# SLS Detector text clients manual

March 12, 2018

## 1 Introduction

This program is intended to control the SLS detectors via command line interface.
This is the only way to access all possible functionality of the detectors, however it is often recommendable to avoid changing the most advanced settings, rather leaving the task to configuration files, as when using the GUI or the API provided.

The command line interface consists in four main functions:

**sls_detector_acquire** to acquire data from the detector

**sls_detector_put** to set detector parameters

**sls_detector_get** to retrieve detector parameters

**sls_detector_help** to get help concerning the text commands

Additionally the program **slsReceiver** should be started on the machine expected to receive the data from the detector.

If you need control a single detector, the use of the command line interface does not need any additional arguments.

For commands addressing a single controller of your detector, the command `cmd` should be called with the index `i` of the controller:
`sls_detector_clnt i:cmd`
where `sls_detector_clnt` is the text client (put, get, acquire, help).

In case more than one detector is configured on the control PC, the command `cmd` should be called with their respective index `j`: `sls_detector_clnt j-cmd` where `sls_detector_clnt` is the text client (put, get, acquire, help).

To address a specific controller `i` of detector `j` use:
`sls_detector_clnt j-i:cmd`

For additional questions concerning the indexing of the detector, please refer to the *SLS Detectors FAQ* documentation.

## 2 Acquisition

By calling:
`sls\_detector\_acquire [j-]`
the detector `j` is started and the data are acquired, postprocessed and written to file according to the configuration and setup of the measurements.
A progress index of the acquisition in percentage is shown on the command line.

For additional questions concerning the acquisition flow, please refer to the *SLS Detectors FAQ* documentation.

## 3 Detector setup

```
sls\_detector\_put [j-][i:]var arg
```

is used to configure the detector parameters `var` with the value `arg`.
It returns the actual value of the variable, as when calling `sls\_detector\_get` with the same command.

### 3.1 Standard commands

**config fname** Load the configuration file fname.
Examples of configuration files are available in the directory `examples`. This should be done every time the configuration of the detectors(s) changes or the control PC is rebooted. Must be executed on all the control PCs, before executing other commands.

**parameters fname** Load the parameter file fname.
The syntax of the commands in the parameter file is exactly the same as for the command line interface. Can be used to load a standard mode of acquisition and/or to hide advanced parameters from the final user. Examples of parameter files are available in the directory `examples`.

**settings sett** Configures the settings of the detector. Refer to detailed detector documentation for more details:
for MYTHEN sett can be: standard, fast, highgain;
for GOTTHARD sett can be: veryhighgain, highgain, mediumgain, lowgain, dynamicgain;
for EIGER sett can be: standard, highgain, lowgain.

**threshold ev** For photon counting detectors, sets the detector threshold in eV. The detector should be properly calibrated, otherwise standard calibration coefficients are used, which can give an uncertainty up to a few keVs.

**timing sync** Sets the timing mode of the detector. Can be auto, gating (works only if at least one of the signals is configured as gate_in), trigger (works only if at least one of the signals is configured as trigger_in), ro_trigger (works only if at least one of the signals is configured as ro_trigger_in),

triggered_gating (works only if one of the signals is configured as gate_in and one as trigger_in).
Refer to the detailed documentation to understand how the different timing modes work.

**outdir path** Defines the path where the output files will be saved to.

**fname prefix** Defines the prefix of the file name for the data output.
The final file name will be:
`prefix[_d`*d*`][_S`*v0*`][_s`*v1*`][_p`*p*`][_f`*f*`]_`*i*`.`*ext*
where:
*d* is the controller index, in case of data receiver and more than one controller;
*v0* is the scan0 variable with the desired precision, if scan0 is enabled;
*v1* is the scan1 variable with the desired precision, if scan1 is enabled;
*p* is the position index, if different positions are configured;
*f* is the frame index of the first frame stored in the file, if many frames and cycles are configured;
*i* is the file index;
*ext* is the file extension e.g. *.raw* for MYTHEN and EIGER raw data, *.dat* for MYTHEN processed data.

**index i** Sets the starting index of the file i at the beginning of the acquisition (automatically incremented for each measurement).

**enablefwrite b** Enables (1) or disables (0) file writing.

**exptime ts** Sets the exposure time of a single acquisition to ts (in s). It is overridden in case the detector is in gating mode.
Refere to detailed documentation to understand how the different timing modes work.

