  35558944 blks, 17363 MB
fx/repartition/expert: partition type = (xfs)
```

19. Enter **xlv**. The system responds:

```
fx/repartition/expert: base in megabytes = (2)
```

20. Enter **770**. The system responds:

```
fx/repartition/expert: size in megabytes (max 16595) = (17363)
```

21. Enter **1024**. The system responds:

```
 after:  type xlv        block 1576960,       770 MB
                          len:  2097152 blks, 1024 MB
fx/repartition/expert: change partition = (8)
```

22. Enter **6**. The system responds:

```
before:  type xfs        block       0,         0 MB
                         len:        0 blks,    0 MB
fx/repartition/expert: partition type = (xfs)
```

23. Enter **xlv**. The system responds:

```
fx/repartition/expert: base in megabytes = (0)
```

24. Enter **1794**. The system responds:

```
fx/repartition/expert: size in megabytes (max 15571) = (0)
```

25. Enter **15570** (that is, max -1) The system responds:

```
after:  type xlv        block 3674112,      1794 MB
                         len:  31887360 blks, 15570 MB
fx/repartition/expert: change partition = (7)
```

26. Enter **..** (two periods). The system responds:

```
----- partitions-----
part  type        blocks            Megabytes   (base+size)
  0: xlv        4096 + 524288         2 + 256
  1: raw      528384 + 1048576      258 + 512
  6: xlv     3674112 + 31887360    1794 + 15570
  7: xlv     1576960 + 2097152      770 + 1024
  8: volhdr        0 + 4096           0 + 2
 10: volume        0 + 35563040       0 + 17365

capacity is 35563040 blocks

----- please choose one (? for help, .. to quit this menu)-----
[ro]otdrive           [o]ptiondrive         [e]xpert
[u]srrootdrive        [re]size
fx/repartition>

* Enter /label/sync

writing label info to dksc(0,1,0)
```

```
----- partitions-----
part  type        blocks            Megabytes   (base+size)
  0: xlv        4096 + 524288         2 + 256
  1: raw      528384 + 1048576      258 + 512
  6: xlv     3674112 + 31887360    1794 + 15570
  7: xlv     1576960 + 2097152      770 + 1024
  8: volhdr        0 + 4096           0 + 2
 10: volume        0 + 35563040       0 + 17365

capacity is 35563040 blocks

----- please choose one (? for help, .. to quit this menu)-----
[ro]otdrive           [o]ptiondrive         [e]xpert
[u]srrootdrive        [re]size
fx/repartition>
```

27. Enter **/exit**.

Installing System Software on the System Disk

When the label is written to disk, the system must be powered on again. Follow these steps:

1. Power on the system. After the PROM executes, a message like the following appears:

   ```
   Autoboot failed.
   dksc(0,1,0)unix: no such file or directory.

   Hit Enter to continue.
   ```

2. Press **Enter**. The System Maintenance Menu appears:

   ```
    System Maintenance Menu

   1) Start System
   2) Install System Software
   3) Run Diagnostics
   4) Recover System
   5) Enter Command Monitor
   ```

3. Choose “Install System Software” (2). The following message appears:

```
Installing System Software...

Press <Esc> to return to the menu.
1) Remote Directory  X) Local CD-ROM

Enter 1-2 to select source type, <esc> to quit,
or <enter> to start:
```

4. Load the IRIX 6.5.4 CD into the remote or local CD-ROM drive.
   - If you are installing from a local CD-ROM, enter **2** and continue at step 7.
   - If you are installing from a remote directory, enter **1**. The system responds:

     ```
     Enter the name of the remote host:
     ```

5. If you are installing from a remote CD-ROM, enter the name of the remote host. The system responds:

   ```
   Enter the remote directory:
   ```

6. Enter **/CDROM/dist**. The system responds:

   ```
   1)[Remote Directory]  X) Local CD-ROM
         *a) Remote directory /CDROM/dist from server remoteHost.

   Enter 1-2 to select source type, a to select the source, <esc> to
   quit, or <enter> to start:
   ```

7. Press **Enter**. The system responds:

   ```
   Copying installation program to disk.
   ......... 10% ......... 20% ......... 30% ......... 40% ....... 50%
   ......... 60% ......... 70% ......... 80% ......... 90% ....... 100%

   Copy complete
   IRIX Release 6.5 IP27 Version 04151556 System V - 64 Bit
   Copyright 1987-1999 Silicon Graphics, Inc.
   All Rights Reserved.

   Setting rbaud to 19200
   WARNING: Cannot load runtime symbol table from bootp’ed kernel.
            Loadable modules will not be registered or loaded.
   Creating miniroot devices, please wait...

   Current system date is Mon May 17 15:30:19 PDT 1999
   ```

```
Mounting file systems:

    /dev/miniroot            on  /
    /dev/dsk/dks0d3s0        on  /root
    /dev/dsk/dks0d3s6        on  /root/usr


Invoking software installation.
```

8. The miniroot prompts for hostname, network address, and netmask; enter these for your system.
9. At the Inst prompt, enter **admin** to display the Admin menu.
10. At the Admin prompt, enter **sh** to get a shell.
11. In the shell, enter

    ```
    # mkfs /dev/dsk/dks0d3s6
    ```

    The system responds:

    ```
    meta-data=/dev/dsk/dks0d3s6   isize=256   agcount=16, agsize=249133
    blks
    data     =                    bsize=4096  blocks=3986116, imaxpct=25
             =                    sunit=0     swidth=0 blks, unwritten=1
    naming   =version 1           bsize=4096
    log      =internal log        bsize=4096  blocks=1000
    realtime =none                extsz=65536  blocks=0, rtextents=
    ```

12. Enter

    ```
    # mount /dev/dsk/dks0d3s6 /root/usr
    # exit
    ```

13. The Admin menu appears; enter **exit** to get to the Inst main menu.
14. Insert the IRIX 6.5 CDs in the following order:
    - Installation Tools
    - Foundation-1
    - Foundation-2
    - Applications
    - NFS

    From the Inst menu, select Open to open the distributions on each CD. Unselect the distributions that you do not want.

15. Install the following non-default subsystems:

```
Inst> install eoe.sw.xlv eoe.sw.xfsrt eoe.sw.xlvplex
Inst> go
```

    Installing the 6.5 subsystems takes a bit of time.

16. Replace the last IRIX 6.5 CD with the IRIX 6.5.4 CD. Select the maintenance stream.

```
Inst> keep N
Inst> go
```

17. Enter the following to edit *root/etc/fstab*:

```
Inst> Admin
Admin> sh
# cat > /root/etc/fstab
/dev/root /  xfs  rw,raw=/dev/rroot 0 0
/dev/usr /usr  xfs  rw,raw=/dev/rusr 0 0
```

18. Enter **Crtrl+D** to close *root/etc/fstab*. Exit from the shell and Admin, and quit Inst.

19. The system prompts for a reboot; enter **yes** at the prompt.

20. Perform an initial system administration as explained in “Initial System Administration” on page 5 in Chapter 1.

Duplicating the System Disk

Once you have installed the system software on the system disk, duplicate it; follow these steps:

1. Choose another disk as a backup option disk (for example, disk 2) and enter:

```
# fx -x "dksc"(0,2)
```

2. Follow steps 8 on page 13 through 27 on page 16.

3. As root, create a filesystem on partition 0:

   ```
   # mkfs /dev/dsk/dks0d2s0
   ```

   The system responds:

   ```
   meta-data=/dev/dsk/dks0d3s0      isize=256    agcount=8, agsize=8192
   blks
   data     =                       bsize=4096   blocks=65536,
   imaxpct=25
            =                       sunit=0      swidth=0 blks,
   unwritten=1
   naming   =version 1              bsize=4096
   log      =internal log           bsize=4096   blocks=1000
   realtime =none                   extsz=65536  blocks=0, rtextents=0
   ```

4. Create filesystem on */dev/dsk/dks0d2s6* and */dev/dsk/dks0d2s7* in a similar way.

5. Use *xfs_copy* to copy the current system disk to the duplicate disk, with root copied into */dev/dsk/dks0d2s0* and */usr* copied into */dev/dsk/dks0d2s6*:

   ```
   xfs_copy /dev/dsk/dks0d1s0 /dev/dsk/dks0d2s0
   xfs_copy /dev/dsk/dks0d1s6 /dev/dsk/dks0d2s6
   ```

   Ignore messages about the mounted source drive.

Installing an XFS Flexlm XLV License

Once you have duplicated the system disk, use the following procedure to set up the plexing:

1. To make sure that plexing is enabled and supported, start *xlv* and run the *show config* command:

   ```
   # xlv_mgr
   xlv_mgr> show config
   ```

   The system responds:

   ```
   Plexing license: present
   Plexing support: present
   ```

2. Exit *xlv_mgr*:

   ```
   xlv_mgr> exit
   ```

3. If a plexing license is not present, to install the license in */var/flexlm/license.dat*. Provide a symlink to the */etc/flexlm* directory:

   ```
   # ln -s /var/flexlm /etc/flexlm
   ```

4. If you do not have a license for XLV, you must request one through the Key-O-Matic Web page http://www.sgi.com/Support/Licensing/. If your request is granted, you receive a license by e-mail or in your Web browser.

5. Once you receive the license, log on to the VST server.

6. As superuser, open the license file.

   ```
   # vi /etc/flexlm/license.dat
   ```

7. Copy and paste license lines from the web site into the license file; for example:

   ```
   FEATURE XLV sgifd 2.000 01-jan-0 0 6CC5738489148F24823C \
   HOSTID=#1761963733 vendor_info="XLV XFS PLEXING" \
   ISSUER="Silicon Graphics, Inc."
   ```

8. Save the file and exit *vi*.

Making the Root Partition on the System Disk Into an XLV Volume

The next procedure is to make the root partition on the system disk into an XLV volume; follow these steps:

1. Enter:

   ```
   # xlv_make
   ```

   The system responds:

   ```
   xlv_make>
   ```

2. Enter **`vol root.`** The system responds:

   ```
   xlv_make>
   ```

3. Enter **`data`**.

   ```
   root.data
   xlv_make>
   ```

4. Enter **`ve -force /dev/dsk/dks0d1s0.`** The system responds:

   ```
   root.data.0.0
   xlv_make>
   ```

5. Enter **end**. The system responds:

   ```
   Object specification completed
   xlv_make>
   ```

6. Enter **vol usr**. The system responds:

   ```
   usr
   xlv_make>
   ```

7. Enter **data**. The system responds:

   ```
   usr.data
   xlv_make>
   ```

8. Enter **ve -force /dev/dsk/dks0d1s6**. The system responds:

   ```
   usr.data.0.0
   xlv_make>
   ```

9. Enter **end**. The system responds:

   ```
   Object specification completed
   xlv_make>
   ```

10. Enter **exit**. The system responds:

    ```
    Newly created objects will be written to disk.
    Is this what you want?(yes)
    ```

11. Enter **yes**. The system responds:

    ```
    Invoking xlv_assemble
    ```

    The result is XLV volumes named *root* and *usr* that contain the root and usr partitions on the system disk. Since XLV preserves the data in partitions, the contents of the root and usr partitions are preserved. The **-force** option to the *ve* command was used because a mounted partition was included in the volume.

12. Open */etc/fstab* and add the following entry:

    ```
    /dev/xlv/usr/   /usr   xfs   rw   0   0
    ```

13. Create a */etc/config/xlv_plexd.options* file with the following content:

    ```
    # cat > /etc/config/xlv_plexd.options
      -b 1024 -w 50

    Press ctrl-D   # to close the file
    ```

    These commands make sure that a revive does not use too much of the disk's bandwidth.

14. Reboot the system so that it switches from running from the root partition on the system disk (*/dev/dsk/dks0d1s0*) to running from the logical volume */dev/xlv/root*.

Creating the Plex for the Root

After making the root partition on the system disk into an XLV volume, create the plex for the root. Follow these steps:

1. Create the plex for the root out of */dev/dsk/dks0d2s0*, and call it *root_plex*; also create plex for usr out of */dev/dsk/dks0d2s6*, and call it *usr_plex*.
2. Enter

   ```
   # xlv_make
   ```

   The system responds:

   ```
   xlv_make>
   ```

3. Enter **`plex root_plex`**. The system responds:

   ```
   root_plex
   xlv_make>
   ```

4. Enter **`ve /dev/dsk/dks0d2s0`**. The system responds:

   ```
   root_plex.0
   xlv_make>
   ```

5. Enter **`end`**. The system responds:

   ```
   Object specification completed
   xlv_make>
   ```

6. Enter **`plex usr_plex`**. The system responds:

   ```
   usr_plex
   xlv_make>
   ```

7. Enter **`ve /dev/dsk/dks0d2s6`**. The system responds:

   ```
   usr_plex.0
   xlv_make>
   ```

8. Enter **`end`**. The system responds:

   ```
   Object specification completed
   xlv_make>
   ```

9. Enter **`exit`**. The system responds:

   ```
   Newly created objects will be written to disk.
   Is this what you want?(yes)
   ```

10. Enter **`yes`**. The system responds:

    ```
    Invoking xlv_assemble
    ```

11. Add *sash* to the volume header of the option disk used for the *root_plex*. It enables booting from of the *root_plex* if the primary root fails.

    ```
    # dvhtool -v get sash /tmp/sash /dev/rdsk/dks0d1vh
    # dvhtool -v add /tmp/sash sash /dev/rdsk/dks0d2vh
    ```

12. Attach the *root_plex* to the root volume, and *usr_plex* to the *usr* volume using *xlv_mgr*.

    ```
    # xlv_mgr
    ```

    The system responds:

    ```
    xlv_mgr>
    ```

13. Enter the following three lines in *xlv_mgr*:

    ```
    xlv_mgr> attach plex root_plex root.data
    xlv_mgr> attach plex usr_plex usr.data
    xlv_mgr> quit
    ```

    When the shell prompt returns, the system automatically begins a plex revive so that the two plexes contain the same data.

Partitioning the Clip Cache

After creating the plex for the root, partition the clip cache. Follow these steps:

1. Determine the SCSI ID and bus info of the Ciprico FC disk arrays.

   ```
   # hinv
   ```

   The system responds:

   ```
   FPU: MIPS R10010 Floating Point Chip Revision: 0.0
   CPU: MIPS R10000 Processor Chip Revision: 2.6
   2 180 MHZ IP27 P
   ...

   Integral SCSI controller 3: Version Fibre Channel AIC-1160, revision
   2
   Disk drive: unit 1 on SCSI controller 3
   Integral SCSI controller 2: Version Fibre Channel AIC-1160, revision
   2
   Disk drive: unit 1 on SCSI controller 2
   ...

   DIVO Video: controller 2 unit 2: Input, Output
   DIVO Video: controller 3 unit 3: Input, Output
   IOC3 external interrupts: 1
   ```

2. Start *fx*:

   ```
   # fx -x
   fx: devicename = (dksc)
   ```

3. Press **Enter**. The system responds:

   ```
   fx: ctlr# = (0)
   ```

4. Enter the controller where the Ciprico device is, for example, **3**. The system responds:

   ```
   fx: drive# = (1)
   ```

5. Press **Enter**. The system responds:

   ```
   fx: lun# = (0)
   ```

6. Press **Enter**. The system responds:

   ```
   ----- please choose one (? for help, .. to quit this menu)-----
   [exi]t            [d]ebug/          [l]abel/          [a]uto
   [b]adblock/       [exe]rcise/       [r]epartition/    [f]ormat
   fx>
   ```

```
----- partitions-----
part  type        blocks            Megabytes   (base+size)
  7: xlv        4096 + 142258752       2 + 69462
  8: volhdr        0 + 4096           0 + 2
 10: volume        0 + 142262848       0 + 69464

capacity is 142262848 blocks

----- please choose one (? for help, .. to quit this menu)-----
[ro]otdrive           [o]ptiondrive         [e]xpert
[u]srrootdrive        [re]size
fx/repartition>
```

7. Enter **o** for option drive. The system responds:

   ```
   fx/repartition/optiondrive: type of data partition = (xfs)
   ```

8. Press **Enter**. The system responds:

   ```
   fx/repartition/optiondrive: create log partition? = (no)
   ```

9. Press **Enter**. The system responds:

   ```
   Warning: you will need to re-install all software and restore user
   data
   from backups after changing the partition layout.  Changing
   partitions
   will cause all data on the drive to be lost.  Be sure you have the
   drive
   backed up if it contains any user data.  Continue? yes

   ----- partitions-----
   part  type        blocks            Megabytes   (base+size)
     7: xlv        4096 + 142258752       2 + 69462
     8: volhdr        0 + 4096           0 + 2
    10: volume        0 + 142262848       0 + 69464

   capacity is 142262848 blocks

   ----- please choose one (? for help, .. to quit this menu)-----
   [ro]otdrive           [o]ptiondrive         [e]xpert
   [u]srrootdrive        [re]size
   fx/repartition>
   ```

10. Enter **/label/sync.** The system responds:

```
writing label info to dksc(0,3,0)

----- partitions-----
part  type        blocks            Megabytes   (base+size)
  7: xlv        4096 + 142258752       2 + 69462
  8: volhdr        0 + 4096           0 + 2
 10: volume        0 + 142262848       0 + 69464

capacity is 142262848 blocks

----- please choose one (? for help, .. to quit this menu)-----
[ro]otdrive           [o]ptiondrive         [e]xpert
[u]srrootdrive        [re]size

fx/repartition>
```

11. Enter **/exit**.

12. Repeat steps 3 on page 25 through step 11 on page 27 for each Ciprico RAID array.

Creating the XLV Volume for the Clip Cache

After partitioning the clip cache, create the XLV volume for it. Follow these steps:

1. Create the XLV volume for the clip cache using the data partitions on the system and option disk, and the two fibre channel RAID arrays. Enter:

   ```
   # xlv_make
   ```

   The system responds:

   ```
   xlv_make>
   ```

2. Enter the volume's name, for example, `vol vtr`. The system responds:

   ```
   vtr
   xlv_make>
   ```

3. Enter **data**. The system responds:

   ```
   vtr.data
   xlv_make>
   ```

4. Enter **plex**. The system responds:

   ```
   vtr.data.0
   xlv_make>
   ```

5. Enter **ve dks0d1s7**. The system responds:

```
vtr.data.0.0
xlv_make>
```

6. Enter **plex**. The system responds:

```
vtr.data.1
xlv_make>
```

7. Enter **ve dks0d2s7**. The system responds:

```
vtr.data.1.0
xlv_make>
```

8. Enter **rt**. The system responds:

```
vtr.rt
xlv_make>
```

9. Enter **plex**. The system responds:

```
vtr.rt.0
xlv_make>
```

10. Enter the following:

```
xlv_make> ve -stripe -stripe_unit 4096 /dev/dsk/dks2d1s7
/dev/dsk/dks3d1s7.
```

The system responds:

```
vtr.rt.0.0
xlv_make>
```

11. Enter **end**. The system responds:

```
Object specification completed
xlv_make>
```

12. Enter **exit**. The system responds:

```
Newly created objects will be written to disk.
Is this what you want?(yes)
```

13. Enter **yes**. The system responds:

```
Invoking xlv_assemble
```

14. Create an XFS filesystem on the clip's logical volume:

```
# mkfs_xfs -b size=16k -r extsize=4m /dev/xlv/vtr
```

15. Create an entry in the filesystem table:

```
# vi /etc/fstab
/dev/xlv/vtr  /usr/vtr/clips xfs     rw,raw=/dev/rxlv/vtr  0 0
```

16. Mount the *clips* filesystem:

