NAME

snapshot − collect data snapshots from an ADC stream

SYNOPSIS

snapshot [*options*] *files*...

DESCRIPTION

The **snapshot** program is responsible for reading data from the array hardware and on request writing sections of the buffered data out to file. The program uses root permission to initialise the Comedi interface to the analog-digital conversion hardware, and to access the elevated real-time priority scheduling options available under Linux. Once this is done, it drops privileges.

snapshot is adapted for collection of short snapshots of the data stream with the possibility of specifying pre- and post-trigger durations. The typical application is in behavioural experimentation, where a behaviour of interest is recognised during its execution. On recognising the interesting behaviour, pulling the trigger results in the capture of a window of data starting some time before the trigger instant and extending some time after it.

While **snapshot** can be commanded to generate a sequence of snapshots that covers a long period of contiguous input stream, for continuous data one should probably use the **grab** program, which is much simpler and less programmable but is specialised for streaming data from the analog hardware to its standard output.

COMMANDS

Once running, **snapshot** waits for command messages via its command socket (set by −−*snapshot*, see below). To each message it returns a reply indicating success (OK) or failure (NO) with an error description. The command verbs may be abbreviated to single characters and may be given as upper or lower case.

The commands accepted by the program are:

Quit **snapshot** winds up its activities and exits tidily.

? rest is a "ping" command which generates the reply "**! rest**" from the program. Useful for determining whether **snapshot** is ready for commands. *rest* can be any string.

Param *parameter list*
allows one to set parameters such as sampling frequency, Comedi streaming buffer size and capture window duration at run time. These can be set individually or in combination since the command takes a comma-separated list of parameter assignments. For instance, *freq=100e3,window=15* would set the sampling frequency to 100 [kHz] and the capture window to 15 [s].

The **Param** command can be issued when the program is in pre-initialised state, on start-up or after a **Halt** command has been executed.

Init asks the program to initialise the data capture system. It checks the current set of parameters for consistency, initialises the Comedi interface, maps and locks the Comedi buffer into memory, and creates and validates the Comedi data capture command. It does not begin acquisition. This command is only valid when the capture process is stopped.

snapshot replies to an **Init** command with the number of ADC channels available and the channel skew in [ns], that is, the period between acquisition of individual samples and hence the time offset between corresponding samples from consecutive channels.

Go requests **snapshot** to begin data acquisition by executing the prepared Comedi command and dealing with the data arriving in the Comedi buffer as required. This command is only valid after initialisation by **Init**.

Once **Go** is executed, the **snapshot** program enters an "armed" state until data is received from Comedi, at which point it switches to "running" state.

If data from the hardware fails to arrive within a reasonable period, the ADC command is aborted

and **snapshot** switches to an error state. In error state, only **Param** (or **Quit**) commands can be issued to the program. Successful completion of a **Param** command returns the program to normal state.

Halt terminates the current data acquisition operation and returns **snapshot** to the pre-initialised state, in which parameter changes can be made. This is only valid when a **Go** has been given.

Snap *parameter list*

instructs **snapshot** to capture some data from the ongoing data stream. It is an error to request a snapshot unless the program is collecting data (in "armed" or "running" state). A correct snapshot command issued when not running receives a "NO" (error) response.

The *parameter list* specifies the range of data to capture, specified either as start and end points or start point and length. It is an error to specify both an endpoint and a length. The parameter *length* gives the number of samples to collect; it is rounded to the nearest multiple of 8 channels so as to include the requested data range.

Start and end points can be given either as sample indices with respect to the start of data acquisition -- using the parameters *start* and *finish* -- or as timepoints (in [ns] from the Unix epoch) using parameters *begin* and *end*. The time-based parameters are converted to sample indices using the time when the program switched from "armed" to "running" state. The initial point of the range is rounded down to a multiple of the number of channels, while the endpoint is rounded up likewise.

The optional *count* parameter allows one to request a repeating snapshot.

