ULTRIX SCSI/CAM Architecture

Release Notes and Installation Instructions

Order Number: AA-PJB8A-TE

January 1992

Product Version:	SCSI/CAM for ULTRIX RISC Version 4.2, Version 1.0
Operating System and Version:	ULTRIX Version 4.2 (RISC)

digital equipment corporation
Maynard, Massachusetts

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by Digital or its affiliated companies.

The following are trademarks of Digital Equipment Corporation:

ALL-IN-1, Bookreader, CDA, DDIF, DDIS, DEC, DECnet, DECstation, DECsystem, DECUS, DECwindows, DTIF, MASSBUS, MicroVAX, Q-bus, ULTRIX, ULTRIX Mail Connection, ULTRIX Worksystem Software, UNIBUS, VAX, VAXstation, VMS, VT, XUI, and the DIGITAL logo.

UNIX is a registered trademark of UNIX System Laboratories, Inc.

Contents

About This Manual

1 SCSI/CAM Version 1.0 Release Notes

2 SCSI/CAM Version 1.0 Installation Instructions

About This Manual

These release notes and installation instructions contain information needed by those who are installing Version 1.0 of SCSI/CAM for ULTRIX RISC Version 4.2.

Audience

This document is intended for the person responsible for installing, managing, and maintaining SCSI/CAM for ULTRIX RISC Version 4.2, Version 1.0. Programmers and other users of the SCSI/CAM software will also find information that affects their work in this document.

Organization

This document is organized as follows:

Chapter 1	SCSI/CAM for ULTRIX RISC Version 4.2, Version 1.0 Release Notes Describes the issues that are currently known regarding the SCSI/CAM software.
Chapter 2	Installation Instructions Describes how to install SCSI/CAM for ULTRIX RISC Version 4.2 software.

Related Documentation

Readers of this document are assumed to be familiar with the following documents:

- American National Standard for Information Systems, *SCSI-2 Common Access Method: Transport and SCSI Interface Module*, working draft, X3T9.2/90-186 Terms used in this guide, such as CAM Control Block (CCB), are defined in that document. Copies may be purchased from Global Engineering, 2805 McGaw St, Irvine, CA 92714, telephone 800-854-7179.
- American National Standard for Information Systems, *Small Computer Systems Interface - 2* (SCSI - 2), X3T9/89-042

The following document contains information that pertains to SCSI/CAM software:

- *Guide to Writing Device Drivers for the ULTRIX SCSI/CAM Architecture Interfaces*
  This manual describes the SCSI/CAM Architecture interfaces. It also describes how to write device drivers for the SCSI/CAM implementation..

Conventions

`%`

The default user prompt is your system name followed by a right angle bracket. In this manual, a percent sign (`%`) is used to represent this prompt.

`%` **`cat`**

A regular constant-width typeface is used for code examples, system prompts in interactive examples, and names of commands and other literal strings in text. A bold constant-width typeface is used for typed user input in interactive examples and for routines in function definitions.

`filename`

In examples, syntax descriptions, and function definitions, this typeface indicates variable values.

cat *file*

In syntax definitions, a bold sans serif typeface is used for literal strings and a sloping sans serif typeface is used for variable values.

`cat`(1)

A cross-reference to a reference page include the appropriate section number in parentheses. For example, a reference to `cat`(1) indicates that you can find the material on the `cat` command in Section 1 of the reference pages.

SCSI/CAM Version 1.0 Release Notes 1

1.1 Introduction

The SCSI/CAM for ULTRIX RISC Version 4.2, Version 1.0 software runs on the following processors:

- DECstation 2100
- DECstation 3100
- DECstation 5000, Model 100 series
- DECstation 5000, Model 200 series
- DECstation 5100
- DECstation 5500

The Q-bus SCSI controller, KZQSA, for the DECstation 5500, is not supported.

This chapter describes the issues that are currently known regarding the SCSI/CAM Version 1.0 software.

1.2 System Configuration Change

The following major system configuration change results from installing the CAM subsystem:

- The CAM subsystem completely replaces the older SCSI device drivers.

1.3 Device Support with Standard Drivers

Digital provides common data structures, macros, and routines with SCSI/CAM software for SCSI/CAM peripheral device driver writers to use. They are described in Chapter 3 of the *Guide to Writing Device Drivers for the ULTRIX SCSI/CAM Architecture Interfaces.* The following issues pertain to device support with standard drivers:

- The Version 1.0 kit includes support for Disks, CDROM/Audio, and Tape devices.
- Disk and CDROM/Audio devices are accessed through the rz# labels in the configuration file and device special file entries.
- Tape devices are accessed through the tz# labels in the configuration file and device special file entries.
- The tape device driver supports four ‘‘densities’’ per device type.

- The Request Sense information is stored by the drivers for access by user level programs. This information is always the latest sense information returned by devices.

1.4 Attaching Third-Party SCSI Devices

The CAM device drivers can access third-party SCSI devices using the `/usr/sys/data/cam_data.c` file. See Chapters 3 and 11 of the *Guide to Writing Device Drivers for the ULTRIX SCSI/CAM Architecture Interfaces* for information about adding third-party SCSI devices.

