NAME

camdd - CAM data transfer utility

SYNOPSIS

camdd <**-i|o** *pass=pass_dev|file=filename,bs=blocksize,[...]*> [**-C** *retry_count*] [**-E**] [**-m** *max_io*] [**-t** *timeout*] [**-v**] [**-h**]

DESCRIPTION

The **camdd** utility is a sequential data transfer utility that offers standard read(2) and write(2) operation in addition to a mode that uses the asynchronous pass(4) API. The asynchronous pass(4) API allows multiple requests to be queued to a device simultaneously.

camdd collects performance information and will display it when the transfer completes, when **camdd** is terminated or when it receives a SIGINFO signal.

The following options are available:

-i | **-o** *args* Specify the input and output device or file. Both **-i** and **-o** must be specified. There are a number of parameters that can be specified. One of the first two (file or pass) MUST be specified to indicate which I/O method to use on the device in question.

pass=dev Specify a pass(4) device to operate on. This requests that **camdd** access the device in question be accessed via the asynchronous pass(4) interface.

The device name can be a pass(4) name and unit number, for instance "pass0", or a regular peripheral driver name and unit number, for instance "da5". It can also be the path of a pass(4) or other disk device, like "/dev/da5". It may also be a bus:target:lun, for example: "0:5:0".

Only pass(4) devices for SCSI disk-like devices are supported. ATA devices are not currently supported, but support could be added later. Specifically, SCSI Direct Access (type 0), WORM (type 4), CDROM (type 5), and RBC (Reduced Block Command, type 14) devices are supported. Tape drives, medium changers, enclosures etc. are not supported.

file=path Specify a file or device to operate on. This requests that the file or device in question be accessed using the standard read(2) and write(2) system calls. The file interface does not support queueing multiple commands at a time. It does support probing disk sector size and capacity information, and tape blocksize and maximum transfer size information. The file interface supports standard files, disks, tape drives, special devices, pipes and standard input and output. If the file is specified as a "-", standard input or standard output are used. For tape devices, the specified blocksize will be the size that **camdd** attempts to use to write to or read from the tape. When writing to a tape device, the blocksize is treated like a disk sector size. So, that means **camdd** will not write anything smaller than the sector size. At the end of a transfer, if there isn't sufficient data from the reader to yield a full block, **camdd** will add zeros on the end of the data from the reader to make up a full block.

bs=N
: Specify the blocksize to use for transfers. **camdd** will attempt to read or write using the requested blocksize.

Note that the blocksize given only applies to either the input or the output path. To use the same blocksize for the input and output transfers, you must specify that blocksize with both the **-i** and **-o** arguments.

The blocksize may be specified in bytes, or using any suffix (e.g. k, M, G) supported by expand_number(3).

offset=N
: Specify the starting offset for the input or output device or file. The offset may be specified in bytes, or by using any suffix (e.g. k, M, G) supported by expand_number(3).

depth=N
: Specify a desired queue depth for the input or output path. **camdd** will attempt to keep the requested number of requests of the specified blocksize queued to the input or output device. Queue depths greater than 1 are only supported for the asynchronous pass(4) output method. The queue depth is maintained on a best effort basis, and may not be possible to maintain for especially fast devices. For writes, maintaining the queue depth also depends on a sufficiently fast reading device.

mcs=N
: Specify the minimum command size to use for pass(4) devices. Some devices do not support 6 byte SCSI commands. The da(4) device handles this restriction automatically, but the pass(4) device allows the user to specify the SCSI command used. If a device does not accept 6 byte SCSI READ/WRITE commands (which is the default at lower LBAs), it will generally accept 10 byte SCSI commands instead.

debug=N
: Specify the debug level for this device. There is currently only one debug

level setting, so setting this to any non-zero value will turn on debugging. The debug facility may be expanded in the future.

-C *count* Specify the retry count for commands sent via the asynchronous pass(4) interface. This does not apply to commands sent via the file interface.

-E Enable kernel error recovery for the pass(4) driver. If error recovery is not enabled, unit attention conditions and other transient failures may cause the transfer to fail.

-m *size* Specify the maximum amount of data to be transferred. This may be specified in bytes, or by using any suffix (e.g. K, M, G) supported by expand_number(3).

-t *timeout* Specify the command timeout in seconds to use for commands sent via the pass(4) driver.

-v Enable verbose reporting of errors. This is recommended to aid in debugging any SCSI issues that come up.

-h Display the **camdd** usage message.

If **camdd** receives a SIGINFO signal, it will print the current input and output byte counts, elapsed runtime and average throughput. If **camdd** receives a SIGINT signal, it will print the current input and output byte counts, elapsed runtime and average throughput and then exit.

EXAMPLES

```
camdd -i pass=da8,bs=512k,depth=4 -o pass=da3,bs=512k,depth=4
```

Copy all data from da8 to da3 using a blocksize of 512k for both drives, and attempt to maintain a queue depth of 4 on both the input and output devices. The transfer will stop when the end of either device is reached.

```
camdd -i file=/dev/zero,bs=1M -o pass=da5,bs=1M,depth=4 -m 100M
```

Read 1MB blocks of zeros from /dev/zero, and write them to da5 with a desired queue depth of 4. Stop the transfer after 100MB has been written.

```
camdd -i pass=da8,bs=1M,depth=3 -o file=disk.img
```

Copy disk da8 using a 1MB blocksize and desired queue depth of 3 to the file disk.img.

```
camdd -i file=/etc/rc -o file=-
```

Read the file /etc/rc and write it to standard output.

```
camdd -i pass=da10,bs=64k,depth=16 -o file=/dev/nsa0,bs=128k
```

Copy 64K blocks from the disk da10 with a queue depth of 16, and write to the tape drive sa0 with a 128k blocksize. The copy will stop when either the end of the disk or tape is reached.

SEE ALSO

cam(3), cam(4), pass(4), camcontrol(8)

HISTORY

camdd first appeared in FreeBSD 10.2

AUTHORS

Kenneth Merry <*ken@FreeBSD.org*>