The mandatory *path* parameter gives the name of the snapshot, which is used when constructing the disk directory to hold the data. It may be a relative or absolute path; relative paths are with respect to the current working directory (see **Dir** below).

Dir *path=<filepath>*

instructs **snapshot** to change working directory to that given by the mandatory *path* parameter. **snapshot** creates the new working directory; for relative paths the new directory is relative to the *−−snapdir* directory. Only one level of tree may be created in one step.

Ztatus *name=<snapshot>*

reports the status of snapshots finished or in progress. It can take an optional *name* parameter to limit its attention to a single snapshot. For in-progress snapshots a short status line is generated. For completed snapshots the final status is that produced by the completion (finish or error) of the operation.

Asking for the status of a completed snapshot frees up the resources it is holding.

SNAPSHOT STRUCTURE

When requested by a **Snap** command, **snapshot** writes out chunks of data from the Comedi buffer to disk in a sub-directory determined by the *path* parameter. If the given *path* is relative, it is interpreted relative to the current working directory recently set by a **Dir** command, or to the *−−snapdir* directory itself if no working directory has been set.

Within the new sub-directory, snapshot data files are written. If the *count* parameter is set, *count* distinct files are created (by default a single file is written).

The name of each file is the 16 byte zero-padded hexadecimal string derived from the index of the first sample it contains, to which is appened a .s16 suffix. The data in the file consists of 16 bit little-endian signed integers representing +1 [V] to -1 [V] analog values. The file length is a multiple of 8 samples.

If an error occurs during the capture of data for a file, that file is aborted. The disk file is removed, and subsequent files of the failing snapshot are also aborted.

OPTIONS

snapshot has a collection of long options that follow the GNU convention of starting with two dashes ('-'). Many of them have an equivalent short option form. All the long options can also be set using environment variables; those options marked (*) can also be set by the **Param** command.

−h | −−help
Show summary of options.

−v | −−verbose
Become more verbose in output; may be repeated for greater effect

−q | −−quiet
Become more quiet. A *−q* alone turns off all output, including warning messages.

−−version
Show the program version number and exit.

−s | −−snapshot=*url*
specifies the ZeroMQ URL of the communication endpoint to talk to (the snapshot program's command socket). The default value is *ipc://snapshot-CMD*, though any of ZeroMQ's standard inter-process transports may be used. For example, a *tcp://...* URL allows the **snapshot** program to run on a different host from the processes (such as **snapchat**, **trig** or **array-cmd**) from which it receives commands.

−−tmpdir=*path*
path specifies the directory in which temporary files are to be created. This directory should exist. The default is /tmp.

−S | −−snapdir=*path*
path specifies the directory in which snapshots should be written. This directory may exist; if not, it will be created by **snapshot**. A relative path here is relative to the temporary files directory. The default is 'snap', i.e. snapshots are by default created in /tmp/snap/.

−d | −−dev=*path*
path is the Comedi device to use to talk to the hardware. The default is /dev(comedi0.

−f | −−freq=*freq* **(*)**
specifies the sampling frequency in [Hz] per channel. The default is 312.5 [kHz], i.e. a total sampling frequency of 2.5 [MHz] across the 8 channels provided by the hardware.

−r | −−range=*500|750* **(*)**
sets the conversion range of the ADC hardware. The default is 750[mV] peak. A smaller range of 500[mV] peak is offered by the USBDUXfast hardware currently used.

−b | −−bufsz=*size* **(*)**
specifies the *size* of the Comedi streaming buffer in [MiB]. It requires root permission to increase the maximum buffer size beyond that set for the device. Maximum buffer size can be pre-set using **comedi-config** to avoid this. The default value is 32 [MiB].

−w | −−window=*duration* **(*)**
determines the minimum duration of valid data present in the streaming buffer at any time. The *duration* in [s] fixes the maximum size of a snapshot, since a single snapshot must fit within the Comedi buffer (currently). The default value is 10 [s].

−B | −−bufhwm=*fraction* **(*)**
determines the proportion of the streaming buffer available for data. The remainder is used as a no-man's land between the newly arriving data from the hardware and the oldest valid data in the buffer.