**subexptime ts** Sets the subexposure time of a single subacquisition to ts (in s) in EIGER autosumming mode (=*dr 32*). Refer to detailed documentation to understand how the different timing modes work.

**period ts** Sets the frames period (in s). It is overridden in case the detector is in gating mode.
Refer to detailed documentation to understand how the different timing modes work.

**delay ts** Sets the delay after trigger in triggered mode (in s).
Refer to the detailed documentation to understand how the different timing modes work.

**gates n** Sets the number of gates per frame in gated (stroboscopic) mode.
Refer to the detailed documentation to understand how the different timing modes work.

**frames n** Sets the number of frames acquired sequentially per cycle (e.g. after each trigger), with the exposure time defined by exptime and the period defined by period (unless in gated mode). The frame index in the output file name will automatically be incremented.
Note that the total number of images will be frames times cycles. Refer to detailed documentation to understand how the different timing modes work.

**cycles n** Sets the number of cycles (e.g. number of triggers). The frame index in the output file name will automatically be incremented.
Note that the total number of images will be by frames times cycles. Refer to the detailed documentation to understand how the different timing modes work.

**probes** Sets the number of probes to accumulate for stroboscopic measurements.
Refer to detailed documentation to understand how the different timing modes work.

**measurements** Sets the number of repetitions of the acquisitions (non real time!). The file index in the file name will be automatically incremented. Refer to detailed documentation to understand how the different timing modes work.

**dr n** Sets the dynamic range n (in bits) of the data for a photon counting detector. For EIGER it can be set to 4, 8, 16 (but the real counter depth will still be limited to 12 bits) or 32 when one wants to activate the internal subframe summing mode.

**flags s** Sets some particular flags for your detector. For MYTHEN s can be *none*, *storeinram* (for buffered readout) or *tot* (for time over threshold). For EIGER, s can be *continous* (for continous readout- still buffer on memories happens), *storeinram* (for buffered readout. Do not use as has no graet advantages), *parallel* for parallel exposure to the next frame and readout of the previous frame, *nonparallel* to decouple sequentially readout and exposure, *safe* (rowclock interleaved).

**help cmd** Returns the help for command cmd.

**lock** Locks (1) or unlocks (0) the detector to this particular control PC. An be unlocked again only from the same PC or by rebooting the detector.

**nmod n** Sets the number of modules for the detector to n for partial readout. Will be replaced by ROI.

## 3.2 Postprocessing commands

**flatfield fname** Sets the flat field file name. File ffdir/fname will be used to calculate the flat field coefficients. *none* to unset flat field corrections.

**ratecorr ns** Sets the deadtime to be used for rate corrections in ns. 0 to unset, -1 to use default dead time for the actual settings.In the case of EIGER, as online data rate correctiosn are applied, then a correction table has to be calculated every time the rate correction $\tau$ is changed, activated, or the subexposure time is changed.

### 3.2.1 Angular conversion

**fineoff deg** Sets the fine offset for the experiment.

**samplex mm** Sets the sample displacement from the center of the diffractometer in the X-ray direction, to improve angular conversion (unused).

**sampley mm** Sets the sample displacement from the center of the diffractometer in the ortogonal direction, to improve angular conversion (unused)

## 3.3 Acquisition

See SLS Detectors Documentation for a detailed description of the acquisition flow.

**positions n p1 p2...pn** Sets the number of positions n and their value.

**startscript s** Sets the script to be executed at the beginning of each measurement. *none* unsets.

**startscriptpar p** Sets the parameter to be passed to the start script

**stopscript s** Sets the script to be executed at the end of each measurement. *none* unsets.

**stopscriptpar p** Sets the parameter to be passed to the stop script.

**scriptbefore s** Sets the script to be executed before each acquisition. *none* unsets.

**scriptbeforepar p** Sets the parameter to be passed to the script before.

**scriptafter s** Sets the script to be executed after each acquisition. *none* unsets.

**scriptafterpar p** Sets the parameter to be passed to the script after.

**headerbefore s** Sets the script to be executed to acquire the header of the acquisition. *none* unsets.

**headerbeforepar p** Sets the parameter to be passed to the header before.

**headerafter s** Sets the script to be executed to append to the header of the acquisition. *none* unsets.

**headerafterpar p** Sets the parameter to be passed to the header after.

**scan0scripts s** Sets the script to execute at scan 0 level. *none* unsets, *threshold, energy, trimbits, position* perform the corresponding scans without need of a custom script.