1.5 CAM CCB Timeout Values During Boot

With the current software, the system's timers are not enabled during the booting process. The CAM subsystem keeps a polled timer of two seconds during SCSI bus probing.

1.6 User Agent Interaction with Virtual Memory Subsystem Problem

There is an interaction between the virtual memory (VM) subsystem and the User Agent driver that puts processes using the User Agent driver to sleep.

The user program indicates that there are buffers that need to be locked, for example, the `data`, `autosense`, and `CDB` buffers in the UAGT_CAM_CCB structure. The User Agent driver uses the VM subsystem to lock pages into memory. If a page is already locked, the VM subsystem puts the process to sleep waiting for the locked page to be freed.

If the user program locates two CAM buffers on the same page, the VM subsystem locks the first buffer on the page. When the User Agent driver attempts to lock the second buffer on that page, the VM subsystem discovers that the page is already locked, puts the process to sleep, and waits for the page to be freed. The locked page will never be freed because the process is asleep, so the process hangs in the sleep state. This interaction can occur with a small data buffer and autosense buffer, or when the data buffer follows the sense buffer in the data space.

To avoid this interaction, the user program must make sure that no allocated buffers contain common pages. The SCSI/CAM kit includes the file `uagt_lib.c`, which contains two local routines that the user program can use to allocate and free page-aligned data buffers.

The first of these routines, `malloc_palign`, allocates and returns to the caller a system-page-aligned buffer with enough space added to make one more page. This allows the pointers to be adjusted to the next page boundary. A local linked list keeps copies of the original and adjusted data buffer addresses. This list is used by the second routine, `free_palign`, to free the correct buffer. The `malloc_palign` routine follows:

```
char *
malloc_palign( size )
    int size;       /* size of the buffer the user wants page aligned */
{
}
```

The free_palign routine returns a previously allocated buffer to the system using the local linked list generated by the malloc_palign routine. The free_palign routine can be used only to free buffers allocated using the malloc_palign routine. The free_palign routine follows:

```
void
free_palign( pa_addr )
    char *pa_addr;       /* the page aligned address */
{
}
```

1.7 TLZ04 Tape Drive Problem

Due to a hardware problem with the TLZ04 tape drives, the initial unit attention is not reported from the device, after the system is booted. The problem occurs if a tape is loaded in the drive when the system is booted.

The unit attention is not reported when the initial Test Unit Ready command is issued to the unit. This prevents the SCSI/CAM tape driver from executing the auto density code within the driver. The auto density code determines the density and blocking factors of the current tape and validates the density information for the unit.

This is not a problem for write commands to the unit, because writes default to the user-selected density. Read commands to the unit cause the following message to appear on the console screen:

```
ctape_strat: READ case and density info not valid.
```

1.8 TZ30 Tape Drive Problem

Due to a hardware problem with the TZ30 tape drive, hard errors can occur on every attempt to access the device. The problem occurs when a SCSI Bus Reset command is issued if a tape is already loaded in the TZ30 drive when the system is booted. Workarounds are to not have a tape loaded on boot or to unload the tape and reload it after the system boots.

1.9 TZK08 (8mm EXABYTE Tape Drive) Problem

Currently, there is a file size limit of 2.147 gigabytes for tape drives on a per-open basis. The TZK08 cartridge is capable of storing slightly over the 2.147 gigabyte limit. Once the limit is reached, an error of EINVAL is returned to the application.

The problem appears either when you do a single tar operation on multiple mount points with the data to be stored totaling more than the 2.147 gigabyte limit, or when an application is run that exceeds the limit on a per-open basis.

1.10 User Agent Driver Issues

The following issues relate to the User Agent driver:

- The User Agent driver issues RELEASE SIM QUEUE CCBs to the CAM subsystem when the close routine is called. The User Agent driver keeps track of all SIM queues that have been reported as ‘‘Frozen’’.

Note

> In UNIX, when the last application program closes a device driver, the `close` routine is called. If other applications are using the User Agent driver when the `close` routine is called, those applications cannot depend on the SIM queues being released when they exit.

- The DEVIOCGET `ioctl()` command is supported. The `file` utility displays the message `special device UAGT`.

1.11 Aborting a SCSI space Command to a Tape Drive

When you use CTRL/C to abort a SCSI `space` command to a TK50 tape, the drive will properly free the bus; however, it will not terminate the SCSI `space` command.

1.12 Installing a Third Party Device Driver

This section describes how to add third party SCSI/CAM device drivers to the CAM subsystem. Refer to the *Guide to Writing and Porting VMEbus and TURBOchannel Device Drivers* for information about adding a device driver to the ULTRIX kernel. See Chapter 3 of the *Guide to Writing Device Drivers for the ULTRIX SCSI/CAM Architecture Interfaces* for information about the SCSI/CAM common routines that you can use when adding a new driver.

Due to the layered architecture of the SCSI/CAM product, the SIM layers and the Configuration driver are the primary pieces of the system configuration process. The SIM module contains the `probe` routines and the Configuration driver contains the `slave` and `attach` routines. The Configuration driver calls other slave and attach routines to allow third party drivers to handle slave and attach processing for the SCSI peripheral devices for which they were written.