```
# mount -av
```

17. Create required links for */var/adm* and */var/tmp:*

```
# mkdir /usr/vtrtmp
# rm -r /var/tmp
# rm -r /usr/tmp
# ln -s /usr/vtrtmp /var/tmp
# ln -s /var/tmp /usr/tmp
```

Configuring Non-Real-Time Filesystems

Normal VST system configuration requires that the clip cache */usr/vtr/clips* reside on a filesystem that has a real-time subvolume. However, in developer or other non-production configurations, the clip cache can be on a filesystem that does not include a real-time subvolume.

If the clip cache filesystem does not contain a real-time subvolume and the filesystem does not reside on a striped (RAID-0) disk volume, the filesystem configuration file */usr/vtr/config/vtrfsinfo.conf* must have an entry that specifies the I/O size and alignment required for the filesystem.

The configuration file includes a line for each disk device or logical volume used for playback or recording of clips; it has the following syntax:

device_name I/O_size alignment_size

For example:

```
/dev/xlv/striped 4m 4m
```

device_name

device_name is the device path for the disk or the XLV volume. The device name is either the block device name of the physical disk itself (for example, */dev/dsk/dks2d1s7*) or is the block device name of the logical volume (for example, */dev/xlv/clips*).

I/O_size

I/O_size is the desired I/O size, in bytes, for the disk device or volume. Select a value of at least 256 KB per disk spindle but no more than 512 KB per disk spindle. Use the suffix "k" as an abbreviation for 1024 bytes and the suffix "m" for 1024*1024 bytes.

For example, to specify a single Ultra-SCSI-attached disk, the I/O size should be from 256 KB to 512 KB. The I/O size for an 8+1 fibre channel RAID subsystem should be from 2 MB (8*256 KB) to 4 MB (8*512 KB).

alignment_size

The *alignment_size* should always be equal to the *I/O_size* unless you transfer the clips to a system with a different alignment size. In that case, use the alignment size of the other system, unless you transfer the clips to a system with a larger alignment size.

Using GRIO

High-performance media streaming is of utmost importance in using VST. Guaranteed Rate I/O (GRIO) provides high-performance streaming by enabling a user application to reserve part of a system's I/O resources for its exclusive use. It is used for real-time retrieval and storage of data streams. GRIO is an installable component in the IRIX operating system.

VST assumes that GRIO is running on real-time subvolumes. The supported configuration requires a filesystem with a real-time subvolume for clip storage and uses GRIO for bandwidth management. Other configurations can be used for development purposes or very small systems, but performance is not guaranteed. For example, record and play do not work while using an edit controller with Rice-encoded clips.

VST also assumes that the disks are striped according to the guidelines specified in “Determining Volume Striping Values” on page 10.

This section consists of the following subsections:

For more information about GRIO and its installation, see *IRIX Admin: Disks and Filesystems*.

Determining Disk I/O Size

The first step for using GRIO is deciding between 256 KB or 512 KB I/O per disk:

- Choose 256 KB for lower latency and less memory usage, with relatively reduced performance.
- Choose 512 KB for greater performance but with greater memory usage, resulting in increased latency. This choice makes sense for systems with plenty of memory.

To use GRIO, you must adjust the maximum bandwidth rating for the controllers and the disks by adding REPLACE or ADD lines to the */etc/grio_disks* file; REPLACE is for controllers, ADD is for disks. The next sections explain adjusting maximum bandwidth for SCSI controllers and disks, respectively.

Adjusting the Maximum Bandwidth of the SCSI Controllers

To specify the bandwidth for a fibre channel SCSI controller, add to the */etc/grio_disks* file a REPLACE line with the hardware tree path to the controller. For example:

```
REPLACE /hw/module/1/slot/io7/MotherBoard/pci/3/scsi_ctlr/0 2048K 40
REPLACE /hw/module/1/slot/io7/MotherBoard/pci/1/scsi_ctlr/0 2048K 40
```

The fibre channel SCSI controller is thus capable of 80 MB/sec each, or 40 I/Os of 2 MB/sec each. The recommended value is 80 MB/sec.

Use the command *grio -P -d disk_devicename* to determine the hardware graph path name for the SCSI controller. The controller is the second hardware graph name listed. For example:

```
matrix 15# grio -P -d /dev/dsk/dks0d1s7
Device Path:

/hw/module/1/slot/io1/MotherBoard/pci/0/scsi_ctlr/0/target/1/lun/0/disk
        /hw/module/1/slot/io1/MotherBoard/pci/0/scsi_ctlr/0
        /hw/module/1/slot/io1/MotherBoard/pci/controller
        /hw/module/1/slot/MotherBoard/node/xtalk/0/xbow
        /hw/module/1/slot/MotherBoard/node

vcpqa5 2# grio -P -d /dev/dsk/dks4d2s7
Device Path:
        /hw/module/1/slot/io2/fibre_channel/pci/1/scsi_ctlr/0/target/2
           /lun/0/disk
        /hw/module/1/slot/io2/fibre_channel/pci/1/scsi_ctlr/0
        /hw/module/1/slot/io2/fibre_channel/pci/controller
        /hw/module/1/slot/MotherBoard/node/xtalk/0/xbow
        /hw/module/1/slot/MotherBoard/node
```

Adjusting the Maximum Bandwidth of the Disks

To adjust the maximum bandwidth of the disks, insert an ADD line into the */etc/grio_disks* file to specify disk bandwidth. Note the following:

- Disk I/O size must have the same value as the stripe unit size of the filesystem on the disk.
- The SCSI identifier must match the SCSI identifier of the attached RAID array. If the identifiers do not match, GRIO and therefore VST are unable to start; playout and recording from that disk are impossible.

Obtaining the RAID Array SCSI Identifier

The SCSI identifier in the ADD line must match the SCSI identifier of the attached RAID. If the identifiers do not match, GRIO and therefore V ST are unable to start; playout and recording from that disk is impossible.

To identify the SCSI ID of attached RAID array or other SCSI disk, follow these steps:

1. Use the *versions* command to make sure that the XFS GRIO subsystem, *eoe.sw.xfsrt*, is installed.

   ```
   % versions eoe.sw.xfsrt
   I  eoe.sw.xfsrt 02/24/97  XFS Realtime & Guaranteed-Rate Support
   ```

2. Use the *hinv* command to produce a list of the disks in the system.:

   ```
   # hinv -c disk
   Integral SCSI controller 3: Version Fibre Channel AIC-1160, revision 2
     Disk drive: unit 1 on SCSI controller 3
   Integral SCSI controller 2: Version Fibre Channel AIC-1160, revision 2
     Disk drive: unit 1 on SCSI controller 2
   Integral SCSI controller 0: Version QL1040B (rev. 2), single ended
     Disk drive: unit 1 on SCSI controller 0
     Disk drive: unit 2 on SCSI controller 0
     Disk drive: unit 3 on SCSI controller 0
     Disk drive: unit 4 on SCSI controller 0
     Disk drive: unit 5 on SCSI controller 0
   Integral SCSI controller 1: Version QL1040B (rev. 2), single ended
   ```

3. Using the information in the previous step, get the SCSI ID with the following command:

   # **fx "dksc(***c*,*n***)"**

   where *c* is the SCSI controller number, and *n* is the disk number on that controller.

   For example, the root disk is controller 0 disk 1. An attached RAID should have its own controller and should be the only disk on that controller.

   In the following example, controller 2 is a fibre channel controller and disk 1 is the RAID array (the only disk on the fibre channel card). The drive type is Ciprico Rimfire 7010 03.1.

   ```
   # fx "dksc(2,1)"
   Unable to determine drive type, defaulting to SCSI, ctrlr 0, drive 1
   fx version 6.5, Apr 13, 1999
   ...opening dksc(2,1,0)
   fx: partition already in use by xlv logical volume
   ```

```
fx: devname             seq owner   state
fx: /dev/rdsk/dks2d1s7    1 xlv     part of xlv vol

fx: Warning:  this disk appears to have mounted filesystems.
         Don't do anything destructive, unless you are sure
         nothing is really mounted on this disk.
...drive selftest...OK
Scsi drive type == Ciprico Rimfire 7010    03.1

----- please choose one (? for help, .. to quit this menu)-----
[exi]t               [d]ebug/             [l]abel/
[b]adblock/          [exe]rcise/          [r]epartition/
fx> exit
```

4. Restart GRIO:

   ```
   # /etc/init.d/grio stop
   # /etc/init.d/grio start
   ```

   The system responds:

   ```
   Total number of licensed grio streams is unlimited.
   #
   ```

   When GRIO is misconfigured, it fails to start and gives a brief description of the problem.

5. When the GRIO daemon runs, check the bandwidth for the storage devices:

   ```
   # grio -C -d /dev/dsk/devname
   ```

   The system responds with the bandwidth information; for example:

   ```
   GRIO information for path of  disk device /dev/dsk/devname

   GRIO information for device:
   /hw/module/1/slot/io7/xbox_dualxtown/pci/0/scsi_ctlr/0/target/2/lun/
   0/disk
   opt i/o size:  4194304 bytes
   reservations:    1 ops currently reserved (max. per quantum),   10
   ops maximum
      bandwidth:    36864 kbytes/sec available,    40960 kbytes/sec
   maximum
   ```

```
GRIO information for device:
/hw/module/1/slot/io7/xbox_dualxtown/pci/0/scsi_ctlr/0
opt i/o size:  4194304 bytes
reservations:    1 ops currently reserved (max. per quantum),   25
ops maximum
   bandwidth:    98304 kbytes/sec available,   102400 kbytes/sec
maximum
```

Inserting the ADD Line for Disk Bandwidth

Insert an ADD line into the */etc/grio_disks* file to specify disk bandwidth. Set the I/O size to the same value as the stripe unit size of the filesystem on that disk.

For example, if you have a Ciprico 7000 fibre channel RAID storage system, you might use the following line:

```
ADD     “Ciprico Rimfire 7010    03.1” 2048K 22
```

This line says that the RAID array should produce 44 MB/sec, or, in this case, 22 I/Os at 2 MB/sec. (The recommended value is 44 MB/sec.) The filesystem on this disk has a stripe size of 2 MB.

You can insert multiple ADD entries for systems that have filesystems with different stripe unit sizes on the same type of disk.

Now the user application can be started. Files created on the real-time subvolume can be accessed using guaranteed-rate I/O.

Disabling GRIO

Although using GRIO is recommended, VST enables you to disable it with the following control setting in */usr/vtr/config/system-defaults*:

```
vtr.storage.fs.grio.enabled false
```

Installing Video Server Toolkit

This chapter provides a step-by-step description of installing the Video Server Toolkit (VST) software in the following sections:

Preparing for Installation

Before you install VST, make sure you have the materials required for the installation and that your workstation meets all the VST requirements. Follow these procedures:

1. To install VST, you need the following CDs:
   - Video Server Toolkit CD
   - IRIX 6.5.4

   Note: VST was tested for compatibility with specific hardware and software configurations. Refer to the VST release notes for the configurations supported for this release.
2. The product release notes contain the latest information for the software required to use VST, such as required patches. To read the release notes from the CD included with the product, become superuser and enter the following command:

   ```
   # /CDROM/CDgrelnotes
   ```
3. Before installing VST, install the patches required for VST, as specified in the release notes. Use Software Manager or the *inst* program to install any required patches.
4. To install VST over a network, you must be successfully connected to a network. For more information about connecting to a network, consult your system administrator.

Installing VST

Once your server is properly configured, use the following procedure for installing VST over a network or from a local CD.

1. Become a superuser by entering the following command:

   % **su**

2. Make sure that the *sysadmdesktop.sw* subsystem is present. This subsystem is included in the IRIX distribution.

3. Start the installation program and identify the source of the VST image, as follows:

   # **inst -f** *servername*:*distribution_directory*

   For example, if you are installing over the network, use a line similar to the following:

   ```
   # inst -f host1 :dist/vst/dist
   ```

   Or, if you are installing from a CD, use

   # **inst -f cdHost:/CDROM/dist**

   Or, if you are installing from a local CD, use

   # **inst -f /CDROM/dist**

4. List the subsystems to install as follows:

   ```
   Inst> l
   ```

   This command lists all the subsytems to install; Table 3-1 summarizes them.

Table 3-1 VST Default and Optional Subsystems

Systems and Subsystems	Default	Description
vst_eoe	Yes	Video Server Toolkit 1.2
vst_eoe.books which includes		VST 1.2 online documentation
vst_eoe.books.AdminGuide		*Video Server Toolkit Installation and Administration Guide* (this manual, online version)
vst_eoe.books.AdminGuideHTML		*Video Server Toolkit HTML Installation and Administration Guide* (online version)
vst_eoe.books.DevelGuide		*Video Server Toolkit Developer's Guide* (online version)
vst_eoe.books.DevelGuideHTML		*Video Server Toolkit HTML Developer's Guide* (online version)
vst_eoe.man which includes	Yes	VST 1.2 man pages
vst_eoe.man.base		VST 1.2 base man pages
vst_eoe.man.ftpd		VST 1.2 man pages for enhanced FTP daemon
vst_eoe.man.relnotes		VST 1.2 release notes
vst_eoe.man.tools		VST 1.2 base applications man pages
vst_eoe.sw which includes		Video Server Toolkit 1.2 software
vst_eoe.sw.base		VST 1.2 base software
vst_eoe.sw.clipmirror		VST 1.2 clip-mirroring module
vst_eoe.sw.diaquest		VST 1.2 Diaquest deck-control module
vst_eoe.sw.divo		VST 1.2 DIVO video module
vst_eoe.sw.dvb-vg		VST 1.2 Viewgraphics DVB/MPEG module
vst_eoe.sw.failsafe		VST 1.2 configuration files for IRIS FailSafe 1.2

Table 3-1 (continued) VST Default and Optional Subsystems

Systems and Subsystems	Default	Description
vst_eoe.sw.fsmon	Yes	VST 1.2 File System Import Monitor (FSmon)
vst_eoe.sw.ftpd	Yes	VST 1.2 enhanced FTP daemon
vst_eoe.sw.horita		VST 1.2 Horita time-code input module
vst_eoe.sw.little-red		VST 1.2 Miranda Little Red time-code input module
vst_eoe.sw.louth		VST 1.2 Louth VDCP control module
vst_eoe.sw.mpeg		VST 1.2 MPEG-2 tools
vst_eoe.sw.o2video		VST 1.2 O2 digital video option module
vst_eoe.sw.sony		VST 1.2 Sony protocol module
vst_eoe.sw.studiocentral		VST 1.2 StudioCentral 2.0 archive interface
vst_eoe.sw.tools	Yes	VST base applications
vst_eoe.sw.vela		VST Vela SCSI MPEG-2 decoder module

5. Mark for installation any optional VST related software you might need. For example, if you are using a DIVO board, or the DVB-ASI, Louth, MPEG, Sony, or Vela protocols, install one or more of the following options, as appropriate:

```
Inst> i vst_eoe.sw.divo
Inst> i vst_eoe.sw.dvb-vg
Inst> i vst_eoe.sw.louth
Inst> i vst_eoe.sw.mpeg
Inst> i vst_eoe.sw.sony
Inst> i vst_eoe.sw.vela
```

6. Initiate the installation, as follows:

```
Inst> go
```

7. Quit the installation program, as follows:

```
Inst> quit
```

8. To run a secondary server that takes over VST operations when the primary server fails, repeat all preceding steps for the secondary server(s). Following instructions in Chapter 5, “Configuring for Redundancy,” install IRIS FailSafe or configure the secondary server(s) for clip mirroring.

Confirming the Installation

To confirm the successful installation of VST, complete the following steps:

1. Start VST by entering the following command as a superuser:

   # **/usr/vtr/bin/vtrstart**

   Note: On an O2 workstation, the VST window is displayed.

2. To see if VST started, enter

   # **/usr/vtr/bin/vtrstat**

   (The utility *vtrstat* is described in "vtrstat" on page 104 in Chapter 8.) If VST is running, the response is as follows:

   Video server on *hostname* is running.

3. Verify the media ports are supported by entering

   # **/usr/vtr/bin/vtrstat -ports**

   vtrstat lists the supported ports; for example:

```
#  Port    Type    Description
-----------------------------------------------------------
0  dq_0    Deck    Diaqusest Deck Control
1  DIVO_0  Video   SGI XT-DIVO Digital Video Option
2  DIVO_1  Video   SGI XT-DIVO Digital Video Option
3  DIVO_3  Video   SGI XT-DIVO Digital Video Option
4  DIVO_4  Video   SGI XT-DIVO Digital Video Option
5  DIVO_5  Video   SGI XT-DIVO Digital Video Option
6  DIVO_6  Video   SGI XT-DIVO Digital Video Option
7  DIVO_7  Video   SGI XT-DIVO Digital Video Option
```

You can also confirm the installation by examining the server log for startup messages: the contents of the */usr/vtr* and */var/adm/vtr/logs* directories:

```
# more /usr/vtr/adm/logs/vtrlog
```

Logging begins with output like the following:

```
I 22-10:46:31.484095  55938 ---- LOGGING STARTS ----
I 22-10:46:31.484885  55938 VCP-Recorder starting
I 22-10:46:35.979391  55938 System:
I 22-10:46:35.979682  55938   Platform: Origin 200 GIGAchannel
(hostname=oo7 serial=690d416b)
I 22-10:46:35.979809  55938   Processors: 2x225MHz R10000 (IP27)
I 22-10:46:35.979970  55938   Secondary Cache: 2 Mbytes
I 22-10:46:35.980128  55938   Memory: 512 Mbytes
I 22-10:46:35.980261  55938 Video devices:
I 22-10:46:35.980377  55938   DIVO_DVC_0 (module=1/GIGAchannel slot=12
serial=HTM962 part=030-1387-001E)
I 22-10:46:35.980501  55938     DVCPRO_CODEC (serial=HTM769
part=030-1388-001B)
I 22-10:46:35.980756  55938     DVCPRO_CODEC (serial=HTM802
part=030-1388-001B)
I 22-10:46:35.981005  55938   DIVO_DVC_1 (module=1/GIGAchannel slot=13
serial=HTM970 part=030-1387-001E)
I 22-10:46:35.981181  55938     DVCPRO_CODEC (serial=HTM789
part=030-1388-001B)
I 22-10:46:35.981586  55938     DVCPRO_CODEC (serial=HTM810
part=030-1388-001B)
I 22-10:46:35.981718  55938   DIVO_DVC_2 (module=1/GIGAchannel slot=14
serial=HJH608 part=030-1387-001E)
I 22-10:46:35.981844  55938     DVCPRO_CODEC (serial=HJH541
part=030-1388-001B)
I 22-10:46:35.981964  55938     DVCPRO_CODEC (serial=HJH546
part=030-1388-001B)
I 22-10:46:35.982120  55938   DIVO_DVC_3 (module=1/GIGAchannel slot=15
serial=HJH287 part=030-1387-001E)
I 22-10:46:35.982346  55938     DVCPRO_CODEC (serial=HJH644
part=030-1388-001B)
I 22-10:46:35.982479  55938     DVCPRO_CODEC (serial=HJH553
part=030-1388-001B)
I 22-10:46:35.982599  55938 Audio devices:
I 22-10:46:35.982713  55938   RAD1 (module=1/GIGAchannel pci=2)
I 22-10:46:35.982830  55938   RAD2 (module=1/O200 pci=5)
I 22-10:46:35.982948  55938   RAD3 (module=1/O200 pci=6)
I 22-10:46:35.983206  55938   RAD4 (module=1/O200 pci=7)
```

The *crash* directory contains crash files for the VST executable *vvtr*.

Table 3-2 describes directories in */usr/vtr*.

Table 3-2 /usr/vtr Subdirectories

Subdirectory	Description
*bin/**	Commands and the VST executable *vvtr*
*clips/**	Clip cache
*config/**	Configuration files
*data/**	Files in this directory are static data or image files used by various VST components
*images/**	The directory in which still images are created when the MediaCache VTR Control Protocol (MVCP) CIMG command is executed (this command is explained in the *Video Server Toolkit Developer's Guide*
*index/**	Clip indices (MPEG and vframe formats)
install/	Files used during installation
lib/perl	Perl modules
lib32/.so*	Core and shared libraries
lib32/modules/.so*	External interface device/control/format modules

Configuring Video Server Toolkit

This chapter describes how to configure VST in the following sections:

Note: To configure startup options, see "Starting and Stopping VST" on page 97 in Chapter 8.

Configuring the Device Interface

Device files contain variables called controls that influence the behavior of specific devices. Each device in your system can have only one device file.

Configuring Audio and Video Devices in Your System

The directory */usr/vtr/config/device-defaults* contains files that correspond to the media devices used by the VST; these files define machine-specific behavior. Each device or device type has one file; the directory *device-defaults* can contain the following files:

- *MEDIA*, for behavior common to all connected media devices
- *mvp*, for an O2 workstation

  Because an O2 workstation can only take one analog or digital video card, other *mvp* names, such as *mvp_0*, are not needed.
- *DIVO*, for all DIVO boards in a system, or *DIVO_0*, *DIVO_1*, *DIVO_2*, and so on to name a specific DIVO board in a VST server
- *DIVO_DVC*, for DIVO-DVC boards
- *vela*, for all Vela devices connected to a system, or *vela_0*, *vela_1*, *vela_2*, and so on. if more than one Vela device is in use in a VST server
- *dq*, or *dq_0*, *dq_1*, *dq_2*, and so on. when VST is using Diaquest software to control external VTRs
- *dvb-asi*, for media devices connected to a DVB-ASI adapter board

Setting Device Parameters

You use VST controls to configure video compression modes, audio sampling parameters, clip formats, and many other elements of the system configuration. For example, for the first DIVO_DVC device, you would create in the */usr/vtr/configd/device-defaults* directory a file called *DIVO_DVC_0*. This file could contain, for example:

```
# Format
vtr.media.video.input.compression.type dvcpro
vtr.media.clip.format movie/dif

# Audio
vtr.media.audio.input.port RAD1.AESIn
vtr.media.audio.output.port RAD1.AESOut
```

This example device defaults file sets the default control values for any units created using the DIVO_DVC_0 port so that new clips are recorded with DVCPRO compression and the audio input and output uses the AES interfaces on the Digital Audio Option board.

Configuring System Defaults

You use VST controls in files in the */usr/vtr/config/system-defaults* directory to specify default system-wide behavior. For example, the file *main* might contain the following:

```
vtr.main.timing_standard 625
```

This line sets the default timing mode for the VST system to 625.

Check the following controls depending on your hardware configuration:

- vtr.media.dbv-asi.seamless_playback: turns on splicing when set to true
- vtr.media.dbv-asi.halt_on_underflow: set to false
- vtr.media.dvb-asi.port.total_bitrate: set this control to the total bit rate required by the downstream MPEG decoder or multiplexer (for example, 52000000)

Refer to Appendix A, “Video Server Toolkit Control Summary,” or to the vst-controls(5) man page for detailed information about all system controls.

Configuring Audio Support

The Vela decoder output includes audio outputs decoded from the MPEG-2 stream. The VST control for adjusting the Vela audio output level is vtr.media.audio.output.level. This control and other Vela controls are summarized at “Device Controls for MPEG Decoder (Vela) Ports” on page 133 in Appendix A and are described fully in the vst-controls(5) man page.

For the DVB outputs, the MPEG-2 stream is transmitted in its compressed form, including the compressed audio.

For the DIVO, DIVO-DVC, and O2 outputs, a unit can select the audio port for:

- audio output during playback
- audio input during recording

The controls that configure the audio port are

- vtr.media.audio.output.port (default DefaultOut)
- vtr.media.audio.input.port (default DefaultIn)

The default audio input and output ports are selected with the IRIX Audiopanel control panel. Audio configuration is managed by the decoder.

For more information about audio ports, see the vst-controls(5M) man page.

Identifying Audio Ports

The default input audio port is DefaultIn, meaning the audio is taken from the default input as set in the server’s audio panel. Similarly, the default output audio port is DefaultOut.

To change the input or output audio port, set either vtr.media.audio.input.port or vtr.media.audio.output.port, respectively. Audio port names take the form [*subsystem.*]*interface*. You must specify subsystem if the server has more than one audio device installed.

For audio ports on the SGI digital Audio Option PCI board:

- *subsystem* is RAD*n*
- *interface* is AESIn, AESOut, ADATIn, or ADATOut

Each Digital Audio Option board has a unique subsystem name, such as RAD1 or RAD2. To determine the name of a board in a particular slot, use *vtrhwinfo*. Here is a slightly abbreviated example:

```
# /usr/vtr/bin/vtrhwinfo
System:
  Platform: Origin 200 GIGAchannel (hostname=system1 serial=xxxx)
  Processors: 2x225MHz R10000 (IP27)
  Secondary Cache: 2 Mbytes
  Memory: 512 Mbytes
Video devices:
  DIVO_DVC_0 (module=1/GIGAchannel slot=12 serial=HTM962
part=030-1387-001E)
    DVCPRO_CODEC (serial=HTM769 part=030-1388-001B)
    DVCPRO_CODEC (serial=HTM802 part=030-1388-001B)
  DIVO_DVC_1 (module=1/GIGAchannel slot=13 serial=HTM970 .
.
.
.
Audio devices:
  RAD1 (module=1/GIGAchannel pci=2)
  RAD2 (module=1/O200 pci=5)
Audio input ports:
  DIVO_DVC_0.DigitalIn
  DIVO_DVC_1.DigitalIn
  DIVO_DVC_2.DigitalIn
  DIVO_DVC_3.DigitalIn
  RAD1.AESIn
  RAD1.ADATIn
  RAD2.AESIn
  RAD2.ADATIn
Audio output ports:
  DIVO_DVC_0.DigitalOut
  DIVO_DVC_1.DigitalOut
  DIVO_DVC_2.DigitalOut
  DIVO_DVC_3.DigitalOut
  RAD1.AESOut
  RAD1.ADATOut
  RAD2.AESOut
  RAD2.ADATOut
```

```
Serial ports:
  tty1 (module=1/O200 built-in)
  tty2 (module=1/O200 built-in)
  tty3 (module=1/GIGAchannel slot=11)
  ...
```

For example, to take audio input for the DIVO_1 port from the AES coaxial input on the Digital Audio Option board identified as RAD1, open the file */usr/vtr/config/device-defaults/DIVO_1*, and add this control setting:

```
vtr.media.audio.input.port RAD1.AESIn
```

Disabling Audio

If no audio input is connected, you must disable audio before recording. Add the following control setting to the appropriate device-defaults file (for example, */usr/vtr/config/device-defaults/DIVO_DVC_2*):

```
vtr.media.audio.input.port ""
```

Configuring VST for Control by Remote Devices (control-in.conf File)

Use the */usr/vtr/config/control-in.conf* file to configure VST so that it can be controlled by remote devices. VST supports devices that use either the Louth Video Disk Communications Protocol (VDCP) or the industry-standard Sony-compatible VTR RS-422 control protocol.

The */usr/vtr/config/control-in.conf* file has three parts:

- control port (serial port) configuration: one line per device
- other control ports: one line per device
- signal port configuration: one line per device

The control port and signal configuration lines are described in separate subsections below.

Control Port Configuration Line

You use the configuration file */usr/vtr/config/control-in.conf* to specify basic connection parameters between the controlling devices (Sony and Louth) and VST. Connection parameters include such information as which serial port controls which video board and how.

For each serial port to be connected to a controlling device (such as an automation or edit controller), insert a line in */usr/vtr/config/control-in.conf* that uses the following format:

```
protocol   type  port  speed   parity     signalport    rate    latency
```

The following line shows a sample configuration:

```
sony    rs422    4    38400    1    1    29.97    0
```

This example specifies that a controlling device using the Sony-compatible VTR protocol is connected via an RS-422 serial line to serial port 4 on the server. The serial connection speed is 38400 bits / sec, using odd parity. The device is controlling the video port identified by the “signal 1” configuration line (also specified in *control-in.conf*) and is using a zero-frame command latency, so that commands are executed as quickly as possible. The preroll should always be 0, except for Louth controllers, for which the preroll is 3.

The following subsections describe each variable in this configuration line.

protocol

Use the value that corresponds to the protocol used by the device you are configuring to work with VST. The valid values of *protocol_name* include:

- sony, p2: Sony-compatible VTR RS-422 control protocol
- louth: Louth Video Disk Communications Protocol
- mvcp: SGI Multiport Video Computer Protocol
- hsip: Horita Serial Interface Protocol
- littlered: Miranda Little Red linear time-code reader protocol

type

The valid values of *type* include:

- rs232: RS-232 control connection
- rs422: RS-422 control connection
- tcp,: socket-based TCP / IP connection

These values must correspond to the type of connection between the control video device and the VST server. The type of cable you use to connect the video device and the server determines this value.

Note: The value rs422 is not supported on the O2 server.

port

The value of *port* specifies the serial port number for RS-232 and RS-422 connections. For example, to specify that the device is connected to port 2 (sometimes referred to as ttyd2), specify just the numeral 2. For TCP connections, *port* specifies the TCP port number.

Note: Serial and TCP ports (communications ports) are unrelated to video ports (signal ports).

speed

The *speed* value specifies the connection speed of the serial connection between the server and the controlling device, usually 38400. On rare occasions, 9600 is used. For TCP connections, enter a hyphen (-) for this setting.

parity

The optional *parity* setting can have one of the following values:

- 0, indicating no parity
- 1, indicating odd parity
- 2, indicating even parity

Parity is usually odd (1). On rare occasions, even or no parity is used.

signalport

The *signalport* value specifies the signal (video) port as identified by a signal configuration line in the *control-in.conf* that is controlled by this remote device connection. For more information, see “Signal Configuration Line” on page 54.

rate

The *rate* value specifies the frame-rate and drop-frame mode for time codes exchanged over the control connection. Valid values are:

- 25: 625 / 50 video
- 29.97: 525 / 59.94 video (drop-frame)
- 30: 525 / 59.94 video (non-drop-frame)

latency

When VST receives a command, the command can apply only to the next frame to enter the video port output queue. For example, if the video port output queue holds three frames, the first frame that can be affected by an incoming command is the third frame in the queue. If a stop command is received, the first two frames in the video port output queue are displayed; the third is not displayed because it is affected by the stop command.

You can modify the number of frames that display before a command begins by setting the *latency* value. Valid values are 0 or positive integers. A value of 0 starts the command’s effect as soon as possible, but the command latency is always greater than 0.

The minimum command latency is the sum of the fixed latency value for the specific compression format and the value of the vtr.media.video.output.max_queued_frames control, which is usually 1 (see the description of this control in the man page, vstcontrols (5M)). To guarantee that all commands are executed in a frame-accurate manner, add 1 to this sum, as follows:

- fixed latency value for the compression format:
  - DVCPRO: 5 frames
  - Rice: 2 frames
  - uncompressed: 2 frames
  - JPEG: 2 frames
- plus 1 (usual value of the vtr.media.video.output.max_queued_frames control)
- plus 1 (to guarantee that all commands are executed in a frame-accurate manner)

The result is 7 frames for DVCPRO, 4 for all others.

A small number for *latency* increases the risk that a frame may be missed (repeated) because of system load. A large number increases the latency in responding to transport commands such as play, stop, or pause.

Some commands in the Sony protocol imply a predefined frame delay; *latency* does not affect these commands.

For the MVCP TCP port, make sure that the *control-in.conf* file includes a control port configuration line:

```
mvcp tcp 5250
```

5250 is the standard MVCP control port number. The initial *control-in.conf* file installed with the VST software includes this line.

Signal Configuration Line

The signal configuration line specifies how to map signal port numbers to video port names. A signal configuration line is required for each video port for it to be controlled by an external device. For example, to specify that signal 1 corresponds to the video port known as DIVO_4, add the following line:

```
signal 1 DIVO_4
```

Example control-in File

Here is a sample *control-in.conf* file:

```
# This defines the normal VST MVCP TCP port

mvcp tcp 5250

# Other control ports

#sony   rs422 2 38400 1 1 29.97 3
louth   rs422 2 38400 1 1 29.97 3
louth   rs422 5 38400 1 2 29.97 3
#
louth   rs422 6 38400 1 3 29.97 3
louth   rs422 7 38400 1 4 29.97 3
#louth   rs422 8 38400 1 5 29.97 3
#
#
# Signal ports
#
#   Each signal (video) port is mapped from a port number to port name
#   with a line of this form:
#
#   signal <port number> <port name>
#
#     port number:      Disk port number (> 0)
#     port name:        VST port name (e.g., DIVO_1)

signal 1 DIVO_DVC_0
signal 2 DIVO_DVC_1
signal 3 DIVO_DVC_2
signal 4 DIVO_DVC_3
#signal 5 DIVO_4
#signal 2 DIVO_3
```

Configuring VST to Control Other Devices (control-out.conf File)

Use the */usr/vtr/config/control-out.conf* file to configure VST so that it can control other VTR-like devices.

Use this file also to specify basic connection parameters between VST and the VTRs to be controlled. Connection parameters include such information as which serial port controls which VTR and how. For each VTR controlled by VST using the Sony protocol, add a line in */usr/vtr/config/control-out.conf* that uses the following format:

```
protocol    connection    serial_port    rate    parity
```

The following line shows a sample configuration:

```
sony    rs422    4    38400    1
```

The line has the same meaning when p2 is substituted for sony.

The following sections describe each variable in this configuration line.

protocol

Use the value that corresponds to the device you are configuring to work with VST. The valid values of *protocol_name* are sony and p2, which are equivalent.

connection

The valid values of *connection* include:

- rs232, for a RS-232 control connection
- rs422, for a RS-422 control connection

These values must correspond with the type of connection between the video device and the VST server. The device and the VST server must agree on the serial connection; the cable between them must be appropriate for that connection.

Note: rs422 is not supported on the O2 system.

serial_port

The *serial_port* value specifies the serial port as identified by a serial configuration line in the *control-in.conf* that is controlled by this remote device connection.

Note that The RS-422 pinout at the DB-9 end of the SGI adapter cable is nonstandard; see Appendix B, “RS-422 Pinouts.”

speed

The *speed* value specifies the connection speed of the serial connection between the server and the controlling device, usually 38400. On rare occasions, 9600 is used. For TCP connections, enter a hyphen (-) for this setting. Sony devices must use 38400.

parity

The optional *parity* setting can have one of the following values:

- 0, indicating no parity
- 1, indicating odd parity
- 2, indicating even parity

The parity must be specified as odd for the currently supported protocols.

Configuring the StudioCentral Interface

VST can be configured to exchange clips with the SGI StudioCentral media management system. VST treats StudioCentral as an archive system to which clips can be saved after VST records them, or from which clips can be retrieved for playback.

The StudioCentral interface module is not selected by default for installation with VST. The *vst_eoe.sw.studiocentral* subsystem must be installed for clip media and metadata to be exchanged with one or more StudioCentral repositories.

For a general description of the archive and how to use it, see the *Video Server Toolkit Developer’s Guide*, which provides an overview of the VST server, ATS server, and archive server.

The configuration file, */usr/vtr/config/studiocentral.conf* specifies the StudioCentral repositories that VST can access as archives. Each line of this file has the following syntax:

repository `ats`

where *repository* is the hostname of the server running the StudioCentral repository. For example, the following configuration file specifies two StudioCentral repositories available to the VST server for archiving:

```
mymediaserver ats
workgroupmedia ats
```

In this example, `mymediaserver` and `workgroupmedia` are the names of the StudioCentral repository servers.

Mapping Physical Ports to Logical Port Names

You can use the */usr/vtr/ports.conf* file to create logical names for the physical video port s on the VST server. For each port, include a line in the configuration file with this syntax:

`port` *physicalname logicalname description*

For example:

```
port DIVO_DVC_0 ProgA “Program A”
port DIVO_DVC_1 ProgB “Program B”
port DIVO_DVC_2 Preview “Preview Output”
port DIVO_DVC_3 Archive “Archive I/O”
```

Configuring for 625/50 Systems

VST supports both 525 / 59.94 and 625 / 50 operation simultaneously. A few operating characteristics default to 525 / 59.94, including the MVCP timing mode (refer to the MVCP *FRAT* command documentation in the latest *Video Server Toolkit Developer's Guide*) and the default video output timing.

To change the default timing, include the following control setting in the system defaults file */usr/vtr/config/system-defaults/main*:

```
vtr.main.timing_standard 625
```

Note: The system timing standard controls the default video output timing only if the control vtr.media.video.output.timing is not changed from its default value of “system”.

In */usr/vtr/config/device-defaults/DIVO*, make sure vtr.media.video.output.timing is set to its default value, system.

Tuning the Operating System for VST (Setting the Maximum DMA Size)

The maximum DMA size limits the maximum amount of data that can be transferred to or from the filesystem in a single operation. The default maximum DMA size is 4 MB, which is too small for most VST installations. To increase the system’s maximum DMA size, use the following procedure:

1. As superuser, start *systune* to change the DMA size:

   ```
   # systune -i
   ```

   The system responds:

   ```
   Updates will be made to running system and /unix.install

   systune->
   ```

2. Enter the new DMA size:

   ```
   systune-> maxdmasz bytes
   ```

   The following example shows 10 MB:

   ```
   systune-> maxdmasz 641
   ```

   The system responds:

   ```
        maxdmasz = 257 (0x101)
       Do you really want to change maxdmasz to 640? (y/n)
   ```

3. Enter **y** for yes. The system responds:

   ```
   In order for the change in parameter maxdmasz to become effective,
   reboot the system
   ```

4. Quit *systune*:

   ```
   systune-> quit
   ```

5. Reboot the server:

   ```
   # reboot
   ```

Configuration Files

Table 4-1 summarizes the files that VST uses so that it works with all the hardware and software in the system.

Table 4-1 Configuration Files

Configuration File	Purpose
/usr/vtr/config/control-in.conf	Configures VST to be controlled by remote devices. Use this file to configure VCP to work with controllers compatible with Sony and Louth protocols, and Miranda or Horita time-code reader devices.
/usr/vtr/config/control-out.conf	Configures VST to control VTRs or VTR-like devices.
/usr/vtr/config/control-out.conf	Configures VST to control other decks. Use this file to configure VST to control VTRs using Diaquest software.
/usr/vtr/config/device-defaults	This directory contains files that specify default control settings for audio, video, and disk devices. The names and contents of the files are described in Appendix A, “Video Server Toolkit Control Summary.”
/usr/vtr/config/system-defaults	This directory contains files that specify default control settings for various VST non-device subsystems (for example, system-wide controls or Sony protocol control ports).
/usr/vtr/config/studiocentral.conf	Specifies StudioCentral repositories that VST can access as archive systems.
/usr/vtr/config/ports.conf	Specifies logical video port names to substitute for the ports’ physical names.
/usr/vtr/config/vtrd.conf	VST server daemon configuration.
/usr/vtr/config/vtr_ioconfig.conf	Contains configuration information for the Vela decoder. For more information, see “Configuring the Vela Decoder” on page 89.
vtrfsinfo.conf	Defines I/O size and alignment size for non-real-time filesystems.
/usr/vtr/config/system-defaults/clipmirror	Stores the name of the primary server with which redundant server synchronizes, and other parameters. See “Configuring a Server for Clip Mirroring” on page 68 in Chapter 5.

Configuring for Redundancy

If you plan to use a dual-server configuration so that a secondary server can take over if the primary server fails, you can use IRIS FailSafe version 1.2. Alternatively, you can set up one or more servers for clip mirroring, with clip caches that mirror the clip cache on a designated primary server. This chapter describes the following:

IRIS FailSafe 1.2 and VST

IRIS FailSafe 1.2 is an SGI software product that allows a pair of servers to be used in a redundant configuration so that access to a storage device, or set of storage devices, can be transferred from the primary server to the backup server should the primary server fail.

Storage devices, such as RAID arrays that can store clips and index filesystems, are physically attached to the two servers in the system, but are owned and accessed by one server at a time. The two servers and the storage form an IRIS FailSafe 1.2 cluster.

Note: Setting up servers and shared storage for IRIS FailSafe 1.2 is performed by qualified SGI System Support Engineers (SSEs) only.

The servers also share a public IP alias or a name users can use to connect to the servers. If a server fails, the other server takes over the shared storage as well as this IP alias, which users still use to connect to the servers. To the user, a failover looks the same as a server that was unavailable for a number of seconds but returned to service thereafter.

The servers communicate via a private serial connection. When the standby server detects a problem with the active server, IRIS FailSafe 1.2 unmounts the clip filesystem from the active server and mounts it on the standby server. The standby VST server detects this action, adds the clips to its internal tables, and becomes the active server, ready to play or record.

Note: VST does not automatically begin playing or recording the material that was being played or recorded on the active server when it failed. The application(s) or automation system must restart the operations.

Installing IRIS FailSafe 1.2 for VST

This section breaks the instructions for installing and configuring IRIS FailSafe 1.2 for use with VST into the following subsections:

Performing Preliminary Steps

Follow these steps:

1. Have ready the following:
   - *IRIS FailSafe Administrator's Guide* (document number 007-3109-003 or later). Do not use the *IRIS FailSafe 1.2 Administrator's Guide*
   - *IRIX Admin: Disks and Filesystems* (document number 007-2825-005 or later)
   - IRIS FailSafe 1.2 CD and associated CD with required patches.

2. Install the IRIS FailSafe 1.2 software from the distribution CD:

```
# inst -f /CDROM/dist
Inst> keep *
Inst> install ha ha_www ha_fsconf
Inst> go
Inst> quit
```

3. Install the IRIS FailSafe 1.2 subsystem *vst_eoe.sw.failsafe* from the main VST images on the VST CD, if this subsystem was not selected when VST was originally installed.