**scan0par p** Sets a parameter to be passed to the scan 0 level script.

**scan0prec i** Sets the number of decimal digits for the scan0 level parameter in the file name (default is 0).

**scan0steps n s1 s2..sn** Sets the number of scan 0 level steps n and their value.

**scan0range min max step** Sets the minimum, the maximum and the step for the scan 0 level steps (easier to use than scan0steps if equally spaced steps in a range)

**scan1script s** Sets the script to execute at scan 1 level. *none* unsets, *threshold, energy, trimbits, position* perform the corresponding scans without need of a custom script.

**scan1par p** Sets a parameter to be passed to the scan 1 level script.

**scan1prec i** Sets the number of decimal digits for the scan1 level parameter in the file name (default is 0).

**scan1steps n s1 s2...sn** Sets the number of scan 0 level steps n and their value.

**scan1range min max step** Sets the minimum, the maximum and the step for the scan 0 level steps (easier to use than scan0steps if equally spaced steps in a range)

## 3.4 Advanced commands

### 3.4.1 Calibration

This operations should be performed only rarely to configure the detector

**trim:mode fname** Trims the detector according to mode (can be noise, beam, improve, fix) and saves the resulting trimbits to file fname. Take care to set a proper exptime and vthreshold before trimming.

**encallog b** Sets (1) or unsets (0) the logging for energy calibration.

**angcallog b** Sets (1) or unsets (0) the logging for angular calibration.

### 3.4.2 Acquisition

It is normally recommended to use `sls\_detector\_acquire [j-]`, which takes care of everything

**status s** Starts (start) or stops (stop) the detector acquisition.

**online b** Sets the detector in online (1) or offline (0) mode.

**resetctr i** GOTTHARD- ADVANCED- resets counter in detector, restarts acquisition if i=1

**resmat i** EIGER- ADVANCED - resets counter in detector before the following acquisition. Default settings is *resmat 1*. *resmat 0* does not reset the counter bit before the acquisition. Note that in EIGER the counter is always reset after the acquisition.

### 3.4.3 Configuration

Advanced commands to configure the detector system. Should be left to the configuration file

**type s** Sets the types of detector controllers in the system. Can be Mythen, Gotthard, EIGER and multiple controllers should be catenated with a + (e.g. Mythen+Mythen for 2 Mythen controllers).

**d:hostname s** Sets the hostname or IP address for the controller d, where d is the controller index within the detector structure.

**d:extsig:i s** Configures the usage of the external IO signals to synchronize the detectors. s can be: off, gate_in_active_high, gate_in_active_low, trigger_in_rising_edge, trigger_in_falling_edge, ro_trigger_in_rising_edge, ro_trigger_in_falling_edge, gate_out_active_high, gate_out_active_low, trigger_out_rising_edge, trigger_out_falling_edge, ro_trigger_out_rising_edge, ro_trigger_out_falling_edge, sync.
Usually left to the configuration file. Gating, triggering etc. are enabled only by calling the timing command.
Please refer to SLS Detectors FAQ documentation for more detailed information about the usage.

**master i** Sets the master of a multi-controller detector to the controller with index i. -1 removes master. Setting a master is useful only if the controllers are synchronized via hardware using the external IO signals. Usually left to the configuration file. Please refer to SLS Detectors FAQ documentation for more detailed information about the usage.

**sync s** Sets the synchronization mode of the various controller within a detector structure. s acn be none, gating, trigger, complementary. Check that the detectors are correctly connected to avoid freezing of the acquisition. Usually left to the configuration file. Please refer to SLS Detectors FAQ documentation for more detailed information about the usage.

**trimdir s** Obsolete. Same ad settingsdir.

**settingsdir s** Sets the path of the drectory where the trim/settings files are stored. Usually left to the configuration file.

**caldir s** Sets the path of the drectory where the calibration files are stored. Can be the same as settingsdir. Usually left to the configuration file.

**trimen n e1 e2 ...en** Unused. Sets the list of energies for which trimfiles exist.

**port p** Sets the port used by the sockets to control the detector. Do not change! Usually left to the configuration file.