The file `/usr/sys/io/cam/cam_config.c` contains an array called `cam_peripheral_drivers[]`. This array is used by the configuration driver's slave and attach routines to call the third party device driver's slave and attach routines. Entries must be made in this table to add a new driver. If a match succeeds on the `ui->ui_devname` and `cpd->cpd_name` strings, the third party slave and attach routines are called. The following code routines are contained in the configuration driver, and are included here for reference only.

```
/***********************************************************************

   cdrv_slave() - Configuration Driver Slave Routine.

   Description:
     This routine is called at auto-configuration time to determine
   if a slave exists or not.  During the auto-configuration process,
   this routine locates the configured driver in the peripheral driver
   table.  If located, the peripheral driver's slave routine is called
   with the standard arguments to allow it to perform its own slave
   initialization.

   PLEASE NOTE: This routine is only needed for an ULTRIX configuration
          process and won't be required for DEC OSF/1 configuration.


   Inputs: ui = Pointer to unit information structure.
```

```
          csr = Virtual address of the CSR address.


   Outputs:    None.

   Return Value:
          Returns 0 / 1 = Slave isn’t alive / Slave is alive.

 ********************************************************************/
int
cdrv_slave (ui, csr)
register struct uba_device *ui;
caddr_t csr;
{
     register struct cam_peripheral_driver *cpd;
     register int i;
     PRINTD(ui->ui_ctlr, NOBTL, NOBTL, CAMD_CONFIG, ("cdrv_slave called0));
     for (cpd = cam_peripheral_drivers, i = 0; i < cam_pdrv_entrys;
          cpd++, i++) {
         if (strcmp (cpd->cpd_name, ui->ui_devname) == 0) {
          return ((*cpd->cpd_slave)(ui, csr));
          /*NOTREACHED*/
         }
     }
     return (0);
}

/* ---------------------------------------------------------------------- */

/*********************************************************************

   cdrv_attach() - Configuration Driver Attach Routine.

   Description:
     This routine is called by the auto-configuration code after a
   slave has been found.  The purpose of this routine is to locate the
   configured driver in the peripheral driver table.  Once located,
   the peripheral driver’s attach routine is called with the standard
   arguments to allow it to perform its own attach initialization.

   PLEASE NOTE:This routine is only needed for ULTRIX configuration
          process and won’t be required for DEC OSF/1 configuration.


   Inputs:ui = Pointer to unit information structure.


   Outputs:    None.


   Return Value:
          Returns 0 / 1 = Attach Failed / Attach Successful.

          The return value is ignored by auto-configuration code.

 ********************************************************************/
int
cdrv_attach (ui)
register struct uba_device *ui;
{
     register struct cam_peripheral_driver *cpd;
     register int i;
     PRINTD(ui->ui_ctlr, NOBTL, NOBTL, CAMD_CONFIG,
         ("cdrv_attach called0));
     for (cpd = cam_peripheral_drivers, i = 0; i < cam_pdrv_entrys;
          cpd++, i++) {
         if (strcmp (cpd->cpd_name, ui->ui_devname) == 0) {
          return ((*cpd->cpd_attach)(ui));
          /*NOTREACHED*/
         }
     }
     return (0);
}

/* ---------------------------------------------------------------------- */
```

1.13 Error Reporting

This section contains information about error reporting.

1.13.1 The cam_report Utility

The `cam_report` utility supplements the `uerf` utility. The `cam_report` utility has supplemental knowledge of SCSI/CAM error packets and entries for additional capabilities for debugging the SCSI/CAM subsystem.

The utility takes either no arguments, one argument, or two arguments. With no arguments, the utility looks up the current hostname and opens the error log file with the filename, `/usr/adm/syserr/syserr.HOSTNAME`, to decode the SCSI/CAM error information the file contains.

The utility prints usage information when passed the `-h` argument.

The utility can also be given another system error log file in which to look for SCSI/CAM error information when the `-f` and filename arguments are given. For example:

```
cam_report -f /usr/adm/syserr/syserr.ebusy
```

1.13.2 The uerf Utility

To see all the CAM error reports when you use the `uerf` utility, use the `-o full` option. For example:

```
uerf -o full | more
```

1.14 Maximum Data Transfer Size for N-Buffered I/O

When asynchronous (N-Buffered) I/O is used, the maximum data transfer size is 64 Kbytes of data. Most utilities have a method for turning off N-Buffered I/O.

1.15 Major/Minor Number Mapping for RZ and TZ Devices

The major/minor device number mapping that the SCSI/CAM software uses is different from the mappings for TZ and RZ devices in the `/dev` directory.

The SCSI/CAM software was written to support Logical Unit Numbers (LUNs), but the current ULTRIX release does not support LUNs. The number of bits required to identify a SCSI device with LUN support is greater than 8, so a 16-bit major/minor device number is not sufficient.

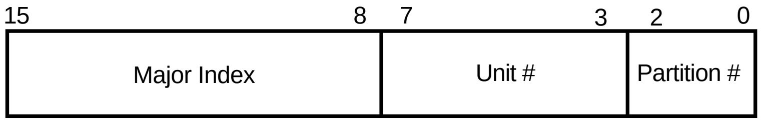
To minimize changes to the SCSI/CAM software and support utilities, the ULTRIX major/minor number definitions in the /dev directory are mapped by the CAM disk and tape drivers to the RZ/TZ kernel entry points expected by the CAM dev_t argument. The CAM common routines require that the CAM dev_t major/minor device number mapping be used when a dev_t argument is passed to them.