```
# inst -f /CDROM/dist
Inst> install vst_eoe.sw.failsafe
Inst> go
Inst> quit
```

4. Install the latest IRIS FailSafe 1.2 patch from the patches CD:

```
# inst -f /CDROM/patches/patch3582/dist
Inst> install *
Inst> go
Inst> quit
```

5. Set the system variables on the primary server:

```
# nvram AutoLoad Yes
# nvram scsihostid 0
# chkconfig failsafe off
```

6. Set the system variables on the secondary server; the recommended value for the SCSI host ID is 8.

```
# nvram AutoLoad Yes
# nvram scsihostid 8
# chkconfig failsafe off
```

Configuring Network Interfaces and Addressing

Follow these steps:

1. From your network administrator, obtain for each server a private hostname, a public hostname, and an IP address. The two private IP addresses and the public IP address should be on the same subnet.
2. Configure the network interfaces on the primary server by editing the */etc/hosts* file to include the following:
   - a private network between the servers (not broadcast)
   - hostname and address for each server
   - a single hostname and address for the entire IRIS FailSafe 1.2 configuration (also called the public IP address)

   Example 5-1 shows example entries in the */etc/hosts* file.

Example 5-1 Example /etc/hosts File

```
# This entry must be present or the system will not work.
128.64.0.1       localhost

# Primary FailSafe Host
192.70.0.100  fsprimary.engr.sgi.com fsprimary
192.0.3.1       priv-fsprimary.engr.sgi.com priv-fsprimary

# Secondary FailSafe Host
192.70.0.101  fssecondary.engr.sgi.com fssecondary
192.0.3.2       priv-fssecondary.engr.sgi.com priv-fssecondary

# FailSafe Alias for single system image
192.70.0.109 fssystem.engr.sgi.com fssystem
```

3. Place the same network information in the */etc/hosts* file on the secondary server.
4. Add the following entries to the primary server's */etc/config/netif.options* file:

   ```
   if1name=ef0
   if1addr=$HOSTNAME

   if2name=ef1
   if2addr=priv-$HOSTNAME
   ```

5. Place the same information in the */etc/config/netif.options* files on the secondary server.

6. Turn routing off on the primary server by modifying the */etc/config/routed.options* file to contain the following:

   ```
   -h -q
   ```

7. Place the same entry in the */etc/config/routed.options* file on the secondary server.
8. Reboot each server to enable the network changes.
9. Use *ping* from the secondary server to check the connection. For example:

   ```
   # ping fsprimary.engr.sgi.com
   ```

10. Use *ping* from the primary server to check the connection. For example:

    ```
    # ping fssecondary.engr.sgi.com
    ```

11. Open */etc/nsswitch.conf*. If necessary, change the resolution order for hosts to use `files` before either `nis` or `dns`. For example:

    ```
    hosts: files nis
    ```

    Set resolution order the same way on the secondary server.

12. Place the following alias in the */etc/aliases* file to handle administrative messaging from IRIS FailSafe 1.2 on the primary server. Use the email address of the person who administers the system. For example:

    ```
    fsafe_admin:postmaster
    postmaster:fs_admin_person@another_system
    ```

13. Use the *newaliases* command the enable the changes:

    ```
    # newaliases
    ```

14. Place the same alias in the */etc/aliases* file on the secondary server and use *newaliases* to enable the changes.

Setting Up the Serial Connection

Follow these steps:

1. Edit the */etc/inittab* file on the primary server to reserve the private IRIS FailSafe 1.2 serial connection between the servers. Make certain that the t2 entry is as follows:

   ```
   t2:23:off:/sbin/getty -N ttyd2 co_9600          # port 2
   ```

2. Use the *init q* command to enable the serial port changes:

   ```
   # init q
   ```

3. Change the */etc/inittab* file t2 entry on the secondary server to match that of the primary server and use *init q* to enable the changes.

Setting Up Logical Volumes and Filesystems

Follow these steps:

1. Create the XLV logical volumes and XFS filesystems to be shared by following the instructions in *IRIX Admin: Disks and Filesystems*.

   Remember that each XLV logical volume must be owned by the node that is the same as the primary node for running VST. To simplify the management of the node names (or owners) of volumes on shared disks, follow these recommendations:

   - Work with the volumes on the shared storage from the primary node.
   - If you did not use the primary to create the volumes, you can change the node name to the primary node using *xlv_mgr*.

2. Copy */usr/vtr/failsafe/ha.conf* to */var/ha/ha.conf* onto the primary and secondary servers. These files are included in the *vst_eoe.sw.failsafe* subsystem.

3. Copy */usr/vtr/failsafe/chkvtr* to */var/ha/actions/chkvtr* onto the primary and secondary servers. These files are included in the *vst_eoe.sw.failsafe* subsystem.

4. If necessary, change the default mount point for the shared file systems in */var/ha/ha.conf* on the primary and secondary servers.

   By default, the shared filesystem is mounted at */usr/vtr/clips*. If you want to install a separate shared filesystem that stores the indices for the clips, do the following:

   - Add a new filesystem block like the one in the sample file for filesystem *fcraid1* in */var/ha/ha.conf*.
   - Add a new volume block, as shown for volume *fcraid1*.

   See the *IRIS FailSafe Administrator's Guide* for more details.

Configuring Network Interfaces and Addressing for IRIS FailSafe 1.2

Follow these steps on the primary and secondary servers:

1. Change the *fsprimary.engr.sgi.com* entry (default hostname) in */var/ha/ha.conf* to the name of your primary server.
2. Change the *fssecondary.engr.sgi.com* entry in */var/ha/ha.conf* to the name of your secondary server.
3. Change the default IP address, 198.29.66.18, in */var/ha/ha.conf* to point to the IP address of your primary server.
4. Change the default IP address 198.29.66.90 in */var/ha/ha.conf* to the IP address of your secondary server.
5. Change the default netmask values and the *broadcast-addr* variable to the correct values for your network on the primary and secondary servers.

   The *broadcast-addr* variable should point to the broadcast address of the IP subnet on which the servers are present. Both servers must be on the same IP subnet.

Checking the Installation and Starting IRIS FailSafe 1.2

Follow these steps:

1. Perform the following steps as a superuser on the primary and secondary servers:

   ```
   # chown root.sys /var/ha/ha.conf
   # chmod 500 /var/ha/ha.conf
   # /usr/etc/ha_cfgchksum
   ```

2. Make sure the output of *ha_cfgchksum* is the same on both servers.
3. Make sure no errors are reported when you enter the following on the primary and secondary servers:

   ```
   # /usr/etc/ha_cfgverify
   ```

4. Make sure the shared filesystems are not shown on the primary and secondary servers in the file */etc/fstab*, or when you enter *df*.
5. Make sure the mount points are the same on both servers for the shared filesystems.
6. Execute the following command first on the primary server and then on the secondary server:

   ```
   chkconfig failsafe on
   /etc/init.d/failsafe start
   ```

Configuring a Server for Clip Mirroring

You can use the VST *clipmirror* subsystem to implement a fully redundant cluster of two or more VST servers, each with its own private clip storage.

One server is designated the primary server and handles normal clip playback and recording, as well as file transfers to and from other non-VST servers or workstations. The VST *clipmirror* subsystem on each redundant (clip-mirroring) server synchronizes the contents of the redundant server's clip cache with the contents of the primary server's clip cache. That is, the clip caches on the redundant servers are maintained as mirrors of the primary server's clip cache.

This section consists of the following subsections:

Using Clip-Mirroring Servers

You set the name of the primary server that a redundant server is configured to mirror in the file */usr/vtr/config/system-defaults/clipmirror* by specifying the value of the control vtr.clipmirror.primary_server.hostname. For example:

```
vtr.clipmirror.primary_server.hostname newprimary
```

The application can dynamically control the clip-mirroring feature by setting this control with the MVCP *SSET* command. For example:

```
SSET clipmirror vtr.clipmirror.primary_server.hostname newprimary
```

During the initial synchronization phase that occurs when VST starts on a redundant server, the *clipmirror* subsystem on the redundant server opens an MVCP connection to the primary server. Using this connection, the redundant server retrieves the list of clips (including modification times) from the primary server. The redundant server compares the primary server's clips with the clips in the local clip cache, checking for clips that are missing, out of date, or extraneous.

- If the file on a redundant server is newer than the file on the primary server, the clip is assumed to be up to date.

  If the clip file on a redundant server is of the same size as the corresponding file on the primary server and also has a later modification time, then the clip is assumed to be up to date.

- If a clip exists on the primary server only, the clip is copied via FTP to the redundant server.

- If a file exists on the redundant server only, a message is added to the VST log file (default */var/adm/vtr/logs/vtrlog*), but no other action is taken. The clip can be removed from the redundant server manually or by an application.

When a clip is recorded on or transferred via a network to the primary server, it is automatically copied to the clip-mirroring server. The speed of the copy operation depends on the bit rate of the clip, the network media connecting the servers, and whether other clips are being copied or are waiting to be copied to the redundant server.

The maximum number of FTP transfers spawned is set by the control vtr.clipmirror.max_threads. The value of this control should be based on the bandwidth of the network connection; the default is 20. The use of this control and other clip mirror controls is explained in “Setting Up Primary and Redundant Servers” on page 71; clip mirror controls are summarized in “System Controls for Clip Mirroring” on page 116 in Appendix A.

Once the clip caches on the primary and redundant servers are synchronized, the redundant server monitors the primary server for changes to the contents of its clip cache.

- Clips added to the primary server’s clip cache are copied via FTP to the redundant server. A clip is copied to the redundant server again if the clip media data is modified on the primary server.
- If a clip is removed, renamed, or has its protection changed, the redundant server performs the same operation on its copy of the clip.

If the primary server becomes unavailable, each redundant server tries to reconnect to the primary server at an interval determined by the vtr.clipmirror.reconnect_interval control.

In a dual-server environment, if the primary server becomes unavailable for an extended period of time, you can reconfigure the redundant server to be the primary server. When the original primary server is available again, you can reconfigure it as the new redundant server. All new clips on the new primary server are then automatically copied to the new redundant server.

Note: The VST clip-mirroring feature does not work correctly with clip formats that require index files (such as vframe and stream).

Setting Up Primary and Redundant Servers

To set up primary and redundant servers, follow these steps:

1. To make sure that the filesystem holding the clips has the same real-time extent on the primary and on each redundant server, as root enter

   ```
   # xfs_growfs -n /usr/vtr/clips
   ```

   Output should resemble the following, which has been slightly reformatted:

   ```
   meta-data=/usr/vtr/clips/   isize=256    agcount=8, agsize=8192 blks
   data     =                  bsize=16384  blocks=65536, imaxpct=25
            =                  sunit=0      swidth=0 blks, unwritten=1
   naming   =version 1         bsize=16384
   log      =internal          bsize=16384  blocks=1000
   realtime =external          extsz=2097152 blocks=4445586, rtextents=34731
   ```

   In the last line, `extsz` is the extent size of the filesystem.

2. Using the *versions* command, make sure that the following subsystems are installed on the redundant servers:

   - *vst_eoe.sw.clipmirror*
   - *vst_eoe.sw.tools*
   - *vst_eoe.sw.fsmon*
   - *vst_eoe.sw.ftpd*

3. On the primary server, make sure the filesystems *vst_eoe.sw.fsmon* and *vst_eoe.sw.ftpd* are installed.

4. On each redundant server (not the primary server), open the configuration file */usr/vtr/config/system-defaults/clipmirror*. This file contains clip-mirroring controls on separate lines, commented out with pound signs (for example, `#vtr.clipmirror.primary_server.hostname ""`). Delete the pound sign and change the default primary server hostname (the null string) to the hostname of the primary server. For example, for a primary server named vst-1:

   ```
   vtr.clipmirror.primary_server.hostname vst-1
   ```

   If you do not want to set the name of the primary server as the default, the application can set the value of this and all other clip mirror controls with the MVCP *SSET* command; for example:

   ```
   SSET clipmirror vtr.clipmirror.primary_server.hostname vst-2
   ```

5. If desired, include a line in the configuration file */usr/vtr/config/system-defaults/clipmirror* on each redundant server to determine how often (in seconds) the redundant server tries to reconnect to a failed primary server. The minimum interval is 1 second; the default is 30. This example sets the interval to 60 seconds:

   ```
   vtr.clipmirror.reconnect_interval 60
   ```

6. If desired, set the value for the control vtr.clipmirror.max_threads on the redundant server(s); this control determines the maximum number of concurrent clip transfers from the primary server to the redundant server. The default limit is 20; the range is 1 to 100. For example:

   ```
   vtr.clipmirror.max_threads 2
   ```

7. On the primary and redundant servers, create a new user account named *vtrsync*:

   /usr/sysadm/privbin/addUserAccount -l vtrsync -u *idnumber* **-g 0 -H /usr/vtr/clips -S /bin/csh -P**

   The user ID (the variable *idnumber* in the example above) must be unique in the system; it can differ for different servers. To check user IDs already in the system, open */etc/passwd* and view the third field. In the following example, the user ID is 994.

   ```
   tutor::994:997:Tutorial User:/usr/tutor:/bin/csh
   ```

8. For the new user account vtrsync (on each server), set the password to vtrsync:

   ```
   # passwd vtrsync
   New Password:
   Re-enter new password:
   ```

   At each prompt shown above, enter **vtrsync**.

Stopping Automatic Backup

To stop the automatic backup, set the name of the primary server to an empty string via an MVCP connection to the redundant server(s):

```
SSET clipmirror vtr.clipmirror.primary_server.hostname ""
```

Note: Clips are always transferred in full between the primary and the redundant servers, regardless of any in and out points that have been set.

Redesignating Servers

If you are redesignating a redundant server as the primary server, follow these steps:

1. Turn off clip mirroring in the redundant server by unsetting the name of the primary server with the following command in an MVCP connection:

   `SSET clipmirror vtr.clipmirror.primary_server.hostname ""`

2. On the server that is to be the new primary server, open the configuration file */usr/vtr/config/system-defaults/clipmirror* and make sure that the name of the primary is unset:

   `vtr.clipmirror.primary_server.hostname ""`

   Unset the primary if necessary.

3. On each redundant server, set the value of vtr.clipmirror.primary_server.hostname to the name of the new primary server.

An application can use the MVCP *SSET* command to change the values of the clip-mirroring controls while VST is running. For example, if vst-1 is the current primary server and vst-2 is the current redundant server, the following MVCP command swaps their roles:

- On vst-2, use

  `SSET clipmirror vtr.clipmirror.primary_server.hostname ""`

- On vst-1, use

  `SSET clipmirror vtr.clipmirror.primary_server.hostname vst-2`

The new redundant server, vst-1, starts the initial synchronization phase as described in “Using Clip-Mirroring Servers” on page 69.

Setting Transfers to Use Nondefault Network Interfaces

By default, spawned FTPs transfer clips between the primary interfaces of the primary and redundant servers. To enable these transfers to utilize another interface that might be higher bandwidth, use the controls vtr.clipmirror.local_server.hostname and vtr.clipmirror.primary_server.hostname in conjunction. FTP uses the server names specified in these controls.

For example, if the redundant server has a 100-Base-T Ethernet interface, vst1-enet, and a fibre channel connection, vst1-fc, you might wish to transfer clips on the higher bandwidth fibre channel connection. To do so, set vtr.clipmirror.local_server.hostname to vst12-fc. Similarly, if you want to use an interface on the primary server called p-vst-fc, set the control vtr.clipmirror.local_server.hostname to that name.

Configuring and Using External Devices

Installation of VST includes configuring it to work with external devices:

- Devices that control VST:
  - video tape recorder edit controller or other device for controlling VTRs, using a Sony-compatible (P2-compatible) RS-422 control protocol
  - automation controller using the Louth Video Disk Communications Protocol
- Devices that VST controls: most broadcast VTRs using the Sony-compatible RS-422 control protocol
- Devices through which media are played or recorded:
  - SGI DIVO video option board: 8- and 10-bit Rec. 601 digital video playback and recording with optional 2:1 lossless data reduction
  - SGI DIVO-DVC video option board: 8- and 10-bit Rec. 601 digital video playback and recording with optional 2:1 lossless data reduction or DVCPRO25 compression
  - Vela Research four-port SCSI MPEG-2 decoder; this is discussed in “Installing the Vela Research MPEG-2 Decoder” on page 87 in Chapter 7
  - Viewgraphics Dynamo MediaPump
  - SGI Digital Audio Option board for 24-bit uncompressed AES or ADAT digital audio
  - DVB-ASI boards: these are discussed in “Installing and Configuring the DVB-ASI Adapter Board” on page 92 in Chapter 7
- Auxiliary devices: house time-code readers

This chapter explains installing and configuring these devices, in the following sections:

Configuring and Using the Sony (P2) Protocol

The Sony / P2 protocol is partially supported by Video Server Toolkit (VST) so that VST can be controlled by standard VTR controllers. The configuration information makes VST behave similar to a video tape deck.

Note: P2 is an alternate name for the Sony protocol. For the remainder of the chapter, the term "Sony" represents both Sony and P2.

The following sections describe how to configure VST to work with the Sony protocol:

Note: The O2 workstation does not support deck emulation.

Caution: Opening a unit owned by a Sony port using MVCP (for example, UOPN) causes unpredictable behavior and is unsupported.

VST does not currently support the entire Sony protocol specification, including:

- Video output parameter selection
- Audio output parameter selection
- Audio split editing
- Insert editing for anything less than all video / audio tracks
- +/-15% playback / recording / editing

  Note: VAR_FORWARD and VAR_REVERSE are fully implemented.

Configuring Sony Protocol Control Ports

Follow these steps:

1. As root, use the following command to verify that the *vst_eoe.sw.sony* subsystem is installed:

```
# /usr/vtr/bin/vtrswinfo -subsys -short
 Installed software:
   vst_eoe (1265452500)
     vst_eoe.man.base (1265452500)
     vst_eoe.man.ftpd (1265452500)
     vst_eoe.man.relnotes (1265452500)
     vst_eoe.man.tools (1265452500)
     vst_eoe.sw.base (1265452500)
     vst_eoe.sw.divo (1265452500)
     vst_eoe.sw.fsmon (1265452500)
     vst_eoe.sw.ftpd (1265452500)
     vst_eoe.sw.sony (1265452500)
     vst_eoe.sw.tools (1265452500)
```

   If necessary, install the Sony subsystem as explained in “Installing VST” on page 38 in Chapter 3.

2. Cable the controlling device to the VST server. For more information about cabling an RS-232 device to a VST server, see Appendix B, “RS-422 Pinouts.”

3. For each Sony device controlled by VST, enter a line in */usr/vtr/config/control-in.conf*. See Chapter 4, “Configuring Video Server Toolkit” for instructions.

4. Edit the control defaults file for each Sony control port as needed.

   The directory */usr/vtr/config/system_defaults* contains files specifying the default control settings for VST system-wide resources, including the control processors managing any Sony protocol control ports.

   - Put control settings that apply to all Sony control ports in *vtr*.
   - Put control settings that are limited to specific control ports in *vtr_port*, where *port* is the VST serial port number to which the controlling device is attached; for example, *vtr_1*, *vtr_2*, and so on.

The controls supported by the Sony control module are:

- vtr.control.device_type_id
- vtr.control.output.idle_mode
- vtr.control.timecode.mode
- vtr.control.clip.name
- vtr.control.superimpose.enabled
- vtr.control.edit.delay
- vtr.control.ee.delay
- vtr.control.ee.mode
- vtr.control.ee.record_select
- vtr.edit.preroll
- vtr.edit.postroll

See the vst-controls(5) man page for more information concerning the use and default values of these controls.

An application can change these controls dynamically with the MVCP *SSET* command; they can also be queried with the MVCP *SGET* command. For example:

```
SSET vtr_1 vtr.control.clip.name newclip
```

This example creates the clip newclip, if it does not already exist, and loads it onto the units controlled by the device attached to serial port 1.

Note: The setting for vtr.control.output.idle_mode overrides the setting for vtr.media.output.idle_mode of the underlying device.

5. Make sure that the EDIT-ON / EE-ON delay time for your edit controller matches the delay time set by VST controls. For more information, see “Setting VST and Edit Controller Delay Time” on page 79.

Setting VST and Edit Controller Delay Time

The DIVO (and DIVO-DVC) output port processing has a finite delay that constrains how quickly VST can respond to control input. To maintain frame-accurate control, VST and an external VTR edit controller must agree on how much delay occurs between the time an EDIT_ON, EDIT_OFF, FULL_EE_ON, or FULL_EE_OFF command is received and when the command takes effect.

The minimum delay supported by VST depends upon the video board and the type of compression. DIVO and DIVO-DVC support a minimum of 5 frames delay for uncompressed and Rice-coded video. DIVO-DVC supports a minimum delay of 8 frames for DVCPRO video.

Note that because the default delay value for most edit controllers is 5 frames, typically no action is necessary when using uncompressed or Rice-coded video. However, for DVCPRO compression, the VTR edit controller must be configured to expect an 8-frame delay.

As required by your edit controller, you can increase (but not decrease) the command delay by setting the vtr.control.edit.delay or vtr.control.ee.delay controls in the appropriate system defaults file. For more information about these controls, see Appendix A in this document or the vst-controls(5M) man page.

Changing the Clip Loaded in a Sony Controlled Logical Unit

The Sony deck control protocol was originally developed to control VTRs, thus it has no facility for clip management operations including the loading and unloading of clips. The control vtr.control.clip.name can be set to change the clip currently loaded.

The VST GUI application *mcclips* (see the *Video Server Toolkit Developer's Guide* for more information on *mcclips*) enables you to load and unload clips on ports controlled by the Sony protocol. To load a clip, for example, follow these steps:

1. Select a clip.
2. Select File > Load On Port.

You can also create a new clip on a port:

1. Select a clip.
2. Select File > Create On Port.

An application can load a new or existing clip onto the units controlled by a Sony protocol device by using the MVCP *SSET* command to set the value of vtr.control.clip.name (see Appendix A or vst-controls(5M)).

Configuring VST to Control a VTR

VST can frame-accurately control a VTR or VTR-like device that supports the industry-standard Sony compatible VTR RS-422 control protocol. The VTR can be controlled interactively through a user interface or through an application for frame-accurate captures and laydowns of clips. Follow these steps:

1. As root, enter the following to verify that the *vst_eoe.sw.diaquest* subsystem is installed:

```
# /usr/vtr/bin/vtrswinfo -subsys -short
 Installed software:
   vst_eoe (1265452500)
     vst_eoe.man.base (1265452500)
     vst_eoe.man.ftpd (1265452500)
     vst_eoe.man.relnotes (1265452500)
     vst_eoe.man.tools (1265452500)
     vst_eoe.sw.base (1265452500)
     vst_eoe.sw.diaquest (1265452500)
     vst_eoe.sw.divo (1265452500)
     vst_eoe.sw.fsmon (1265452500)
     vst_eoe.sw.ftpd (1265452500)
     vst_eoe.sw.tools (1265452500)
```

   If necessary, install the Diaquest subsystem as explained in “Installing VST” on page 38 in Chapter 3.

2. Cable the VST server to the VTR. For more information about cabling an RS-442 device to a VST server, see Appendix B, “RS-422 Pinouts.”

3. For each Sony device controlled by VST, enter a line in */usr/vtr/config/control-out.conf*. See Chapter 4, “Configuring Video Server Toolkit” for instructions.

   The following example shows the control-out.conf configuration line for a VTR connected to serial port 3:

```
vtr rs422 3 38400 1
```

4. Edit the control defaults file as needed.

   The directory */usr/vtr/config/device_defaults* contains files specifying the default control settings for VST devices, including external VTRs controlled by VST.

   - Put control settings that apply to all controlled VTRs control ports in *dq*.
   - Put control settings that are limited to a specific controlled VTR in *dq_port*, where *port* is the VST serial port number to which the VTR is attached; for example, *dq_1*, *dq_2*, and so on.

   The only VST controls supported by the Diaquest VTR control module are:

   - vtr.edit.preroll
   - vtr.edit.postroll
   - vtr.media.output.mode
   - vtr.edit.coincidence.preroll
   - vtr.media.sync_port

     For information on the controls, see Appendix A, "Video Server Toolkit Control Summary," or the vst-controls(5M) man page.

     To guarantee frame-accurate control, this control must be set to the name of the VST video port that is connected to the VTR. The application *mcpanel* does this automatically, but other applications must ensure that the control is set correctly.

     If the same VST video port is always used with the VTR, you can set the value of vtr.media.sync_port to the name of the video port in the appropriate *dq_port* device defaults file. However, if the controlled VTR might be connected to different VST video ports at different times, set the control via the application using the MVCP *SET* command. For example, before performing a frame-accurate capture from the controlled VTR to the DIVO_DVC_2 video port, set this control:

     ```
     SET dqunitname MED vtr.media.sync_port DIVO_DVC_2
     ```

5. Put the VTR into remote (or slave) mode.
6. Repeat all these steps for as many VTRs as you wish to control and as are connected to the VST server.

Configuring VST for Control by the Louth Video Disk Communications Protocol

The Louth Video Disk Communications Protocol (VDCP), defined by Louth Automation, provides full-featured control of the VST using RS-232, RS-422, and TCP / IP. The VST Louth processor supports:

- back-to-back play and record (subject to restrictions imposed by the video I / O port capabilities)
- archival management
- control of multiple video (signal) ports from a single communications (control) port

Caution: Do not open any units (using MVCT *UOPN*) belonging to the Louth control protocol processor and try to control them.

This section consists of these subsections:

- "Louth Video Disk Communications Protocol" on page 82
- "Using the Louth ADC-100 Automation Controller to VST" on page 83

Louth Video Disk Communications Protocol

Table 6-1 summarizes the VDCP commands that VST supports.

Table 6-1 VDCP Commands Supported by VST

System	Immediate	Reset/Select	Sense Request
Delete From Archive	Stop	Rename ID	Open Port
Delete Protect ID	Play	Reset Std. Time	Next
UnDelete Protect ID	Record	New Copy	Last
	Still	Sort Mode	Port Status Request
	Step	Close Port	Position Request
	Continue	Select Port	Active ID Request
	Jog	Record Init	Device ID Request
	Vari. Play	Play Cue	Device Type Request

Table 6-1 (continued) VDCP Commands Supported by VST

System	Immediate	Reset/Select	Sense Request
		Cue with Data	Syst. Status Request
		Delete ID	ID Liist
		Get from Archive	ID Size Request
		Clear	ID's Added to Arch.
		Send to Archive	ID Request
		% to Signal Full	ID's Added List
		Record Init with Data	ID's Deleted List
		Disk Preroll	Multi Port Status Request

Note: VST supports Deferred (Timeline) commands.

Using the Louth ADC-100 Automation Controller to VST

An ADC-100 Louth automation controller is connected to a VST server using one or two serial ports per the video port to be controlled. If you want to play and record at the same time, two serial port connections are required, unless you are using VDCP multiple-port command support.

O2 workstation serial ports run only in RS-232 mode. Consequently, to connect the Louth ADC-100 automation controller to the serial ports of the O2 workstation, do one of the following:

- Insert a 422 / 232 protocol converter between the Louth and the O2 workstation.
- Use a special cable that uses the A (negative) and ground line from the RS-422 to provide a RS-232-compatible connection.

Follow these steps to connect and configure an ADC-100 Louth automation controller:

1. Cable the automation controller to the VST server. For more information about cabling an RS-442 device to a VST server, see Appendix B, “RS-422 Pinouts.”
2. Configure the automation controller. The information in Table 6-2 specifies how to configure a Louth ADC-100 to control VST. If you are using another automation controller that uses Louth VDCP to control a video server, you might need to configure that controller in a similar way.

   Configure each Louth communications port that controls one or more VST video ports to use the Standard Video Disk device protocol.

Table 6-2 Louth Device Parameters

Device Parameter	Setting
VIDEO INPUT PORT IN DISK	Video port number; corresponds to signal configuration line in *control-in.conf*
VIDEO OUTPUT PORT IN DISK	Video port number; corresponds to signal configuration line in *control-in.conf*
UPDATE EVENT DURATIONS FROM DISK	Enable only on one port for each VST server
CONFIGURE INSTANT PLAY PREROLLS	Enabled, 0 seconds 4 frames
CONFIGURE DISK PREROLLS	4 frames (Rice, uncompressed)
NUMBER (=) OF FRAMES TO SEND PLAY EARLY	3 frames (DVCPRO)
ENABLE BACK TO BACK PLAY	3 frames (DVCPRO)
ENABLE BACK TO BACK RECORD	3 frames (DVCPRO)
DISK HAS ARCHIVE	Enabled if StudioCentral archive is available; otherwise disabled
BACKUP PLAY FROM ARCHIVE SUPPORTED	Disabled
CACHE RECORD DISK SERIAL COMM. PORT NUMBER	0

3. For each Louth communications port connected to a VST serial port, include a control port configuration line in */usr/vtr/config/control-in.conf*. See “Control Port Configuration Line” on page 51 in Chapter 4 for instructions, for example, for a RS-422 connection.

4. For each video port to be controlled through VDCP, include a signal port configuration line like that described in “Control Port Configuration Line” on page 51 in Chapter 4

The following example shows the VST control-in.conf file for a configuration where VDCP is used to control 4 output video ports and 2 input video ports. The Louth communications ports controlling the outputs are connected to serial ports 3 through 6 and control DIVO_DVC_0 through DIVO_DVC_3, respectively. The Louth communications ports controlling the inputs are connected to serial ports 7 and 8 and control DIVO_DVC_0 and DIVO_DVC_1, respectively.

```
louth rs422 3 38400 1 1 29.97 8
louth rs422 4 38400 1 2 29.97 8
louth rs422 5 38400 1 3 29.97 8
louth rs422 6 38400 1 4 29.97 8
louth rs422 7 38400 1 -1 29.97 8
louth rs422 8 38400 1 -2 29.97 8


signal 1 DIVO_DVC_0
signal 2 DIVO_DVC_1
signal 3 DIVO_DVC_2
signal 4 DIVO_DVC_3
```

Using House Time-Code Readers With VST

VST supports the Miranda Little Red and Horita PR-232 time-code readers (LTC-to-serial translators), which are RS-232 (not RS-422) serial devices. One of these devices can be attached to a VST server to provide a frame-accurate time-of-day reference so that VST operations can be synchronized with other studio equipment or with scheduled live or downlinked feeds.

The time-of-day signal connected to a VST server is used as the reference for triggering timed unit commands for playback, recording, and so on. For media devices (video ports) that support it, frame-accuracy is guaranteed when time-triggered commands are executed.

The time-of-day signal is also used to slave the time-of-day maintained by the IRIX operating system. The time-of-day is typically maintained within 1 ms of the input time signal.

To install a time code reader and configure the VST server for it, follow these steps:

1. Connect the time-code reader's serial port to the desired RS-232 port on the VST server.
2. Include a control port configuration line in */usr/vtr/config/control-in.conf*. See "Signal Configuration Line" on page 54 in Chapter 4 for more information. Note the following:
   - For time-code input, the signal port field ("signalport" on page 53) of the configuration line specifies the time channel number, which must be 1 for this version of VST.
   - Both the Miranda Little Red and Horita PR-232 connect at a serial port speed of 9600 bits / sec with no parity.

   The following example shows the VST *control-in.conf* file for a Horita PR-232 connected to serial port 3:

   ```
   hsip rs232 3 9600 0 1 29.97
   ```

   This example shows a Little Red in a 625 / 50 configuration:

   ```
   little-red rs232 3 9600 0 1 25
   ```
3. Add a line for each device to *control-in.conf* to configure VST to work with Horita and the Miranda Little Red time-code readers; for example:

   ```
   hsip rs232 2 9600 0 1 29.97
   little-red rs232 2 9600 0 1 29.97
   ```

   The format of the configuration line is explained in "Configuring VST for Control by Remote Devices (control-in.conf File)" on page 50 in Chapter 4.

Configuring and Using Vela and DVB-ASI Devices

This chapter explains how to install DVB-ASI option boards and the Vela Research four-port MPEG-2 decoder and how to configure the VST server to work with them, in these sections:

Installing the Vela Research MPEG-2 Decoder

For MPEG-2 decoding, VST supports the Vela Research four-port SCSI-attached decoders. These decoders communicate with the VST server over a single-ended or differential fast / wide SCSI bus and decode MPEG-2 transport or program streams to composite analog video.

The Vela decoder supports two kinds of SCSI ports: differential and single-ended. Make sure to use the Vela board that supports your SCSI ports:

- The built-in external SCSI port on the O2 is single-ended.
- The Origin 200 system does not have a built-in SCSI port.
- The SGI MSCSI XIO card has four SCSI ports: port 4 is auto-sensing for single-ended or differential.
- PCI expansion cards are available for both single-ended and differential SCSI.

VST supports up to three four-port decoders (12 MPEG-2 streams) on each SCSI bus. No other SCSI devices should be attached to the bus that is attached to the Vela decoders.

This section describes how to install and configure the Vela decoder, in the following subsections:

Installing and Connecting the Vela Decoder

Follow these steps to install and connect the equipment:

1. Install the Vela decoder, following instructions in its manual.
2. Attach the Vela decoder to a SCSI port on the VST system with a standard SCSI-3 cable.

   Because the decoder does not provide a SCSI loop-through connector, if you are connecting more than one decoder to a single bus, you must use a bus-type cable, usually available as a ribbon cable with four to six SCSI-3 connectors attached.

   Note: Do not use a SCSI-3 cable designed for use inside a computer chassis, because this type of cable is not shielded.
3. Set the decoder's SCSI ID, following instructions in the Vela documentation.

Configuring the Vela Decoder

To configure the Vela decoder, follow these steps:

1. As root, enter the following command to ensure that the *vst_eoe_sw.vela* and *vst_eoe_sw.mpeg* subsystems are installed.

   ```
   # /usr/vtr/bin/vtrswinfo -subsys -short
   ```

   The system responds:

   ```
   Installed software:
     vst_eoe (1275521220)
       vst_eoe.man.base (1275521220)
       vst_eoe.man.ftpd (1275521220)
       vst_eoe.man.relnotes (1275521220)
       vst_eoe.man.tools (1275521220)
       vst_eoe.sw.base (1275521220)
       vst_eoe.sw.fsmon (1275521220)
       vst_eoe.sw.ftpd (1275521220)
       vst_eoe.sw.mpeg (1275521220)
       vst_eoe.sw.tools (1275521220)
       vst_eoe.sw.vela (1275521220)
   ```

   If necessary, install the Vela and MPEG subsystems as explained in “Installing VST” on page 38 in Chapter 3.

2. Verify that the Vela decoders are recognized and contain the correct firmware revision. VST version 1.2 requires firmware revision 4035. Enter

   ```
   # /usr/vtr/bin/vtrhwinfo
   ```

   The system responds:

   ```
   System:
     Platform: Origin 200 (hostname=vcpqa4 serial=)
     Processors: 2x225MHz R10000 (IP27)
     Secondary Cache: 2 Mbytes
     Memory: 512 Mbytes
   Video devices:
     vela_0 (bus=2 unit=7 lun=0 dev=sc2d7l0 firmware=<SCSI Decoder X4 4035>)
     vela_1 (bus=2 unit=7 lun=1 dev=sc2d7l1 firmware=<SCSI Decoder X4 4035>)
     vela_2 (bus=2 unit=7 lun=2 dev=sc2d7l2 firmware=<SCSI Decoder X4 4035>)
     vela_3 (bus=2 unit=7 lun=3 dev=sc2d7l3 firmware=<SCSI Decoder X4 4035>)
     vela_4 (bus=3 unit=9 lun=0 dev=sc3d9l0 firmware=<SCSI Decoder X4 4035>)
     vela_5 (bus=3 unit=9 lun=1 dev=sc3d9l1 firmware=<SCSI Decoder X4 4035>)
     vela_6 (bus=3 unit=9 lun=2 dev=sc3d9l2 firmware=<SCSI Decoder X4 4035>)
     vela_7 (bus=3 unit=9 lun=3 dev=sc3d9l3 firmware=<SCSI Decoder X4 4035>)
   ```

```
Audio devices:
  - none -
Audio input ports:
Audio output ports:
Serial ports:
  tty1 (module=1/O200 built-in)
  tty2 (module=1/O200 built-in)
```

If the incorrect firmware revision is loaded, refer to the Vela decoder documentation for instructions to load the firmware.

Changing Vela Decoder Port Names

When VST is started, each of the four ports on each decoder is assigned a port name. The names are sequential; for example, if one decoder is attached, the ports are named vela_0, vela_1, vela_2, and vela_3. If a Vela decoder is connected to the next SCSI controller, its ports are vela_4, vela_5, vela_6, and vela_7, and so on.

Adding or removing a SCSI device might change the logical ordering of the decoder ports. You can create a fixed mapping of logical port numbers to physical devices through */usr/vtr/config/vtr_ioconfig.conf*. This file functions like the */etc/ioconfig.conf* file that IRIX uses to map logical controller numbers to physical hardware devices.

If *vtr_ioconfig.conf* does not already exist on the VST server, create it with the following:

```
% /usr/vtr/bin/vtrvclahinv > /usr/vtr/config/vtr_ioconfig.conf
```

The file contains one entry for each recognized Vela decoder port (four ports per decoder). For example, one decoder attached to each of two SCSI buses might create the following configuration:

```
0 /hw/module/1/slot/MotherBoard/node/xtalk/8/pci/5/scsi_ctlr/0/target/7/lun/0/scsi
1 /hw/module/1/slot/MotherBoard/node/xtalk/8/pci/5/scsi_ctlr/0/target/7/lun/1/scsi
2 /hw/module/1/slot/MotherBoard/node/xtalk/8/pci/5/scsi_ctlr/0/target/7/lun/2/scsi
3 /hw/module/1/slot/MotherBoard/node/xtalk/8/pci/5/scsi_ctlr/0/target/7/lun/3/scsi
4 /hw/module/1/slot/MotherBoard/node/xtalk/8/pci/7/scsi_ctlr/0/target/9/lun/0/scsi
5 /hw/module/1/slot/MotherBoard/node/xtalk/8/pci/7/scsi_ctlr/0/target/9/lun/1/scsi
6 /hw/module/1/slot/MotherBoard/node/xtalk/8/pci/7/scsi_ctlr/0/target/9/lun/2/scsi
7 /hw/module/1/slot/MotherBoard/node/xtalk/8/pci/7/scsi_ctlr/0/target/9/lun/3/scsi
```

Each entry includes the logical port number followed by the IRIX hardware graph name for the SCSI LUN associated with each decoder port.

Removing a decoder does not change the logical port numbers associated with the remaining decoder. If decoders are added to the system, their ports are assigned logical port numbers above the highest existing logical port number.

Using the External Sync Reference

You can configure the Vela decoder to sync the output video to an external reference by setting vtr.media.video.output.sync_source to external. The default value, internal, causes the decoder to generate its own video sync.

Set the sync source control in a device defaults file in the */usr/vtr/config/device-defaults* directory. Add the control setting to the file *vela* to set the sync source or all Vela ports or to the file *vela_port* to set the control for a specific port, such as *vela_1*, *vela_2*, and so on.

You can also set the sync source control dynamically with the MVCP *SET* command.

Vela Controls and Incompatibilities

Table 7-1 summarizes Vela decoder controls and their default values.

Table 7-1 Vela Controls

Control	Default
vtr.media.vela.error.recover_realtime	TRUE
vtr.media.vela.error.retry	TRUE
vtr.media.vela.output.write_size	65536
vtr.media.vela.scsi.timeout	0
vtr.media.vela.command_latency	0
vtr.media.vela.query_decoder_location	TRUE
vtr.media.vela.seamless_playback	FALSE

For more information about any of these controls, see Appendix A or the vst-controls(5M) man page.

The Vela decoder does not support the following MVCP commands:

- *GOTO*
- *JOG* (in reverse)
- *SHTL* (in reverse)

  SHTL does not provide continuously variable speeds.

Installing and Configuring the DVB-ASI Adapter Board

The Viewgraphics MediaPump is a PCI-based adapter board. It multiplexes MPEG-2 transport streams and transmits them over coaxial cables using the Digital Video Broadcast Asynchronous Serial Interface (DVB-ASI) protocol.

This chapter describes how to install and configure a DVB-ASI adapter board for use with VST. It consists of these sections:

Installing the Adapter Board

The DVB-ASI adapter board requires a full-length PCI slot. These slots are supported on the Origin 200 GIGAchannel system, but not in the O2 chassis. Follow these steps:

1. Install the adapter board(s) in a PCI slot, following the instructions for installing a PCI board in the *Origin200 and Origin200 GIGAchannel Maintenance Guide*.

2. As root, enter the following command to verify that the *vst_eoe.sw.dvb-vg* and *vst_eoe.sw.mpeg* subsystems are installed:

   ```
   # /usr/vtr/bin/vtrswinfo -subsys -short
   ```

   The system responds:

   ```
   Installed software:
     vst_eoe (1275521220)
       vst_eoe.man.base (1275521220)
       vst_eoe.man.ftpd (1275521220)
       vst_eoe.man.relnotes (1275521220)
       vst_eoe.man.tools (1275521220)
       vst_eoe.sw.base (1275521220)
       vst_eoe.sw.fsmon (1275521220)
       vst_eoe.sw.ftpd (1275521220)
       vst_eoe.sw.mpeg (1275521220)
       vst_eoe.sw.tools (1275521220)
       vst_eoe.sw.vela (1275521220)
   ```

   If necessary, install the Viewgraphics and MPEG subsystems. The Viewgraphics driver software, *mediapump.sw.base*, is included with the VST software images and is a prerequisite for installing the *vst.sw.dvb-vg* subsystem. For more information about installing this software, follow the installation instructions for VST presented in Chapter 3, “Installing Video Server Toolkit.”

   Reboot after installing the subsystems.

3. To verify that the system recognizes the DVB-ASI board(s), open */etc/ioconfig.conf*. This file is automatically generated and maintained; it shows how PCI card slots correspond to DVB-ASI boards in the Origin 200 chassis; for example:

   ```
   0 /hw/module/1/slot/io7/xbox_dualxtown/pci/3/Dynamo
   1 /hw/module/1/slot/io1/MotherBoard/pci/5/Dynamo
   ```

   The first line says that DVB-ASI 0 is in PCI slot 3 of the Origin 200 GIGAchannel chassis. The second line says that DVB-ASI 1is in PCI slot 5 of module 1 motherhoard.

   The DVB-ASI boards also appear in the hardware inventory (*hinv*) as Unknown:

   ```
   hinv: Unknown type 117 class 12
   hinv: Unknown type 117 class 12
   ```

   The software installation for the DVB-ASI board appears as an option in the Inst package that comes with VST. To use the DVB-ASI adapter card, you must install *vst_eoe.sw.dvb-asi*.

Maximizing Output Streams

To enable the MPEG2 stream splicing capability of the Viewgraphics MediaPump board, set the control vtr.media.dvb-asi.seamless_playback to true in the device defaults file, */usr/vtr/config/device-defaults/dvb-asi*. You can also set this control dynamically with the MVCP *SET* command.

To maximize the number of output streams, set the following control:

```
vtr.media.dvb-asi.seamless_playback false
```

The default value is false.

You must change the program ID before cueing a clip. The new program ID remains in effect for that unit even after the clip has finished.

Note: The DVB-ASI board might work without any change to its default configuration. If you have resource conflicts, however, see the *Video Server Toolkit Developer's Guide* for information on setting specific device controls to enable DVB-ASI playback.

DVB-ASI Time-Delay Server

VST includes an HDTV time-delay server feature that allows you to pause or stop a compressed MPEG-2 transport stream. This feature works in conjunction with the Viewgraphics MediaPump DVB-ASI PCI card.

VST can record a DVB-ASI stream for a predefined length of time. You provide a time delay (expansion) in the output stream with an MVCP STOP command, or by a STOP-CUE-PLAY sequence. For example:

CUE *unit in-point*
@time PLAY *unit*

VST finds the closest possible position before the in point you specify, and starts playing. If the position VST finds is five frames before the in point, play starts five frames before the specified time. Thus the correct frame for the specified time plays.

When the output stream is stopped, it can be cued accurately to continue the playout. Content can be inserted locally, downstream of the decoder.

Supported MVCP Commands

The DVB-ASI adapter board supports only the following MVCP commands:

- *CUE*
- *CUER*
- *LOAD*
- *PLAY*
- *REC*
- *STOP*
- *UNLD*

Running VST

This chapter consists of the following:

Starting and Stopping VST

Video Server Toolkit (VST) is normally started automatically when the VST server boots. In a development environment, you may choose to disable automatic startup and start it manually instead.

This section explains how to start VST in the following sections:

Note: For information on booting from a backup plex, see “Configuring for Booting From a Backup Plex” on page 113 in Chapter 9.

Configuring VST to Start Automatically

By default, when VST is installed, it does not enable itself to be automatically started when the system boots. As superuser, you can configure VST to start automatically at boot time using *chkconfig* with the **vtr** option. The following enables automatic startup:

```
# chkconfig vtr on
```

The following disables automatic startup:

```
# chkconfig vtr off
```

Starting VST Manually

You must be root to start VST. Start VST with

```
# /usr/vtr/bin/vtrstart
vtrstart: Starting video server
```

Note: The message is displayed only if the system has verbose logging enabled (*chkconfig verbose on*).

If VST is already running, *vtrstart* displays an error:

```
vtrstart: Video server is already running (use -f to force restart)
```

As indicated, to stop the VST instance currently running and start a new VST instance, add the -f option:

```
# /usr/vtr/bin/vtrstart -f
vtrstart: Stopping video server
vtrstart: Starting video server
```

Stopping VST

Stop VST with

```
# /usr/vtr/bin/vtrstop
vtrstart: Stopping video server
```

Setting Startup Options

When the VST daemon *vtrd* starts, it reads the configuration file */usr/vtr/config/vtrd.conf*. The daemon *vtrd* is the VST server daemon that manages the VST server processes.

The default installation operates normally without any change to the startup options. However, you can change the startup options for *vvtr*, the VST server process, by editing */usr/vtr/config/vtrd.conf* and adding them immediately after the program path, */usr/vtr/bin/vvtr*.

Table 8-1 describes the startup options for *vvtr*.

Table 8-1 VST Server Process (*vvtr*) Startup Options

Option	Description
-l *logopts*	Sets the log options, which control the format of the messages written to the log. If this option is omitted, each message contains the severity code, a timestamp, the process ID, and the text of the message. The following may be specified for this option: -l l, do not include the severity code in each log message -l t, do not include the time stamp in each log message -l p, do not include the process ID in each log message See "VST Log Message Structure" on page 101 for more information about log message format.
-p	Specifies that VST should not run at high priority (p). If this option is omitted, VST runs at high real-time system priority to ensure no interruptions in video/audio input or output. Include this option only at the direction of VST support personnel.

Checking VST Status

You can use *vtrstat* to check whether VST is running on the system. If VST is not running, *vtrstat* displays the following message:

```
# vtrstat
Video server on <host> is stopped
```

If VST is running and responding to MVCP connections, *vtrstat* displays

```
Video server on <host> is running.
```

For additional VST status information displayed by *vtrstat*, see “Monitoring the System” on page 104.

Using the O2 Graphics Window

An optional graphics port is available on O2 workstation that allows you to see the outgoing video in the VST X window in the graphics display. This window is sized to display a 625 / 50 video frame with non-square pixels (720 x 576). If you are playing video with a smaller image size, the image is centered within the VST X window.

Note: If you are playing video with non-square pixels (CCIR601 timing), for example a 525 / 59.94 video frame, the aspect ratio in the graphics window is incorrect; pixels are square on the graphics display and VST does not rescale the image.

The graphics window is enabled by default on O2 workstation. To disable it, edit */usr/vtr/config/device-defaults/mvp* and set *vtr.media.graphics.enabled* to false.

If you wish to use the graphics port, you cannot start VST automatically at boot time. The *vvtr* X Window appears only when you use */usr/vtr/bin/vtrstart* to start VST.

To disable automatic startup, enter this command;

```
# chkconfig vtr off
```

Logging VST Events

The VST logging feature provides a mechanism for storing a text record of events that occur during operation of the system. The default logging configuration generally records only events that are generated by various error conditions, but you can enable additional logging to help track and trouble-shoot VST behavior.

This section explains

VST Log Message Structure

Each VST log message has the following format:

c dd-hh:mm:ss.mmmmmm pppp <log message>

where

- *c* is the severity code of the message
- *dd* is the day of the month
- *hh:mm:ss.mmmmmm* is a time stamp that indicates when the message was written
- *pppp* is the process ID of the process that wrote the message to the log
- *log message* is the actual text message

Table 8-2 shows the VST log severity levels and codes, which are listed in decreasing order of severity.

Table 8-2 Log Levels

Severity Level	Severity Code	Description
Emergency	P	Panic condition.
Alert	A	A condition that should be corrected immediately, such as a corrupted system file.
Critical	C	A critical condition that has system-wide impact, such as a hard device error; immediate action is required.
Error	E	A problem that needs correcting, but does not require immediate action.
Warning	W	Possible problem, but could be a transient problem that corrects itself.
Notice	N	Condition that might require attention, but isn't an error condition.
Info	I	Informational message.
Debug	*n*	Information message that normally is of use only to engineers for debugging; can be Debug1, Debug2, or Debug3, with Debug3 producing the most amount of debugging information.

The following is an example of a message that has a severity code of 2 (Debug2 severity level). The message was written on day 14 of the month at the time that is shown in the message, and the ID of the process that wrote the message is 8254:

```
2 14-22:23:50.316766  8254 mvcp/ninety9 <-- 100 VTR Ready
```

The following example shows a Notice-level message.

```
N 29-09:26:11.490919   3064 U1 Unit ERROR (err=22): Clip timecode type
not know
```

The following example shows an Info-level message.

```
I 29-09:26:12.758644   3065 (littlered_1) Timecode input acquired
tc=09:27:35.02 (offset=* -27000000)
```

The following example shows a Warning-level message.

```
W 28-14:06:56.807983  35339 U16(DIVO_DVC_2) Video input missing
```

Configuring Logging

Logging is configured by setting the value of the following controls in the system defaults file */usr/vtr/config/system-defaults/main*:

- vtr.main.log_level.file
- vtr.main.log_level.syslog

For example, to enable increased logging by including the first level of debug events in the events that are logged to the server log, */usr/vtr/adm/logs/vtrlog*, include the following control setting in the main system defaults file:

```
vtr.main.log_level.file 1
```

The value of the control is relative to the Info log event severity. Positive values enable more verbose logging (1 = Debug1, 2 = Debug2, ...); negative values disable all but the more severe events (-2 = Warning, -3 = Error, ...).

Managing Log Rollover

Log rollover is the practice of saving the current log file and restarting logging into an empty log file. Rollover avoids the problems of running out of disk space and of having a list of log messages too long to handle easily.

VST manages the vtrlog server log file. By default, the log file is saved and a new log file begun every night at 2:00 a.m. if the log file is at least 10 MB. The default number of log files retained on the system is ten.

The command *rotatelogs* is used to manage the log files and is executed by the system's cron job handler at the right time. Table 8-3 shows the list of available options.

Table 8-3 rotatelogs Command Options

Option	Specifies	Default Value
-b	Log filename	vtrlog
-d	Log file directory	/var/adm/vtr/logs
-h	Lists these options	
-l	Daemon to notify	vtrd
-m	Maximum number of backups retained on the server	10
-s	Minimum log file size to trigger rollover	10 MB
-D	Debug level	Off

To change any defaults (such as the time of the rollover or the log size), the crontab entry of rotatelogs in the system must be edited (as root, using *crontab -e*). This entry was installed as part of the initial VST installation by an exitop.

For example, to retain the last 20 log files on a machine, the rotatelogs entry in crontab must look like:

```
1       2       *       *       *       /usr/vtr/bin/rotatelogs -m 20
```

The debug option is off by default. It can be turned on using the **-D** option. There is only one level of debugging information (on).

Monitoring the System

Once Video Server Toolkit (VST) is installed and running, you can monitor its operation using the tools mentioned in this chapter. Two tools, *vtrstat* and *mcstat*, work directly with VST. You can also use standard SGI IRIX tools to monitor various system resources.

This chapter describes how to monitor VST in the following sections:

vtrstat

vtrstat is a command-line tool that tells you:

- Whether or not VST is running
- Which units are open, if the **-units** option is used
- Which media ports are available, if the **-ports** option is used

Example 8-1 shows an example output of *vtrstat*.

Example 8-1 vtrstat Output

```
vsta 7# /usr/vtr/bin/vtrstat -units
Video server on vsta is running.

Unit  Owner        Port    Clip       Function Location
-------------------------------------------------------------
U1    louth        DIVO_0  d/REC12    STOP     01:04:30.05
U2    louth        DIVO_0  *          STOP     *
U3    louth        DIVO_1  d/rPOP     CUE      00:00:00.00
U4    louth        DIVO_1  d/rSEL     STOP     00:00:00.00
U5    louth        DIVO_1  d/REC01    STOP     01:04:30.06
U6    louth        DIVO_1  *          STOP     *
U7    louth        DIVO_0  d/rSEL     CUE      00:00:00.00
U8    louth        DIVO_0  d/rSTOK    STOP     00:03:30.01
U9    louth        DIVO_7  d/rJOR     CUE      00:00:00.00
U10   louth        DIVO_7  d/rSEL     STOP     00:00:00.00
U11   louth        DIVO_3  d/rPOP     CUE      00:00:00.00
U12   louth        DIVO_3  d/rSTOK    PLAY     00:03:02.29
U13   louth        DIVO_4  d/rSEL     CUE      00:00:00.00
U14   louth        DIVO_4  d/rSTOK    STOP     00:00:00.00
U15   louth        DIVO_5  d/rSEL     CUE      00:00:00.00
U16   louth        DIVO_5  d/rSTOK    STOP     00:03:30.01

vsta 8# /usr/vtr/bin/vtrstat -ports
Video server on vsta is running.

#  Port    Type    Description
-----------------------------------------------------------
1  DIVO_0  Video   SGI XT-DIVO Digital Video Option
2  DIVO_1  Video   SGI XT-DIVO Digital Video Option
3  DIVO_3  Video   SGI XT-DIVO Digital Video Option
4  DIVO_4  Video   SGI XT-DIVO Digital Video Option
5  DIVO_5  Video   SGI XT-DIVO Digital Video Option
6  DIVO_6  Video   SGI XT-DIVO Digital Video Option
7  DIVO_7  Video   SGI XT-DIVO Digital Video Option
8  dq_0    Deck    Diaquest Deck Control
```

A description of these functions can be found in the *Video Server Toolkit Developer's Guide* in Chapter 11, "Completing Common Tasks Using MVCP Commands."

mcstat

mcstat graphically displays the activity of all units that are currently open on a VST server. It is recommended that you install *vst_eoe.sw.base* and *vst_eoe.sw.tools* on the SGI workstation you are using to monitor the VST server and run *mcstat* on that workstation instead of the server.

Figure 8-1 shows an example of *mcstat*'s output when VST is running. When VST is not running, *mcstat* exits.

Figure 8-1 mcstat Output

mcstat displays a list of all the media ports supported by the VST server. For each port, it displays all the units that are opened on that port; each unit is on a separate line.

The unit display includes the name of the unit, the unit owner information, the name of the clip currently loaded on the unit, the last function executed by the unit (or the one currently executing), the current clip time code, and the status of the function.

IRIX Tools

SGI provides a collection of monitoring tools that can be used with VST, including

- *sar*: system activity reporter; reports operating system activity
- *gr_osview*: graphical system monitor; graphically displays real-time usage of certain system resources
- *osview*, a text version of *gr_osview*
- Performance Co-Pilot (PCP): serves as an interface for existing reporting tools, such as graphing performance data over time

Before exporting media data recorded by VST to IRIX tools, you must manipulate the data. For more information about exporting media data, see *Video Server Toolkit Developer's Guide*.

Using sar

sar, the System Activity Reporter, is an activity counter. The command line options allow you to specify the kinds of activities you want measured, such as disk utilization. To use *sar*, follow these steps:

1. Enable *sar* using the following command:

   ```
   # chkconfig sar on
   ```

2. Reboot your system.

   When your system reboots, the *sar* data collector starts.

3. Specify the system activity you want *sar* to measure; for example, to measure disk utilization, use a command similar to the following:

   ```
   # sar -d 2 10
   ```

   The **-d** command-line option specifies the display of disk utilization. The numbers, 2 and 10, specify how often you want to take reports. In this example, 10 reports are taken every 2 seconds.

Note: For a complete list of *sar* command line options, see the sar(1M) man page.

The following is example output of *sar*.