**stopport p** Sets the port used by the sockets to stop/get the status of the detector. Do not change! Usually left to the configuration file.

**add s** Avoid using it. Adds the controller s to the detector structure.

**remove i** Avoid using it. Removes the controller in position i from the detector structure.

**id:i l** Avoid using it. configures the id of the detector structure. i is the detector position in a multi detector system and l is the id of the detector to be added.

**free i** Avoid using it. Frees the shared memory.

**exitserver** Avoid using it. Turns off the communication server on the detector.

### 3.4.4 Receiver - GOTTHARD/EIGER

**detectormac mac** sets the mac of the detector udp interface to mac (if configurable). Should be left to the configuration file.

**rx_tcpport i** sets the communication port between client and receiver. Should be left to the configuration file.

**rx_udpport i** sets the communication port between detector and receiver. Should be left to the configuration file.

**rx_hostname s** sets the hostname (or IP address) of the receiver for the TCP/IP interface with the client.

**rx_udpip ip** sets the IP address of the receiver for the UDP interface with the detector.

**rx_fifodepth v** sets receiver fifo depth to value v. Default for EIGER is 100 frames betweeen listening and writing.

**r_online b** sets the receiver in online (1) or offline (0) mode.

**r_lock b** Locks (1) or unlocks (0) the receiver to this PC.

**receiver s** starts/stops the receiver to listen to detector packets. - can be start or stop

### 3.4.5 Postprocessing

Some advanced commands to configure data postprocessing.

**ffdir dir** Sets the directory where the flat field files are stored. Normally left to the configuration file.

**darkimage fname** GOTTHARD- ADVANCED- Sets fname as dark image file for the detector.

**gainimage fname** GOTTHARD- ADVANCED- Sets fname as gain image file for the detector.

**badchannels fname** Sets the bad channel file to fname. Bad channels will be omitted in the .dat file. *none* to unset. Normally left to the configuration file.

**threaded b** Avoid changing it. Sets if the data are written to disk in parallel with the acquisition (1) or after the acquisition (0). Normally left to the configuration file.

**Angular conversion**

**globaloff deg** Sets the offset of the beamline i.e. angular position of channel 0 when angular encoder at 0. Normally left to the configuration file.

**angconv fname** Sets the file with the coefficients for angular conversion. *none* disables angular conversion. Normally left to the configuration file.

**binsize deg** Sets the size of the angular bins for angular coversion. Normally left to the configuration file.

**angdir i** Sets the angular direction of the detector (1 means channel number in the same direction as the angular encoder, -1 different direction). Normally left to the configuration file.

**d:moveflag i** Related to a single controller d. 1 if the detector modules move with the angular encoder, 0 if they are static (useful for multidetector systems). Normally left to the configuration file.

### 3.4.6 Testing - EIGER specific

Some VERY ADVANCED testing functions implemented for EIGER:

**pulsechip n** sets the chip into test mode with $resmat = 0$ and $externalenable$ =1. Pulses chip by togglying the enable n number of times. The acquire is then done with no pixel matrix reset before the acquisition. If n= $-1$, the chip will be set into normal mode. This is necessary to restore normal chip operations after the test.

**pulse n x y** Pulses pixel at coordinates (x,y) n number of times.

**pulsenmove n x y** Pulses pixel n number of times and moves relatively by x value (x axis) and y value (y axis)

## 3.5 Detector settings

Advanced settings changing the analog or digital performance of the acquisition. Use them only if you are sure of what you are doing!