1.15.1 Mapping for RZ Disk Devices

The minor number bits for RZ devices contained in the /dev directory are defined as follows:

```
bits 0-2 = partition number
bits 3-7 = unit number
        (0-7   for devices on controller 0)
        (8-15  for devices on controller 1)
        (16-23 for devices on controller 2)
        (24-31 for devices on controller 3)
```

The 16 bits are allocated as follows:

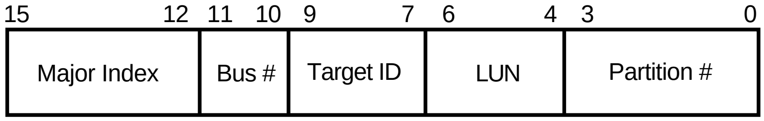


ZK−0492U−R

The bit definitions for the dev_t argument expected by the CAM common routines are as follows:

```
bits 0-3   = partition number (0-16)
bits 4-6   = LUN (0-7)
bits 7-9   = target ID (0-7)
bits 10-11 = controller number (0-3)
```

The 16 bits are allocated as follows:



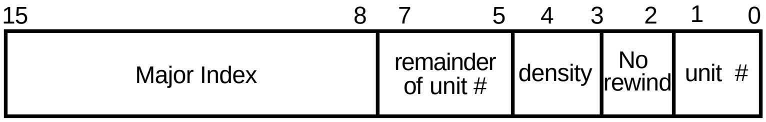
ZK−0493U−R

1.15.2 Mapping for TZ Tape Devices

The minor number bits for TZ devices contained in the /dev directory are defined as follows:

```
bits 0-1 = part of unit number
bit  2   = no rewind
bits 3-4 = density
bits 5-7 = rest of unit number
```

The 16 bits are allocated as follows:

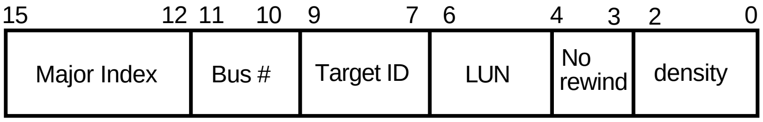


ZK−0494U−R

The bit definitions for the `dev_t` argument expected by the CAM common routines are as follows:

```
bits 0-2   = density
bits 3     = no rewind
bits 4-6   = LUN (0-7)
bits 7-9   = target ID (0-7)
bits 10-11 = controller number (0-3)
```

The 16 bits are allocated as follows:



ZK−0495U−R

1.16 Updated Reference Pages

This section contains the `mt`(1) and `crash`(8) reference pages, which have been updated to reflect SCSI/CAM software.

The updated `crash`(8) reference page describes a new `crash` utility that is included with the SCSI/CAM software to replace the existing ULTRIX Version 4.2 `crash` utility. If the CAM subsystem is ever unloaded from your system, reload the original `crash` utility, `/usr/bin/crash`, which is one of the files you save before you start your installation of SCSI/CAM software.

Name

mt – magnetic tape manipulating program

Syntax

mt [**–f tapename**] *command* [*count*]

Description

The `mt` command permits the operation of a magnetic tape drive.

Options

The **–f** flag option uses the specified tape device (next argument) in place of either that tape device defined by your TAPE environment variable (.login or .profile) or /dev/nrmt0h.

Some operations may be performed multiple times by specifying *count* . By default, `mt` performs the requested operation once.

The *command* argument defines the operation to be performed. Only as many characters as are required to uniquely identify a command need be specified.

The following is a list of commands:

bsf Backspace *count* files.

bsr Backspace *count* records.

cache Allows `mt` to use the cache buffer on a tape drive that has the cache buffer feature.

clhrdsf Clear hardware/software problem. Works with tape drives which use the TMSCP tape controller interface `tms`(4). This command is restricted to root access only.

clserex Clear serious exception. Works with tape drives which use the TMSCP tape controller interface `tms`(4).

clsub Clear subsystem. Works with tape drives which use the TMSCP tape controller interface `tms`(4). This command is restricted to root access only.

eof, weof Write *count* end-of-file marks at the current position on the tape.

eotdis Disable end-of-tape detection. When the end of tape is reached, the tape will run off the reel. Only the superuser can issue this command. The command remains in effect for the device until end-of-tape detection is enabled with the **eoten** command.

eoten Enable end-of-tape detection. When the end-of-tape markers are reached, the tape is halted on the reel, between the two end-of-tape markers. Only the superuser can issue this command. The command remains in effect for the device until end-of-tape detection is disabled with the **eotdis** command. This is the default mode after a system boot.

fsf Forward-space *count* files.

fsr Forward-space *count* records.

online Bring a SCSI tape to the ready state if a tape is in the tape path. Currently, this command is only supported in the SCSI CAM tape driver. Not all SCSI tape drives support this functionality.

nocache Disables the use of the cache buffer for any tape drive that has the cache buffer feature.

offline, rewoffl Rewind the tape and place the tape unit off-line.

retension Retensions the tape. Retension means moving the tape one complete pass between EOT and BOT.

rewind Rewind the tape.

seod Space to end of recorded data. Currently, this command is only supported in the SCSI CAM tape driver. Not all SCSI tape drives support this functionality.

status Print status information about the tape unit.