```
vsta 100# sar -d 2 10

IRIX64 vsta 6.4 02121744 IP27    12/19/1997

11:15:10     device %busy  avque  r+w/s  blks/s    w/s wblks/s  avwait
avserv
11:15:12
dks2d1     8    1.0      3   12002      0       0     0.0    28.3
dks3d69    8    1.0      1   12257      0       0     0.0    53.3
dks0d1     8    5.4      7     120      7     120    49.3    11.3
dks1d1     0    0.0      0       0      0       0     0.0     0.0
dks0d2     0    0.0      0       0      0       0     0.0     0.0
dks1d2     0    0.0      0       0      0       0     0.0     0.0
dks0d3     0    0.0      0       0      0       0     0.0     0.0
dks1d3     0    0.0      0       0      0       0     0.0     0.0
dks0d4     0    0.0      0       0      0       0     0.0     0.0
dks1d4     0    0.0      0       0      0       0     0.0     0.0
dks0d5     0    0.0      0       0      0       0     0.0     0.0
dks1d5     0    0.0      0       0      0       0     0.0     0.0
dks0d6     0    0.0      0       0      0       0     0.0     0.0
dks1d6     0    0.0      0      16      0      16     0.0     0.0
```

Using gr_osview

gr_osview provides a graphical display of system resources use. This display provides a real-time window into the overall operation of the system, as shown in Figure 8-2.

Figure 8-2 gr_osview

For more information about this utility, see its man page, gr_osview(1M)

Using osview

osview is a text version of *gr_osview*. The following show example output of *osview:*

```
Osview 2.1 : One Second Average   vcpqa2      10/02/98 09:30:25 #14
int=5s
 Load Average            4MB pages    0    vidintr        0
   1 Min      0.008      16MB pages   0    drop_add       0
   5 Min      0.000  System Activity     TCP
   15 Min     0.000    syscall      206    conns          0
 CPU 0 Usage            read           3    sndtotal       0
   %user       0.00    write          0    rcvtotal       0
   %sys        1.20    fork           0    sndbyte       38
   %intr       0.00    exec           0    rcvbyte        0
   %sxbrk      0.00    readch        45  UDP
   %idle      98.80    writech       78    ipackets       1
 CPU 1 Usage            iget           0    opackets       0
   %user       0.00  Block Devices         dropped        0
   %sys        0.60    lread          0    errors         0
   %intr       0.00    bread          0  IP
   %sxbrk      0.00    %rcache      0.0    ipackets       1
   %idle      99.40    lwrite         0    opackets       0
 Wait Ratio             bwrite     22.4K    forward        0
   %IO          0.0    wcancel        0    dropped        0
   %Swap        0.0    %wcache      0.0    errors         0
   %Physio      0.0    phread         0  NetIF[ef0]
 System Memory          phwrite        0    Ipackets       2
   Phys      512.0M *Swap                   Opackets       0
    kernel    26.7M *System VM              Ierrors        0
     heap      7.7M  Memory Faults         Oerrors        0
      stream  48.0K    vfault         0    collisions     0
     zones     7.7M    protection     0 *NetIF[lo0]
     ptbl      2.9M    demand         0  Scheduler
    fs ctl     9.8M    cw             0    runq           0
    fs data  105.9M    steal          0    swapq          0
     delwri       0    onswap         0    switch        65
    free     350.5M    oncache        0 *Large page stats
    userdata  19.2M    onfile         0 *Interrupts
    pgallocs      0    freed          0 *PathName Cache
 Node[0]                unmodswap      0 *Heap
     Totalmem505.1M    unmodfile      0 *EfsAct
     freemem 350.5M    iclean         0 *XfsAct
     64k page  1.6K *TLB Actions        *Getblk
     256k pages   0  Video              *Vnodes
exit 1MB pages    0    vidioctl       0
```

Using Performance Co-Pilot (PCP)

Performance Co-Pilot (PCP) provides a suite of tools that co-operate to deliver distributed, integrated performance-monitoring and performance-management services across a spectrum of performance domains, as shown in Figure 8-3.

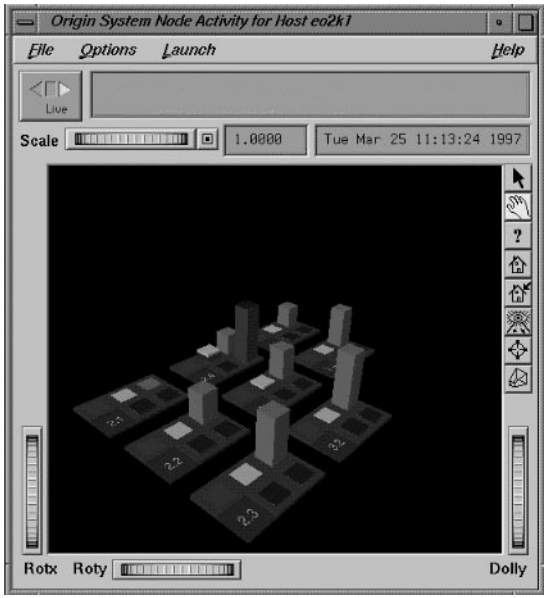

Figure 8-3 Performance Co-Pilot

For more information about Performance Co-Pilot, see its man page, pcp(1M).

Troubleshooting

This chapter lists an assortment of common problems and their solutions.

The problems solved include:

625/50 Clips Do Not Play

The default values for the controls are NTSC-specific. If you are using a 625/50 system, set the following system control in */usr/vtr/config/system-defaults/main*:

```
vtr.main.timing_standard 625
```

This control sets the timing of the system, including the DIVO cards, to 625/50.

In */usr/vtr/config/device-defaults/DIVO*, make sure the control vtr.media.video.output.timing is set to its default value, *system*.

Adding a Clip Takes Excessively Long

Adding clips always has a lower priority than playout and recording.

DIVO Is Left in Bad State

If you find that the DIVO board has been left in a bad state, for example, you cannot reset it, use the *divo_reset* command, included in the *divo.sw.diag* images, to correct the problem. For example, the DIVO reset command is in the directory, */usr/dmedia/DIVO/bin*.

If you are using a 625/50 system, see "625/50 Clips Do Not Play" on page 111.

Crash: Semaphore Limit Exceeded

If *vvtr* crashes, you might have exceeded the number of configured pollable semaphores. The log would appear as follows:

```
C 06-14:29:23.348170   1386 Failed opening semaphore file descriptor
(sems open=147): No space left on device
A 06-14:29:23.348985   1386 FATAL SYSTEM ERROR:
SYSTEM CONFIGURATION ERROR
```

To fix the problem, follow these steps:

1. Edit */var/sysgen/master.d/usema* and increase *USMAXDEVS*, for example, to 1050.
2. Rebuild the kernel (*autoconfig*) and reboot the system to activate the change.

Crash: Audio Sync Problems When Audio Is Missing

VST crashes after a few seconds if you do not have an audio source. If you are having audio sync problems when there is no audio, do the following:

1. Turn the audio off: in the */usr/vtr/config/device-defaults* directory, edit the DIVO file and add:

   `vtr.media.audio.input.channels.clip "0"`

2. If you do not have an audio source, disable input audio by setting the unit control:

   `vtr.media.audio.input.port ""`

Crash: Audio Source Lacking

VST crashes after a few seconds if you do not have an audio source. To fix this problem, disable input audio by setting the unit control:

```
vtr.media.audio.input.port ""
```

Configuring for Booting From a Backup Plex

The system normally boots from the primary root plex, that is, *root.data.0.0*. If the primary plex becomes unavailable, you can either label the disks and swap master and slave, or you can use the following procedure to set up the system so that it can boot from the secondary root plex, for example, *system.data.1.0*.

1. From the System Maintenance Menu, choose Enter Command Monitor (5).
2. Display the PROM environment variables:

   ```
   >> printenv
   SystemPartition=dksc(0,1,8)
   OSLoadPartition=dksc(0,1,0)
   root=dks0d1s0
   ...
   ```

   The swap PROM environment variable (which is set below) is not displayed because it is not saved in NVRAM.
3. Reset the SystemPartition, OSLoadPartition, and root environment variables to the values of the disk partition that contains the alternate plex and the swap environment variable to have the value of the alternate swap partition. For example:

   ```
   >> setenv SystemPartition dksc(0,2,8)
   >> setenv OSLoadPartition dksc(0,2,0)
   >> setenv root dks0d2s0
   >> setenv swap /dev/dsk/dks0d2s1
   ```
4. Exit the Command Monitor and restart the system:

   ```
   >> exit
   ...
   Option? 1
   Starting up the system...
   ...
   ```

VST Crash Files

Inside the */usr/vtr/adm/crash* directory is a subdirectory for each program that crashes. If a program crashes, a directory is dynamically created with that program name. Inside that directory a core file is created containing the crash information.

When VST detects a program crash, it renames the core file to *core.number*, where *number* is the next incremental number. This scheme prevents core files from overwriting each other.

VST crash files are saved to aid SGI support personnel in diagnosing VST software errors.

Video Server Toolkit Control Summary

VST controls are configuration, control, and status variables implemented by VST system and device interface modules. These controls provide an easily extensible mechanism for configuring and controlling device-independent and device-specific features.

Note: This appendix contains tables summarizing the controls. For complete information on all controls, see the controls man page, vstcontrols(5M).

The controls fall into the following categories:

Note: In the Default column in the tables in this appendix, the entry

(none)

signifies that the control has no default setting. (Parentheses also enclose other explanatory comments in the Default column.) An entry of

none

signifies that the word "none" is the default.

System Controls

System controls are configuration variables not associated with a particular device or unit. These values can be set at startup with a system defaults file. After startup, they can also be queried and set via the MVCP commands *SGET* and *SSET*, respectively.

A system defaults file resides in */usr/vtr/config/system-defaults* and is named after the subsystem. For example, */usr/vtr/config/system-defaults/main* holds the settings for the main system defaults.

The rest of this section explains the controls, in these subsections:

System Controls for Clip Mirroring

Table A-1 summarizes clip mirror controls. The defaults file is */usr/vtr/config/system-defaults/clipmirror*.

Table A-1 System Controls for Clip Mirroring

Control	Default	Use
vtr.clipmirror.primary_server.hostname	"" (null string; mirroring disabled	Specifies name of server to mirror and starts local mirroring of the specified server's clip cache. Unsetting this control halts mirroring. See "Setting Up Primary and Redundant Servers" on page 71 in Chapter 5.
vtr.clipmirror.local_server.hostname	"" (null string; mirroring disabled	Specifies name of redundant server. See "Setting Up Primary and Redundant Servers" on page 71 in Chapter 5.
vtr.clipmirror.max_threads	20	Specifies the number of concurrent ftp threads allowed. Enter any integer from 0 to 100, based on the available network bandwidth between the two servers.
vtr.clipmirror.reconnect_interval	30	Specifies how often (in seconds) the clip mirror server tries to reconnect to the primary server after the connection is lost. Enter any integer greater than 0.

System Controls for Filesystems

Table A-2 summarizes filesystem controls. The defaults file is */usr/vtr/config/system-defaults/fs*.

Table A-2 System Controls for Filesystems

Control	Default	Use
vtr.storage.fs.grio.enabled	true	Enables or disables use of Guaranteed Rate I/O (GRIO)
vtr.storage.fs.grio.quantum_increment	50000	Specifies the increment in microseconds used to round down GRIO time quantums for reservations
vtr.storage.fs.grio.reap_interval	10000000000	Specifies the interval (in nanoseconds) for deallocating unused GRIO bandwidth reservations
vtr.storage.fs.grio.reserve_mode {standard | maximum | average}	standard	Specifies disk bandwidth reserved for accessing a clip
vtr.storage.fs.io_size	0 (not specified)	Specifies minimum number of bytes read from or written to a clip file
vtr.storage.fs.latency_warning_threshold	400	Specifies threshold (in milliseconds) for a warning message if the threshold is exceeded by a single I/O operation during playback or recording
vtr.storage.fs.major_alignment	0 (not specified)	Specifies alignment boundary used to optimize access to clip file
vtr.storage.fs.max_cue_reservations	1	Specifies maximum bandwidth reservations per real-time filesystem for cueing clips for playback
vtr.storage.fs.max_io_vectors	0 (use system maximum)	Specifies maximum I/O vectors used in a single I/O operation
vtr.storage.fs.minor_alignment	0 (not specified)	Specifies alignment boundary used to match the access requirements of the clip file

System Controls for the Main System

Table A-3 summarizes main system controls. The defaults file is */usr/vtr/config/system-defaults/main*.

Table A-3 System Controls for the Main System

Control	Default	Use
.vtr.main.heap.dump	0	Internal control; do not change
.vtr.main.heap.trace_heap_pid	0	Internal control; do not change
vtr.main.heap.trace_pid	0	Internal control; do not change
.vtr.main.list_threads	0	Internal control; do not change
.vtr.main.ulock_spin_count	0	Internal control; do not change
.vtr.main.ulock_yield_count	0	Internal control; do not change
vtr.main.log_level.console	0	Specifies maximum log level of messages sent to stdout
vtr.main.log_level.file	0	Specifies maximum log level of messages sent to server log file (usually, */usr/vtr/adm/logs/vtrlog*)
vtr.main.log_level.syslog	0	Specifies maximum log level of messages sent to system log (usually */var/adm/SYSLOG*)
vtr.main.thread.cpu_limit	0	Internal control; do not change
vtr.main.thread.cpu_limit_interval	10	Internal control; do not change
vtr.main.timing_standard	525	Specifies default timing standard for system (525 or 625)
vtr.main.unit_heap_size	8388608	Sets size (in bytes) of the heap allocated to each unit

System Controls for the Time Subsystem

Table A-4 summarizes main system controls. The defaults file is */usr/vtr/config/system-defaults/time*.

Table A-4 System Controls for the Time Subsystem

Control	Default	Use
vtr.time.slave_system_time	true (for channel 1, unless another channel specifically enabled)	Specifies whether system time (maintained by IRIX) is slaved to time code inputs for this time channel.
vtr.time.offset	-27000000 (Miranda Little Red) -6600000 (Horita PR-232)	Sets offset (in nanoseconds) between actual timebase and decoded input time code

System Controls for Sony Protocol Devices (VTR Emulation)

Table A-5 summarizes device controls for devices that use the Sony protocol.

Note: Settings in the defaults file */usr/vtr/config/system-defaults/sony* apply to all Sony-protocol control ports. Settings that apply to individual control ports are in the file */usr/vtr/config/system-defaults/sony_n*, where *n* is the serial port number.

Table A-5 System Controls for Sony Protocol Devices

Control	Default	Use
vtr.control.clip.name	(none)	Specifies clip to load on a Sony-protocol-controlled port
vtr.control.default_id.name	(none)	Specifies default ID (clip) to be loaded on a port controlled by the Sony protocol
vtr.control.description	(none)	Specifies a descriptive name for the control port
vtr.control.device_type_id	0xd803	Specifies device type identifier response to a DEVICE-TYPE protocol request
vtr.control.edit.delay	5	Specifies delay after an EDIT_ON or EDIT_OFF command is received before specified operation commences
vtr.control.ee.delay	5	Specifies delay after an EE_ON or EE_OFF command is received before the specified operation commences
vtr.control.ee.mode	player	Specifies whether the video output is delayed so that seamless playback / EE switching occurs (recorder mode), or whether video output matches time code reported by VST (player mode)
vtr.control.ee.record_select	true	Specifies whether output is switched to EE mode during recording.
vtr.control.output.idle_mode	hold	Specifies mode of output port when no playback is occurring
vtr.control.preread	false	Unsupported
vtr.control.sony.input_discard	true	Specifies whether input port is always active and discarding frames when no recording is occurring
vtr.control.sony.log_level	3	Specifies log level of control processor debugging messages
vtr.control.sony.timecode_log_level	0	Specifies log level of control processor time-code debugging messages

Table A-5 (continued) System Controls for Sony Protocol Devices

Control	Default	Use
vtr.control.superimpose.enabled	false	Specifies whether the superimpose display is enabled; not effective when clips with Rice or DVCPRO compressed video are played
vtr.control.timecode.mode	drop-frame	Specifies whether control port reports “drop-frame” or “non-drop-frame” time code for 525/59.94 video
vtr.edit.postroll	3:00	Specifies the postroll for an AUTOEDIT operation
vtr.edit.preroll	5:00	Specifies the preroll for an AUTOEDIT operation

Device Controls

Device controls are configuration, control, and status variables that are associated with a particular unit instance of a particular device interface. Device controls can be set at startup through the use of a *device-defaults* file. In addition, a unit’s device controls can be set and queried through the MVCP *GET* and *SET* commands.

A *device-defaults* file resides in */usr/vtr/config/device-defaults*. These files are loaded according to a hierarchical scheme enabling common control settings to be shared among devices.

When a unit device interface is initialized, control settings are loaded from *device-defaults* files in the following order:

1. Settings for all devices: ALL
2. Settings for node type: MEDIA or STORAGE
3. Settings for port type: VIDEO, NETWORK, DECK, or DISK
4. Settings for device class: *deviceclass*
5. Settings for device: *devicename*

For example, when the unit device interface for the DIVO_1 video device is initialized, settings are loaded from the device defaults files as follows:

1. ALL
2. MEDIA

3. VIDEO
4. DIVO
5. DIVO_1

This section discusses device controls in the following subsections:

Device Controls for DIVO, DIVO-DVC, and O2Video (mvp) Ports

Table A-6 summarizes device controls for supported video devices. The defaults file is

- */usr/vtr/config/device-defaults/DIVO* for a DIVO board; */usr/vtr/config/device-defaults/DIVO_n* is the port-specific version of the file, *n*, where *n* is the port number
- */usr/vtr/config/device-defaults/DIVO_DVC* for a DIVO-DVC board; */usr/vtr/config/device-defaults/DIVO-DVC_n* is the port-specific version of the file
- */usr/vtr/config/device-defaults/mvp* for an O2 workstation with O2Video

Table A-6 Device Controls for DIVO, DIVO-DVC, and O2Video (mvp) Ports

Control	Default	Use
vtr.edit.postroll	3:00	Specifies postroll duration for an edit performed by the unit. Not the same as the postroll set for a machine control port (such as for the Sony or Louth protocol).
vtr.edit.preroll	5:00	Specifies preroll duration for an edit performed by the unit. Not the same as the preroll set for a machine control port (such as for the Sony or Louth protocol).
vtr.media.audio.input.channel_map {<map> | *}	*	Specifies mapping from device input audio channels to clip audio channels

Table A-6 (continued) Device Controls for DIVO, DIVO-DVC, and O2Video (mvp) Ports

Control	Default	Use
vtr.media.audio.input.channels	2	Specifies number of audio input channels to be read from audio device
vtr.media.audio.input.channels.clip	-1	Specifies number of audio channels to put in a new clip
vtr.media.audio.input.edge_detect	0	Enables an internal test mode
vtr.media.audio.input.port	DefaultIn	Specifies audio input port
vtr.media.audio.input.rate	48000	Specifies audio sampling rate in samples per second
vtr.media.audio.input.sample.format	twos-complement	Specifies format of audio samples
vtr.media.audio.input.sample.width	24	Specifies the number of bits of precision in each audio sample
vtr.media.audio.input.skew	0	Specifies a skew value (in nanoseconds) for adjusting synchronization between input audio and video streams
vtr.media.audio.input.sync_source	<blank>	Specifies sync source for clocking input audio
vtr.media.audio.output.channel_map {<map> | *}	*	Specifies mapping from clip audio channels to device output audio channels
vtr.media.audio.output.channels	2	Specifies number of audio output channels to be written to audio device
vtr.media.audio.output.fade.duration.in	10000000	Specifies time (in nanoseconds) for fading in audio at a clip transition point
vtr.media.audio.output.fade.duration.out	10000000	Specifies nanoseconds for fading out audio at a clip transition point
vtr.media.audio.output.mute	false	Specifies whether normal audio samples are sent to attached audio port or whether zero samples (digital silence) are sent
vtr.media.audio.output.port	DefaultOut	Specifies audio output port
vtr.media.audio.output.rate	48000	Specifies audio sampling rate in samples per second
vtr.media.audio.output.sample.format	twos-complement	Specifies format of audio samples
vtr.media.audio.output.sample.width	24	Specifies bits of precision in each audio sample
vtr.media.audio.output.skew	0	Specifies a skew value (in nanoseconds) for adjusting synchronization between output audio and video streams

Table A-6 (continued) Device Controls for DIVO, DIVO-DVC, and O2Video (mvp) Ports

Control	Default	Use
vtr.media.audio.output.sync_source	Video	Specifies sync source for clocking output audio
vtr.media.audio.output.tone.frequency	1000	Specifies frequency (in Hz) of tone output when unit output mode is set to “image”
vtr.media.audio.output.tone.level	-20	Specifies level (in dB) of tone output when unit output mode is set to “image”
vtr.media.clip.format	default	Specifies format of new clips
vtr.media.clip.limit.enabled	true	Specifies whether starting or ending limits are enforced when clip is played back
vtr.media.clip.limit.end	*	Sets upper limit (end) for unit
vtr.media.clip.limit.start	*	Sets lower limit (start) for unit
vtr.media.clip.location.preset	* (none)	Sets location to which clip should be cued if playback or recording is started without the unit’s being manually cued
vtr.media.clip.segmented.format	(none)	Sets format for creating segmented clips
vtr.media.clip.segmented.insert_mode	overwrite	Specifies whether a new segment recorded into a segmented clip overwrites the existing frames in the clip, or whether it is inserted into clip at cued location
vtr.media.clip.start.mode	preset	Specifies mode for determining time code of first frame recorded into an empty clip if no time code is specified when unit is cued for recording
vtr.media.clip.start.preset	01:00:00:00	Specifies time code of first frame recorded into an empty clip if no time code is specified when unit is cued for recording and vtr.media.clip.start.mode is set to “preset”
vtr.media.ee.offset_frames	0	Sets offset (in video frames) between input video / audio and output video / audio when EE mode is active
vtr.media.ee_follows_record	false	Not supported
vtr.media.graphics.enabled	true	Specifies whether video output is displayed in a window (O2 systems only)
vtr.media.graphics.icon_name	Video server	Specifies icon name associated with graphical display window (O2 systems only)

Table A-6 (continued) Device Controls for DIVO, DIVO-DVC, and O2Video (mvp) Ports

Control	Default	Use
vtr.media.graphics.window_name	Video server	Specifies window name (O2 systems only)
vtr.media.input. abort_on_dropped_frame	false	Specifies whether a record operation is automatically aborted if an input frame is dropped
vtr.media.input.buffer_depth	3	Specifies memory for buffering incoming video data before it is written to storage
vtr.media.input.frame_log_level	4	Sets logging level for input frame debugging messages
vtr.media.input.idle_mode	off	Sets idle mode behavior for input port
vtr.media.input.mode	store	Sets recording behavior for input port
vtr.media.input.preemptible	true	Specifies whether unit can be preempted while idle if idle mode is set to a value other than “off”
vtr.media.input.timecode.preset	0	Sets initial value of free-running counter used by VITC and user bits free-running mode
vtr.media.input.trigger.mode.in	none	Sets type of trigger used to begin recording
vtr.media.input.trigger.mode.out	none	Sets type of trigger used to end recording
vtr.media.input.trigger.vitc.in	(none)	Sets VITC of first frame recorded when input trigger-in mode is VITC
vtr.media.input.trigger.vitc.out	(none)	Sets VITC of frame following last frame recorded when input trigger-out mode is VITC
vtr.media.input.userbits.mode	source	Specifies data stored in the time code user bits for each recorded frame
vtr.media.input.userbits.preset	0	Sets initial value of the free-running and running user bits counters
vtr.media.output.buffer_depth	3	Specifies amount of memory for buffering outgoing video data after it is read from storage system
vtr.media.output.cue_buffer_depth	1	Specifies amount of memory for buffering outgoing video data for a clip being cued to play after it is read from storage system
vtr.media.output.cued_mode	output	Sets unit’s output mode when it is cued
vtr.media.output.idle_mode	hold	Sets unit’s output mode when it is not playing

Table A-6 (continued) Device Controls for DIVO, DIVO-DVC, and O2Video (mvp) Ports

Control	Default	Use
vtr.media.output.mode	clip	Sets unit's output mode when it is playing
vtr.media.output.pause_at_limits	false	Specifies whether unit should pause (instead of stop) when start or end limit is reached
vtr.media.output.preemptible	true	Specifies whether unit can be preempted while idle if idle mode is a value other than "off"
vtr.media.output.speed.fast_forward	50000	Sets speed of playback when a fast forward (FF) command is issued
vtr.media.output.speed.rewind	-50000	Sets speed of playback when a rewind (REW) command is issued
vtr.media.output.timecode.preset	0	Sets initial value of free-running counter used by VITC and user bits free-running mode
vtr.media.output.userbits.mode	source	Specifies data sent in the time code user bits of each frame
vtr.media.output.userbits.preset	0	Sets initial value of the free-running and running user bits counters
vtr.media.port.enabled	true	Specifies whether this media port is enabled for use by VST applications
vtr.media.port.shared_access	true	Specifies whether access to this media port is shared between VST and other applications
vtr.media.port.use_fillpoints	auto	Internal control; do not change
vtr.media.video. beep_on_dropped_frame	no	Specifies whether a bell character (\007) is written to log when frame is dropped
vtr.media.video.input.bit_rate. feedback_gain	0.01	Specifies magnitude of feedback applied in adjusting compression quality when recording a clip using a lossy compression method with a target bit rate
vtr.media.video.input.bit_rate.target	25000000	Specifies target bit rate for compression stage when a clip is recorded using a lossy compression method
vtr.media.video.input.colorspace	ccir601	Specifies color-space encoding for recording input video
vtr.media.video.input.compare. command	(none)	Enables an internal diagnostics mode; do not change.

Table A-6 (continued) Device Controls for DIVO, DIVO-DVC, and O2Video (mvp) Ports

Control	Default	Use
vtr.media.video.input.compare.host	(none)	Enables an internal diagnostics mode; do not change.
vtr.media.video.input.compression.dithering	off	Not currently implemented
vtr.media.video.input.compression.precision	8	Specifies bits of precision used by compression algorithm to store video component data
vtr.media.video.input.compression.quality	0.94	Specifies value of quality parameter supplied to a lossy compression algorithm
vtr.media.video.input.compression.sampling	422	Specifies type of video component sampling used by compression algorithm
vtr.media.video.input.compression.type	default	Specifies type of compression for video frames being recorded
vtr.media.video.input.field_dominance	f1	Specifies input field dominance
vtr.media.video.input.format	normal	Specifies format of input video data
vtr.media.video.input.frame_mode	false	For compression modes that support it, specifies whether input video fields are stored as interleaved frames
vtr.media.video.input.hard_reset	false	When this control is set to true, the associated hardware input port is reset
vtr.media.video.input.interface_precision	10	Specifies precision of DIVO Serial Digital Interface (SDI) inputs
vtr.media.video.input.multiplex.transmission_rate	4	Specifies transmission rate expected for input from a multiplexed video stream (such as SDTI)
vtr.media.video.input.packing	R242_8	Specifies video component packing
vtr.media.video.input.restart_attempts	4	Specifies number of times VST tries to restart a failed input video transfer (recording) before raising an error on the unit
vtr.media.video.input.signal_loss.timeout	30000000000 (30 seconds)	Specifies how long (in nanoseconds) VST waits for a video input signal to be reacquired before signalling error condition on active unit
vtr.media.video.input.source	normal	Specifies source for input video
vtr.media.video.input.timing	auto	Specifies timing for input video

Table A-6 (continued) Device Controls for DIVO, DIVO-DVC, and O2Video (mvp) Ports

Control	Default	Use
vtr.media.video.input.vitc.mode	source	Specifies mode for storing VITC for frames as they are recorded into a clip
vtr.media.video.input.vitc_line_offset	14	Specifies video line used to get VITC from input video signal
vtr.media.video.output.boot_image. blank.pixel	0x28005c00	Specifies RGBA pixel value for DIVO output when video port is reset and vtr.media.video.output.boot_image.type is set to “blank”
vtr.media.video.output.boot_image. info.text	$port	Specifies text displayed in center of DIVO output when video port is reset and vtr.media.video.output.boot_image.type is set to “info”
vtr.media.video.output.boot_image. type	info	Specifies initial DIVO output when video port is reset
vtr.media.video.output. fast_shuttle_repeat_count	2	Specifies number of times each displayed frame is repeated when unit is playing forward or backward at a speed greater than normal
vtr.media.video.output. field_dominance	f1	Specifies output video field dominance
vtr.media.video.output.format	normal	Specifies format of output video data
vtr.media.video.output.hard_reset	false	When this control is set to true, the associated hardware output port is reset
vtr.media.video.output.image.name	black	Specifies image displayed when the output mode is “image” and vtr.media.output.image.type is “user”
vtr.media.video.output.image.type	bars	Specifies image displayed when unit output mode is “image”
vtr.media.video.output. interface_precision	10	Specifies precision of DIVO Serial Digital Interface (SDI) outputs
vtr.media.video.output. max_queued_frames	1	Specifies maximum video frames queued for video output
vtr.media.video.output.multiplex. transmission_rate	4	Specifies transmission rate for output to a multiplexed video stream (such as SDTI)
vtr.media.video.output.overlay.mode	fast	Specifies whether text overlay should be done with as little CPU overhead as possible or so that output text is always clean

Table A-6 (continued) Device Controls for DIVO, DIVO-DVC, and O2Video (mvp) Ports

Control	Default	Use
vtr.media.video.output.overlay.offset.horizontal	64	Specifies horizontal pixel offset of left edge of text overlay
vtr.media.video.output.overlay.offset.vertical	-64	Specifies horizontal pixel offset of upper edge of text overlay
vtr.media.video.output.overlay.text	(none)	Specifies text of overlay drawn into each outgoing video field
vtr.media.video.output.repeat_both_fields	no	Specifies whether both fields of video frame are displayed when frame is repeated
vtr.media.video.output.restart_attempts	4	Specifies the number of times VST tries to restart a failed output video transfer (playback) before raising an error on the unit
vtr.media.video.output.skip_on_drop	true	When one or more frames are dropped (and the previous frame is repeated) because of a delay in providing the next frame to the video port, specifies whether the same number of frames are automatically skipped in the clip so that clip playback duration does not change
vtr.media.video.output.sync.offset.vertical	0	Specifies vertical offset (in lines) applied to output video relative to incoming sync reference
vtr.media.video.output.sync_source	external	Specifies source of synchronization for video output
vtr.media.video.output.timing	system	Specifies default timing standard for video output
vtr.media.video.output.vitc.line_offset	14	Specifies video line where VITC is placed in output signal
vtr.media.video.output.vitc.mode	clip	Specifies data sent for VITC of each frame

Device Controls for Deck Control (Diaquest) Ports

Table A-7 summarizes deck and editing device (Diaquest) controls. The defaults file is */usr/vtr/config/device-defaults/dq*.

Table A-7 Device Controls for Deck Control Ports

Control	Default	Read Only	Use Summary
vtr.deck.error	(none)	Yes	Returns last error detected from deck control interface.
vtr.deck.status	(none)	Yes	Returns current deck transport status.
vtr.edit.coincidence.preroll	10	No	Specifies how far in advance of edit in point that VST attempts to sync. Under some circumstances, the actual coincidence point is a few frames later than specified, so this control should be set to at least 10 frames.
vtr.edit.preroll	5:00	No	Specifies deck preroll for edit operations
vtr.edit.postroll	3:00	No	Specifies deck postroll for edit operations
vtr.edit.status	{none)	Yes	Returns current status of edit operation that is in progress
vtr.media.video.frame_rate	{none)	Yes	Returns video frame rate reported by deck
vtr.media.video.sync_port	{none)	No	Specifies port number from which the deck control interface should derive video sync timing
vtr.media.output.mode	pb	No	Specifies current output mode of deck (pb for normal playback, ee for E-to-E)

Device Controls for DVB-ASI (Viewgraphics) Ports

Table A-8 summarizes DVB-ASI controls for DVB-ASI ports (Viewgraphics Dynamo MediaPump).

Note: The port-specific version of the defaults file, */usr/vtr/config/device-defaults/dvb-asi_n*, where *n* is the serial port number, is loaded only when VST is started. This file specifies only vtr.media.dvb-asi.port.total_bitrate and vtr.media.mpeg.pcr.frequency. All other controls are loaded from the non-port-specific defaults file */usr/vtr/config/device-defaults/dvb-asi*.

Table A-8 Device Controls for DVB-ASI Ports

Control	Default	Use
vtr.media.output.buffer_depth	0	Specifies amount of memory for buffering outgoing video data after it is read from the storage system
vtr.media.dvb-asi.fifo_buffer.size	65536	Specifies parameter (fifoBufferSize) setting to **mpMuxOpen()**
vtr.media.dvb-asi.io_buffer.size	65536	Specifies parameter (ioBufferSize) setting to **mpMuxOpen()**
vtr.media.dvb-asi.transfer_buffer.size	240640	Specifies parameter (transferBufferSize) setting to **mpMuxOpen()** and **muMuxStreamOpen()**
vtr.media.dvb-asi.transfer_buffer.depth	6	Specifies parameter setting (userBufferCount) to **mpMuxStreamOpen()**
vtr.media.dvb-asi.port	0	Selects the Viewgraphics DVB-ASI board for playing the loaded clip
vtr.media.dvb-asi.port.total_bitrate	0	Specifies total aggregate bit rate of DVB-ASI port
vtr.media.dvb-asi.drain_latency	4	Specifies the number of frames before the current stream finishes playing that the next stream begins cueing; internal control, do not change
vtr.media.dvb-asi.drain_on_loop	true	When set to false, subsequent passes of a loop mode playout start playing immediately without waiting for previous pass to finish; internal control, do not change
vtr.media.dvb-asi.seamless_playback	false	Specifies whether to use splicing to provide a seamless concatenation of MPEG streams
vtr.media.dvb-asi.halt_on_underflow	false	Specifies parameter (keepAlive) setting to **mpMuxOpen()**

Table A-8 (continued) Device Controls for DVB-ASI Ports

Control	Default	Use
vtr.media.mpeg.pid.video	-1 (do not remap)	Allows video PID of the clip played to be remapped to the one specified
vtr.media.mpeg.pid_offset.audio	1 (video PID + 1)	Specifies offset of audio PID from video PID when video PID is being remapped
vtr.media.mpeg.pid_offset.pcr {<PCR PID offset from video PID>}	0	Specifies offset of the PCR PID from video PID when video PID is being remapped
vtr.media.mpeg.pid.pmt	-1	Specifies that program map table (PMT) PID of the clip being played is to be remapped to specified PID
vtr.media.mpeg.bit_rate	-1	Specifies bit rate that the hardware uses to send out the clip; internal control, do not change
vtr.media.mpeg.pat.frequency	30	Specifies number of times per second the program association table (PAT) is sent
vtr.media.mpeg.pmt.frequency	30	Specifies number of times per second the PMT is sent
vtr.media.mpeg.pcr.frequency	30	Specifies number of times per second the program clock reference (PCR) is sent
vtr.media.mpeg.program_number	-1	Remaps the MPEG-2 transport steam program number to the specified program number

Device Controls for Filesystem Access

Table A-9 summarizes device controls for storage devices. The defaults file is */usr/vtr/config/device-defaults/fs*.

Table A-9 Device Controls for Filesystem Access

Control	Default	Use
vtr.storage.clear_after_play	false	Specifies that frames are automatically cleared from clip after they are played
vtr.storage.continue_after_error	true	Specifies that the unit continues playing after an I/O error
vtr.storage.io_log_level	4	Specifies log level for I/O logging messages
vtr.storage.max_open_files	8	Specifies maximum number of clip files that the unit can have open concurrently

Device Controls for MPEG Decoder (Vela) Ports

Table A-10 summarizes device controls for Vela devices. The defaults file is */usr/vtr/config/device-defaults/vela*; */usr/vtr/config/device-defaults/vela_n* is the device-specific version of the file {where *n* is the serial port number).

Table A-10 Device Controls for MPEG Decoder Ports

Control	Default	Use
vtr.media.output.buffer_depth	0	Specifies memory for buffering outgoing video data after it is read from the storage system
vtr.media.output.sync_source	Internal	Specifies sync source for the video output
vtr.media.output.write_size	65536	Sets size of SCSI write operations used to send MPEG stream to decoder
vtr.media.vela.error.retry	false	Specifies whether software should attempt error recovery when hardware failure is encountered
vtr.media.vela.error.recover_realtime	true	Specifies whether software should skip over the time that hardware is not available when attempting error recovery
vtr.media.vela.scsi.timeout	0	Sets number of seconds to wait before a SCSI timeout is detected
vtr.media.vela.command_latency	0	Sets command latency of hardware and software in frames
vtr.media.vela.query_decoder_location	true	Specifies whether to query Vela decoder for current location
vtr.media.vela.seamless_playback	false	Specifies whether to use splicing to provide a seamless concatenation of MPEG streams
vtr.media.vela.preblack	1	Sets number of frames blacked out by Vela decoder before video playout
vtr.media.audio.output.level	-1.0	Specifies attenuation of Vela decoder's audio output volume in dB

RS-422 Pinouts

The 6-inch mini-DIN8 to male DB-9 adapter cable (SGI part number 018-0650-001) makes the DB-9 end of the cable the same as that of an Origin 200 server ttyd1 / ttyd2 for RS-232 only, not for RS-422.

The RS-422 Mini-DIN8 to DB-9 adapter cannot be used with Video Server Toolkit.

Table B-1 summarizes the SGI RS-422 pinouts at the DB-9 end of the Macintosh mini-DIN8 to female DB-9 adapter cable.

Table B-1 Apple Macintosh Female DB-9 Pinout

Pin Number	Purpose
2	
3	
4	TX+
5	TX-
6	
7	
8	RX+
9	RX-

Video Server Toolkit Man Pages

Table C-1 summarizes VST man pages:

Table C-1 Man Pages

Man Page	Describes
vst(1)	Video Server Toolkit
vtrstart(1)	Startup
vtrstop(1)	Shutdown
vtrstat(1)	Status
vtrclip(1)	Utility for adding and removing clips to or from clip cache
mcpanel(1)	Media control panel
mcclips(1)	Clip manager
mcstat(1)	Status display
mccompstats(1)	Compression monitor
vstcontrols(5)	VST controls
mvcp(5)	Multiport Video Computer Protocol
vvtr(1)	VST server
vtrd(1)	VST daemon
vtrvfutil(1)	VST vframe clip utility
vtrftpd(1)	VST FTP daemon

You can list the VST man pages after installing them by entering the following command:

```
% versions long vst_eoe | grep man
```

Index

D

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V

W

X