**vthreshold n** Sets the DAC value of the detector threshold to n.

**vcalibration n** Sets the DAC value of the calibration pulse amplitude to n.

**vtrimbit n** Sets the DAC value defining the trimbits LSB size to n.

**vpreamp n** Sets the DAC value of the preamp feedback to n.

**vshaper1 n** Sets the DAC value of the shaper1 feedback to n.

**vshaper2 n** Sets the DAC value of the shaper2 feedback to n.

**vhighvoltage n** Sets the DAC value of the high voltage to n (in V).

**vapower n** CHIPTEST BOARD ONLY - Sets the DAC value of the analog voltage to n.

**vddpower n** CHIPTEST BOARD ONLY - Sets the DAC value of the analog voltage to n.

**vshpower n** CHIPTEST BOARD ONLY - Sets the comparator power supply in dac units (0-1024).

**viopower n** CHIPTEST BOARD ONLY - Sets the FPGA I/O power supply in dac units (0-1024).

**vref_ds n** Sets vrefds

**vcascn_pb n** Sets vcascn_pb

**vcascp_pb n** Sets vcascp_pb

**vout_cm n** Sets vout_cm

**vcasc_out n** Sets vcasc_out

**vin_cm n** Sets vin_cm

**vref_comp n** Sets vref_comp

**ib_test_c n** Sets ib_test_c

**vsvp n** Sets vsvp DAC to n

**vsvn n** Sets vsvn DAC to n

**vtr n** Sets vtr DAC to n

**vrf n** Sets vrf DAC to n

**vrs n** Sets vrs DAC to n

**vtgstv n** Sets vtgstv DAC to n

**vcmp_ll n** Sets vcmp_ll DAC to n

**vcmp_lr n** Sets vcmp_lr DAC to n

**vcmp_rl n** Sets vcmp_rl DAC to n

**vcmp_rr n** Sets vcmp_rr DAC to n

**vcall n** Sets vcall DAC to n

**rxb_rb n** Sets rxb_rb DAC to n

**rxb_lb n** Sets rxb_rb DAC to n

**vcp n** Sets vcp DAC to n

**vcn n** Sets vcn DAC to n

**vis n** Sets vis DAC to n

**iodelay n** Sets iodelay to n

**reg a d** Write to register of address a the data d

**clkdivider n** Sets the clock divider for the readout. Can be increased for longer cables. For EIGER options are 0 (full speed), 1 (half speed), 2 (quarter speed), and 3 (slow).

**setlength n** Changes the length of the set/reset signals in the acquisition. Never reduce it!

**waitstates n** Sets the wait states for CPU/FPGA communication. Do not change it!

**totdivider n** Sets the tot clock divider.

**totdutycycle n** Sets the tot duty cycle.

**setup s** Loads the setup files to the detector (config, parameters, trimbits etc.).

**trimbits fn** Loads the trimbit files fn.snxxx to the detector

## 3.6 Debug

**digibittest i** only for GOTTHARD. If i=1, the acquisition will return a unique channel identifier, instead of data, if i=0 normal acquisition.

# 4 Retrieving detector parameters

```
sls\_detector\_get [j-][i:]var [arg]
```

is used to retrieve the detector parameters `var`.
For some commands, an additional argument `arg` is needed.

## 4.1 Standard commands

All the commends return two strings, where string1 is the command, string2 is teh actual returned string.

**config fname** Dumps the current configuration of the detector to the file fname.

**parameters fname** Dumps the current acquisition parameters of the detector to the file fname.

**settings** Returns the current settings of the detector. Returns a string

**threshold** For photon counting detectors, returns the detector threshold in eV, -1 if undefined. Returns "threshold value_in_eV". If it fails, the returned threshold is the old set value.

**timing** Returns the acquisition timing mode of the detector. Refer to the detailed documentation to understand how the different timing modes work.

**outdir** Returns the path where the output files are saved to.

**fname** Returns the prefix of the file name for the data output.

**enablefwrite** Returns if data are written to file (1) or not (0).

**exptime** Returns the exposure time of a single acquisition in seconds. Example: "exptime 1.000000000" Refer to detailed documentation to understand how the different timing modes work.

**period** Returns the frames period (in s). Example: "period 1.000000000" Refer to detailed documentation to understand how the different timing modes work.

**delay** Returns the delay after trigger in triggered mode (in s). Refer to detailed documentation to understand how the different timing modes work.

**gates** Returns the number of gates per frame in gated (stroboscopic) mode. Refer to detailed documentation to understand how the different timing modes work.

**frames** Returns the number of frames acquired sequentially per cycle (e.g. after each trigger), with the exposure time defined by exptime and the period defined by period (unless in gated mode). Returned as a string to be interpreted as an integer "frames integer" Note that the total number of images is frames times cycles. Refer to detailed documentation to understand how the different timing modes work.

**cycles n** Returns the number of cycles (e.g. number of triggers). Returned as atring to be interpreted as an integer "cycles integer" Note that the total number of images is frames times cycles. Refer to detailed documentation to understand how the different timing modes work.