Examples

This example shows how to rewind the tape `rmt0l`:

```
mt –f /dev/rmt0l rewind
```

This example shows how to backspace the tape nmt1h three files:

```
mt –f /dev/nrmt1h bsf 3
```

This example shows how to write two end-of-file marks at the current position on tape nmt6h:

```
mt –f /dev/nrmt6h eof 2
```

Return Values

In shell scripts, `mt` returns a 0 exit status when the operations were successful, 1 if the command was unrecognized, and 2 if an operation failed.

Files

/dev/rmt?h or /dev/rmt?l
Raw magnetic tape interface with rewind when closed

/dev/nmt?h or /dev/nmt?l
Raw magnetic tape interface with no rewind when closed

See Also

dd(1), tar(1), ioctl(2), mtio(4), tms(4), environ(7)

Name

crash – examine system images

Syntax

/usr/bin/crash [*system*] [*namelist*]

Description

The `crash` utility is an interactive program that lets you examine the core image of the operating system. This utility has facilities for interpreting and formatting the various control structures in the system and certain miscellaneous functions that are useful when perusing a dump.

The arguments to the `crash` utility are the file name where the *system* image can be found and a *namelist* file to be used for symbol values.

The default values are `/dev/mem` and `/vmunix`; hence, the `crash` utility specified without arguments can be used to examine an active system. If a *system* image file is given, it is assumed to be a system core dump and the default process is set to be that of the process running at the time of the crash. This is determined by a value stored in a fixed location by the dump mechanism.

Commands

Use the following input to the `crash` utility:

command [*options*] [*structures*]

If used, *options* modify the format of the printout. If a specific structure element is not specified, all valid entries are used. For example, **proc 12 15 3** prints only process table slots 12, 15, and 3, but **proc** prints the entire process table in standard format.

In general, those commands that perform I/O with addresses assume hexadecimal on 32-bit machines and octal on 16-bit machines.

The commands include the following:

user [*process table entry*] [**proc address*] [#*pid*]

Aliases: `uarea`, `u_area`, `u`.
Prints the user structure of the named process as determined by the information contained in the process table entry. If an entry number is not given, the information from the last executing process is printed. Swapped processes produce an error message since their uareas are swapped.

ufile [*process table entry*] [**proc address*] [#*pid*]

Prints the open file table for the given process.

trace [–] [*process table entry*] [**proc address*] [#*pid*]

Aliases: `t`.
Generates a kernel stack trace of a process. The process is either a process slot number, an address of process slot, or the running process. If the process is not running, the trace begins at the `pcb`. If an entry

number is not given, the information from the last executing process will be printed. It is not possible to trace the executing process on a running system. Using the – flag allows the registers and variables for each stack frame to be dumped.

stack [*process table entry*] [**proc address*] [*#pid*]

Aliases : `s`, `stk`.
Generates a kernel stack dump of a process. This is an unformatted display of the kernel stack.

proc [–r] [*process table entry*] [**proc address*] [*#pid*]

Aliases: `p`.
Formats the process table. The **–r** option causes only runnable processes to be printed.

proclock [–r] [*process table entry*] [**proc address*] [*#pid*]

Shows the SMP sleep locks held by a non-running process. Same optional arguments as **proc**.

pcb [*process table entry*] [**proc address*] [*#pid*]

Prints the process control block of the current process. The process control block is a part of the user area (VAXen only). If no entry number is given, the information from the last executing process will be printed.

ps [*process table entry*] [**proc address*] [*#pid*]

Prints the process slots, process id's, and process names for all processes.

pcb [*process table entry*] [**proc address*] [*#pid*]

Prints the process control block for the given process.

ppte [*process table entry*] [**proc address*] [*#pid*]

Prints the pte's associated with the given process.

spt

Dumps the system page table.

cmap <*page frame number*>

Prints the memory freelist or the cmap for the given PFN.

cmap –i *index*

Prints the core map for the given coremap index.

cmap –h *index*

Prints the core maps for the given hash starting at index.

cmap –b *block*

Prints the core maps hashed on the given block.

cmap –a *at*

Prints the core map at the given address.

gnode [–] [*gnode table entries*] [**gnode address*]

Aliases: `gno`, `g`.
Formats the gnode table.

gnode –maj <*major number*>

Formats all gnodes with the given major number.

gnode –min <*minor number*>

Formats all gnodes with the given minor number.

gnode –fs <*mount slot number*>

Formats all gnodes for a given file system slot.

gnode –gno <*gnode number*>

Formats all gnodes with the given gnode number.

gnode –uid <*user id*>

Formats all gnodes owned by the given uid.

gnode –gid <*group id*>

Formats all gnodes owned by the given gid.

gnode –lmod <*file permissions*>

Formats all gnodes with the given protection. The command **gnode –lmod 777** finds all gnodes that allow read/write/execute permission to every one.

gnode –hmod <*file type*>

Formats all gnodes with the given file type. The **gnode –hmod 2** command finds all gnodes that are character special files.

gnode –amod <*file modesfR*>

Formats all gnodes that match the given modes exactly.