−P | −−rtprio=*pri*
If this option is given, the program attempts to set real-time scheduling mode for reading data from Comedi and writing to the filesystem. The *pri* argument must be a valid priority for a SCHED_FIFO process; see **sched_setscheduler(2)** for more information. This option implies the

two real-time options below. The default is not to request real-time treatment.

−R | −−rdprio=*pri*
If this option is given, the program attempts to set real-time scheduling mode for reading data from Comedi. The *pri* argument must be a valid priority for a SCHED_FIFO process; see **sched_setscheduler(2)** for more information. The default is not to request real-time treatment.

−W | −−wrprio=*pri*
If this option is given, the program attempts to set real-time scheduling mode for writing data to the filesystem. The *pri* argument must be a valid priority for a SCHED_FIFO process; see **sched_setscheduler(2)** for more information. The default is not to request real-time treatment.

−u | −−user=*name*
is used, when **snapshot** is running as root, to set the user id **snapshot** will switch to when it drops privileges. The *snapdir* directory is given this owner if it is created by **snapshot**. Default is not set. If not specified the value is taken from the process owner. Note that **snapshot** can be started as root, if necessary to have the capabilities it needs, but it will not run as root.

−g | −−group=*name*
is used likewise to set the process group. Default is not set. If *--user* is given alone, that user's set of group memberships is used. If *--group* is given alone, the process owner is used as the user. If *--group* is given, the user's supplementary groups are not loaded. Multiple *--group* arguments result in multiple groups being installed.

−m | −−ram=*size*
sets the amount of RAM in [MiB] requested for handling data transfers. Up to this much RAM will be locked by **snapshot** in the course of scheduling and executing capture requests. Default is 64 [MiB].

−c | −−chunk=*size*
specifies the default size in [kiB] of an atomic chunk of data to transfer from the capture buffer to a snapshot file. **snapshot** will never transfer more than this, nor less than half this amount, in a single transfer. Default is 1024 [kiB], i.e. 1 [MiB].

−o | −−wof=*0-1*
is the "write-overbooking fraction", between zero and one, which specifies how much extra data can be enqueued for writing beyond the size of the transfer RAM. Default is 0.5.

EXIT STATUS

snapshot exits normally after receiving the Quit command. It exits with code 1 for parameter errors and with code 2 for errors in establishing the necessary ZeroMQ messaging environment.

ENVIRONMENT

Each long option available on the command line has a matching environment variable through which it can also be set. Environment variables are matched using a case-insensitive comparison. Command-line arguments take precedence over environment variable values (which in turn take precedence over built-in defaults).

REAL-TIME OPERATION

To use the real-time priority scheduling for reading data from Comedi, **snapshot** needs to be able to change its scheduling priority. This requires running as root, possessing CAP_SYS_NICE capability (see capabilities(7) for details) or having the necessary resource limit sufficiently relaxed. The real-time priority limits can be set using the **prlimit** command.

NOTES

The time at which sampling began is determined when the first batch of data is received fom Comedi. This timepoint is some short time after the data actually starts arriving in the kernel buffers and may be several millseconds late in the current implementation. Very short capture windows may miss their intended wall-clock target because of this offset.

The parameters **−−chunk**, **−−freq**, **−−window**, **−−bufsz** and **−−bufhwm** are interdependent. The code enforces the following constraints:

the product of **−−window** and **−−freq**, a number of samples, must fit inside the product of **−−bufsz** and **−−bufhwm** which is the active section of the buffer.

the part of the buffer outside the active section must be at least as big as two **−−chunk** blocks.

The quotient of **−−chunk** and **−−freq**, that is, the time it takes to receive a chunk from the ADC hardware, is used to determine the loop frequency of the thread that processes incoming data.

SEE ALSO

snapchat(1a)**, adc**(1a)**, array-cmd**(8a)**, trig**(1a)**, comedi-config**(8)**, sched_setscheduler**(2)**, prlimit**(1)**, capabilities**(7)**.**