**probes** Returns the number of probes to accumulate for stroboscopic measurements. Refer to detailed documentation to understand how the different timing modes work.

**measurements** Returns the number of repetitions of the acquisitions (non real time!). Refer to detailed documentation to understand how the different timing modes work.

**dr** Returns the dynamic range n (in bits) of the data for a photon counting detector. Returns a string that should be interpreted as an integer.

**flags s** Returns the flags set for your detector.

**help cmd** Returns the help for command cmd.

**lock** Returns if the detector is locked to a single PC.

**lastclient** Returns the last client which has connected to the detector.

**nmod n** Returns the number of modules which are read out. Will be replaced by ROI.

**maxmod** Returns the maximum number of modules (size) of the detector. Will be replaced by size.

## 4.2 Postprocessing commands

**flatfield** Returns the flat field file name.

**ratecorr** Returns the dead time used for rate corrections.

### 4.2.1 Angular conversion

**fineoff** Returns the fine offset used to convert channel number to angles

**samplex** Returns the sample displacement from the center of the diffractometerin the X-ray direction, to improve angular conversion (unused).

**sampley** Returns the sample displacement from the center of the diffractometer in the ortogonal direction, to improve angular conversion (unused)

## 4.3 Acquisition

See SLS Detectors Documentation for a detailed description of the acquisition flow.

**positions** Returns the number of positions n and their value.

**startscript** Returns the script to be executed at the beginning of each measurement.

**startscriptpar** Returns the parameter to be passed to the start script

**stopscript** Returns the script to be executed at the end of each measurement.

**stopscriptpar** Returns the parameter to be passed to the stop script.

**scriptbefore** Returns the script to be executed before each acquisition.

**scriptbeforepar** Returns the parameter to be passed to the script before.

**scriptafter** Returns the script to be executed after each acquisition.

**scriptafterpar** Returns the parameter to be passed to the script after.

**headerbefore** Returns the script to be executed to acquire the header of the acquisition.

**headerbeforepar** Returns the parameter to be passed to the header before.

**headerafter** Returns the script to be executed to append to the header of the acquisition.

**headerafterpar** Returns the parameter to be passed to the header after.

**scan0scripts** Returns the script to execute at scan 0 level.

**scan0par** Returns a parameter to be passed to the scan 0 level script.

**scan0prec** Returns the number of decimal digits for the scan0 level parameter in the file name (default is 0).

**scan0steps** Returns the number of scan 0 level steps n and their value.

**scan0range** Same as scan0steps.

**scan1script** Returns the script to execute at scan 1 level.

**scan1par** Returns a parameter to be passed to the scan 1 level script.

**scan1prec** Returns the number of decimal digits for the scan1 level parameter in the file name (default is 0).

**scan1steps** Returns the number of scan 0 level steps n and their value.

**scan1range** Same as scan1steps.

## 4.4 Debug

Commands to be used to retrieve information about the detector version or perform tests.

### 4.4.1 Version

**moduleversion[:i** ] Returns the version of the module firmware.

**detectornumber** Returns the serial number of the module (normally the MAC address).

**modulenumber[:i** ] Returns the serial number of the module i.

**detectorversion** Returns the version of the controller firmware.

**softwareversion** Returns the version of the software running on the detector.

**thisversion** Returns the version of the control software which is being used.

**detectorsvnversion** Returns the SVN version of the software on the detector.

### 4.4.2 Tests

**digitest[:i** ] Makes a digital test of module i. Afterwards the detector must be reconfigured for the acquisition (settings, threshold, exptime, dr, frames etc.). Returns 0 if succeeded, otherwise an error mask.

**bustest** Makes a digital test of the communication between CPU and FPGA. Returns 0 if succeeded, otherwise the number of errors.