gnode –amod 20777

Finds all gnodes for character devices that allow read/write/execute permission to everyone.

gnode –all

Displays a more extensive list of the gnode's contents.

gnode –lock

Shows the SMP lock contained in the gnode.

gfree

Prints the list of all inactive gnodes.

block [–] [*gnode table entries*] [**gnode address*]

Prints the gnode data block addresses.

rnode [*gnode table entries*] [**gnode address*]

Alias: `v`.
Prints the associated rnode values for gnodes that are remote (via NFS).

file [*file table entries*] [**file address*]

Alias: `f`.
Formats the file table.

cred *address*

Prints the credentials at the given address.

crred *address*

Verifies the references of a credential at the given address.

crcheck

Verifies all references to all credentials.

mount [–`s`] [*mount table entries*] [**mount address*]

Aliases: `mnt`, `m`.
Formats the mount table. The `–s` option gives a abbreviated format.

fsdata [*mount table entries*] [**mount address*]

Alias: `df`.
Prints the fsdata structure associated with a mount entry.

mntinfo [*mount table entries*] [**mount address*]

Alias: `mi`.
Prints the mntinfo data structure associate with an NFS file system.

buf [–] [*buffer headers*] [**buffer header address*]

Aliases: `hdr`, `bufhdr`.
Formats the system buffer headers. By using the – option, all buffer headers (including ones marked invalid) are printed.

bufgp [*gnode slots*] [**gnode address*]

Alias: `cache`.
Prints the buffer headers associated with the given gnode.

buflock

Prints the buffer headers on the locked list.

buflru

Prints the buffer headers on the lru list.

bufage

Prints the buffer headers on the aged list.

bufempty

Prints the buffer headers on the empty list.

buffer [*format*] [*list of buffers*]

Alias: `b`.
Prints the data in a system buffer according to *format* . If *format* is omitted, the previous *format* is used. Valid formats include `decimal`, `octal`, `hex`, `character`, `byte`, `directory`, `gnode`, and `write`. The last creates a file in the current directory (see the Files section) containing the buffer data.

text [*text table entries*] [**text address*]

Aliases: `txt`, `x`.
Formats the text table.

ftext

Alias: `freet`.

Prints the list of free texts.

callout

Aliases: `calls`, `call`, `c`, `timeout`, `time`, `tout`.

Prints all entries in the callout table.

arp [–] [*address*]

Prints the arp table. The – option prints the entire table.

socket [*file slot*] [**file address*]

Prints the socket structure associated with the given file slot.

tty [–] [*process slot*] [**proc address*] [*#pid*]

Alias: `term`.

Prints the terminal structure attached to a process. The – option allows for the raw, canonical, and output clists.

tty –clist

Includes clists in display of the tty struct.

tty -addr *address*

Prints the contents of a tty structure at the specified address.

map [*map names*]

Formats the named system map structures.

nm [*symbols*]

Prints the symbol value and type as found in the *namelist* file.

ts [*text addresses*]

Finds the closest text symbols to the given addresses.

ds [*data addresses*]

Finds the closest data symbols to the given addresses.

od [*symbol name or address*] [*count*] [*format*]

Aliases: `dump`, `rd`.

Dumps *count* data values starting at the symbol value or address given according to *format*. Allowable formats are `octal`, `longoct`, `decimal`, `longdec`, `character`, `hex`, or `byte`.

dis *address* [*address*]

Disassembles starting at the first address and continuing until the second address. These addresses may be symbolic (that is, syscall+33).

stat

Prints useful statistics pertaining to the buffer cache, dnlc, namei translation cache, and others.

dupreq

Displays the contents of the duplicate request cache. This is useful in finding out the recent history of NFS requests made to the server.

mbuf *address*

Displays the `mbuf` chain starting at the given address.

inpcb –udp –tcp

Displays the `inpcb` chain of the corresponding protocol, or both TCP and UDP if no protocol is specified.

client

Displays the client table, which contains client handles used to initiate rpc requests.

sync

Resynchronizes the proc, mount, gnode, buffer, file and other internal tables up to the current state of `/dev/kmem`. This command is useful for looking at changing values in runing kernels. However, you chould not use it when looking at `vmcore` files.

svcxprt *address*

Prints the svcxprt structure located at address.

cam -controller

Prints SCSI/CAM controller information.

cam -edt

Prints CAM Equipment Device Table (EDT) information.

cam -target

Prints SCSI target information.

cam -devtab

Prints SCSI/CAM Device Descriptor Structure entry information.

cam -cmd

Prints SCSI message and command data.

cam -transfer

Prints SCSI transfer information per target.

cam -bbr

Prints SCSI Bad Block Replacement data.

cam -error

Prints SCSI/CAM error information.

cam -recovery

Prints SCSI direct access device recovery information.

cam -all

Prints all SCSI bus information.

scs

Traverses data structures in the System Communications Services (SCS) tree and displays the contents of the data structures.

scs -cb *address*

Displays the contents of an SCS connection block data structure at the specified address.

scs -cib *address*

Displays the contents of an SCS connection information block data structure at the specified address.

scs -pb *address*

Displays the contents of an SCS path block data structure at the specified address.

scs -pib *address*

Displays the contents of an SCS path information block data structure at the specified address.

scs -sb *address*

Displays the contents of an SCS system block data structure at the specified address.

scs -sib *address*

Displays the contents of an SCS system information block data structure at the specified address.

lock [-all] *address*

Prints the SMP lock structure located at address. The `all` flag displays all global SMP locks.

mscp

Traverses both the mscp disk and tape subsystems, and prints the data structures for class blocks, connection blocks, unit blocks, and active request blocks.

mscp -disk

Traverses the mscp disk subsystem and prints the data structures for the mscp class block, mscp connection blocks, mscp unit blocks, and active request blocks.

mscp -tape

Traverses the tmscp tape subsystem and prints the data structures for the tmscp class block, tmscp connection blocks, tmscp unit blocks, and active request blocks.

mscp -config

Traverses both the mscp disk and tape subsystems, and display the system configuration. This is done by printing summary information from the class, connection, and unit blocks.

mscp -connb *address*

Prints the contents of an mscp connection block at the specified address.

mscp -classb *address*

Prints the contents of an mscp class block at the specified address.

mscp -unitb *address*

Prints the contents of an mscp unitb block at the specified address.

mscp -reqb *address*

Prints the contents of an mscp request block at the specified address.

mscp -dtable

Displays all the elements of the mscp disk unit table. Unused elements of the array will be specified as NULL.

mscp -ttable

Displays all the elements of the tmscp tape unit table. Unused elements of the array will be specified as NULL.

!

Escapes to the shell.

#[*history*]

Repeats the last command. If a number is given (that is, **#5**), that command number is re-executed.

#h

Alias: `history`,
Shows the history list.

q

Exits from `crash`.

?

Prints a synopsis of commands.

Aliases

There are built-in aliases for many of the *formats* as well as those listed for the commands. Some of them are:

byte

b.

character

char, c.

decimal

dec, e.

directory

direct, dir, d.

hexadecimal

hexadec, hex, h, x.

gnode

gno , g.

longdec

ld, D.

longoct

lo, O.

octal

oct, o.

write

w.

Restrictions

Many of the flags are abbreviated making them difficult to interpret. A source listing of the system header files would be helpful when using the `crash` utility.

Examing the stack of the current process on a running system and processes running at the time of a crash does not work.

Files

`/usr/include/sys/*.h`

Header files for table and structure info

`/dev/mem`

Default system image file

`/vmunix`

Default namelist file

`buf.#`

Files created containing buffer data

See Also

mount(8), nm(1), ps(1), sh(1), stty(1), pstat(8)

SCSI/CAM Version 1.0 Installation Instructions 2

2.1 Introduction

These instructions describe how to install the the SCSI/CAM for ULTRIX RISC Version 4.2, Version 1.0 kit. These instructions assume that you have already installed ULTRIX, Version 4.2, on your system and have logged in as the superuser.

Note

> If you have installed the optional software subset SHRMIRROR100, which contains disk shadowing software, delete the subset before you start your installation and reinstall it after your SCSI/CAM software installation is complete. The SCSI/CAM software overwrites parts of the disk shadowing software.

Before you begin your installation of the SCSI/CAM software, save the following files:

```
/usr/bin/crash
sys/conf/mips/files.mips
sys/data/presto_data.c
sys/data/tc_option_data.c
sys/h/devio.h
sys/h/errlog.h
sys/h/mtio.h
sys/h/rzdisk.h
sys/machine/common/conf.c
sys/machine/mips/debug.c
sys/machine/mips/debug.h
sys/net/net/conf_net.c
sys/net/net/netisr.h
```

2.2 Installing the SCSI/CAM Software

This section describes how to install the SCSI/CAM software. The system used in the sample procedure is named MYSYSTEM.

1. Load the tape labeled OPEN SCSI CAM V1.0 (RISC) FOR ULTRIX V4.2 in your tape drive.

2. Enter a sequence like the following to change to the `root` directory and rename `/vmunix` to another filename so that it cannot be overwritten by the CAM kernel:

   ```
   # cd /
   # mv /vmunix /vmunix.nocam
   ```

3. Use the `df` utility to determine the amount of space available on your disk. The SCSI/CAM software requires about 1.7 Mbytes of disk space. For example:

   ```
   # df
   Filesystem   Total    kbytes   kbytes   %
   node         kbytes   used     free     used  Mounted on
   /dev/rz9a      15343    9624     4185    70%   /
   /dev/rz9g     227079   85913  118459    42%   /usr
   ```

4. Choose the device from which you want to load the CAM subsets and invoke the `setld` utility. For example:

   ```
   # setld -l /dev/rmt0h
   ```

   The following messages appear:

   ```
   Please make sure your installation tape is mounted and on-line.
   Are you ready (y/n)?
   ```

5. Confirm that your installation tape is mounted and online, then enter `y`. The following menu appears:

   ```
   Positioning Tape

   *** Enter Subset Selections ***

   The subsets listed below are optional:
    1) CAM Kernel Configuration Files2) CAM Base System Files

    3) All of the Above
    4) None of the Above
    5) Exit without installing subsets

   Enter your choice(s):
   ```

   Enter `3` to install the CAM subsets.

6. The following message appears:

   ```
   You are installing the following subsets:
      CAM Kernel Configuration Files          CAM Base System Files

   Is this correct? (y/n):
   ```

   Enter `y` to confirm your choice. Messages appear stating that the subsets are being copied and verifying their installation.

7. When the system prompt appears, run the `/etc/doconfig` program to configure your system:

   ```
   # /etc/doconfig
   ```

   The program prompts you to enter your system name and asks a series of questions about the system configuration file. See `doconfig`(8) in the *ULTRIX Reference Pages* for more information.

8. When the program finishes, it displays a message showing the path and location of the new `vmunix` kernel.

9. When the kernel has built successfully, save the existing default kernel and rename the kernel containing the CAM code to be the default kernel. For example:

   ```
   # mv /vmunix /vmunix.nocam
   # mv /sys/MIPS/MYSYSTEM/vmunix /vmunix
   ```

10. Shut down your system. When the boot prompt appears, reboot your system.

How to Order Additional Documentation

Technical Support

If you need help deciding which documentation best meets your needs, call 800-343-4040 before placing your electronic, telephone, or direct mail order.

Electronic Orders

To place an order at the Electronic Store, dial 800-234-1998 using a 1200- or 2400-baud modem from anywhere in the USA, Canada, or Puerto Rico. If you need assistance using the Electronic Store, call 800-DIGITAL (800-344-4825).

Telephone and Direct Mail Orders

Your Location	**Call**	**Contact**
Continental USA, Alaska, or Hawaii	800-DIGITAL	Digital Equipment Corporation P.O. Box CS2008 Nashua, New Hampshire 03061
Puerto Rico	809-754-7575	Local Digital Subsidiary
Canada	800-267-6215	Digital Equipment of Canada Attn: DECdirect Operations KAO2/2 P.O. Box 13000 100 Herzberg Road Kanata, Ontario, Canada K2K 2A6
International	—————	Local Digital subsidiary or approved distributor
Internal*	—————	SSB Order Processing - WMO/E15 *or* Software Supply Business Digital Equipment Corporation Westminster, Massachusetts 01473

* For internal orders, you must submit an Internal Software Order Form (EN-01740-07).

Reader's Comments

ULTRIX SCSI/CAM Architecture
Release Notes and
Installation Instructions
AA-PJB8A-TE

Please use this postage-paid form to comment on this manual. If you require a written reply to a software problem and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Thank you for your assistance.

Please rate this manual:	Excellent	Good	Fair	Poor
Accuracy (software works as manual says)	□	□	□	□
Completeness (enough information)	□	□	□	□
Clarity (easy to understand)	□	□	□	□
Organization (structure of subject matter)	□	□	□	□
Figures (useful)	□	□	□	□
Examples (useful)	□	□	□	□
Index (ability to find topic)	□	□	□	□
Page layout (easy to find information)	□	□	□	□

What would you like to see more/less of? ______________________________
____
____

What do you like best about this manual? ______________________________
____
____

What do you like least about this manual? ______________________________
____
____

Please list errors you have found in this manual:

Page Description

________ ______________________________
________ ______________________________
________ ______________________________
________ ______________________________
________ ______________________________

Additional comments or suggestions to improve this manual:

____
____
____
____

What version of the software described by this manual are you using? ____________

Name/Title ______________________________ Dept. ____________
Company ______________________________ Date ____________
Mailing Address ______________________________
____________________ Email ____________________ Phone ____________

Do Not Tear − Fold Here and Tape

digital™

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST−CLASS MAIL PERMIT NO. 33 MAYNARD MA

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
OPEN SOFTWARE PUBLICATIONS MANAGER
ZKO3−3/Y32
110 SPIT BROOK ROAD
NASHUA NH 03062−2698

Do Not Tear − Fold Here

Cut
Along
Dotted
Line

Reader's Comments

ULTRIX SCSI/CAM Architecture
Release Notes and
Installation Instructions
AA-PJB8A-TE

Please use this postage-paid form to comment on this manual. If you require a written reply to a software problem and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Thank you for your assistance.

Please rate this manual:	Excellent	Good	Fair	Poor
Accuracy (software works as manual says)	□	□	□	□
Completeness (enough information)	□	□	□	□
Clarity (easy to understand)	□	□	□	□
Organization (structure of subject matter)	□	□	□	□
Figures (useful)	□	□	□	□
Examples (useful)	□	□	□	□
Index (ability to find topic)	□	□	□	□
Page layout (easy to find information)	□	□	□	□

What would you like to see more/less of? ______________________________
____
____

What do you like best about this manual? ______________________________
____
____

What do you like least about this manual? ______________________________
____
____

Please list errors you have found in this manual:

Page ________ Description ____
________ ____
________ ____
________ ____
________ ____
________ ____

Additional comments or suggestions to improve this manual:
____
____
____
____

What version of the software described by this manual are you using? ____________

Name/Title ________________________________ Dept. ____________
Company ________________________________ Date ____________
Mailing Address ____
________________________ Email ____________________ Phone ____________

Do Not Tear − Fold Here and Tape

digital™

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST−CLASS MAIL PERMIT NO. 33 MAYNARD MA

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
OPEN SOFTWARE PUBLICATIONS MANAGER
ZKO3−3/Y32
110 SPIT BROOK ROAD
NASHUA NH 03062−2698

Do Not Tear − Fold Here

Cut
Along
Dotted
Line