## 4.5 Advanced commands

### 4.5.1 Calibration

This operations should be performed only rarely to configure the detector

**encallog** returns whether the logging for energy calibration is enabled.

**angcallog** returns whether the logging for angular calibration is enabled.

### 4.5.2 Acquisition commands

It is normally recommended to use `sls\_detector\_acquire [j-]`, which takes care of everything

**acquire** Same as `sls\_detector\_acquire`

**data** Gets, saves and processes all data stored on the detector, if any.

**frame** Gets, saves and processes one frame stored on the detector, if any in a Firt-In/First-Out mode.

**status** Returns the detector status - can be: running, error, transmitting, finished, waiting or idle

**online** Returns whether the detector is in online or offline mode.

**checkonline** Returns whether the detector is in online or offline mode.

**readctr i fname** GOTTHARD related - reads counter in detector to file fname, restarts acquisition if i=1

**exptimel** Returns the exposure time left for the current frame.

**periodl** Returns the period left for the current frame.

**delayl** Returns the delay after trigger left for the current frame.

**gatesl** Returns the number of gates left for the current frame.

**framesl** Returns the number of frames left for the current cycle.

**cyclesl** Returns the number of cycles left for the current acquisition.

**now** Returns the current timestamp of the detector clock.

**timestamp** Returns the timestamp of the acquisitions in a First-In/First-Out mode i.e. every time it is called it returns the timestamp of the first acquisition start of readout. The FIFO is reset everytime the acquisition is started.

### 4.5.3 Configuration

Advanced commands to configure the detector system. Should be left to the configuration file

**type** Returns the types of detector controllers in the system.

**hostname** Returns the hostnames or IP addresses for the detector

**d:extsig:i** Returns the usage of the external IO signal i of the controller d.

**master** Returns the master of the acquisition in a multicontroller detector. -1 is none.

**sync** Returns the synchronization mode of the various controller within a detector structure.

**trimdir** Same ad settingsdir.

**settingsdir** Returns the path of the directory where the trim/settings files are stored.

**caldir** Returns the path of the directory where the calibration files are stored.

**trimen n e1 e2 ...en** Unused. Returns the list of energies for which trimfiles exist.

**port** Returns the port used by the sockets to control the detector.

**stopport** Returns the port used by the sockets to stop/get the status of the detector.

**id[:i** ] returns the id of the detector structure. i is the detector position in a multi detector system

**free** Avoid using it. Frees the shared memory.

Settable communication parameters:

**txndelay_left** EIGER advanced: Set transmission delay of sending the left port frame

**txndelay_right** EIGER advanced: Set transmission delay of sending the right port frame

**txndelay_frame** EIGER advanced: Set transmission delay of sending the entire frame In addition to left and right. This value has to be greater than the maximum of the transmission delays of each port.

#### 4.5.4 Receiver - GOTTHARD only

**detectormac** returns the mac of the detector udp interface to mac (if configurable). Should be left to the configuration file.

**rx_tcpport** returns the communication port between client and receiver. Should be left to the configuration file.

**rx_udpport** returns the communication port between detector and receiver. Should be left to the configuration file.

**rx_hostname** returns the hostname (or IP address) of the receiver for the TCP/IP interface with the client.

**rx_udpip** returns the IP address of the receiver for the UDP interface with the detector.

**r_online b** Returns whether the receiver in online (1) or offline (0) mode.

**r_checkonline** Returns whether the receiver in online (1) or offline (0) mode.

**framescaught** Returns the number of frames received. Returns: "framescaught n"

**resetframescaught n** Sets the number of frames received to 1

**frameindex** Returns the index of the last frame received.

**r_lock** Returns whether the receiver is locked (1) or unlocked (0).

**r_lastclient** Returns the IP of the last client which connected to the receiver.

#### 4.5.5 Postprocessing

Some advanced commands to configure data postprocessing.

**ffdir** Returns the directory where the flat field files are stored.

**darkimage fname** GOTTHARD- ADVANCED- Returns the dark image file for the detector.

**gainimage fname** GOTTHARD- ADVANCED- Returns gain image file for the detector.

**badchannels fname** Returns bad channel file to fname.

**threaded b** Returns whether the data are written to disk in parallel with the acquisition (1) or after the acquisition (0).

**Angular conversion**

**globaloff** Returns the offset of the beamline i.e. angular position of channel 0 when angular encoder at 0.

**angconv** Returns the file used for the coefficients for angular conversion.

**binsize** Returns the size of the angular bins for angular conversion.

**angdir** Returns the angular direction of the detector (1 means channel number in the same direction as the angular encoder, -1 different direction).

**d:moveflag** Related to a single controller d. Returns 1 if the detector modules move with the angular encoder, 0 if they are static (useful for multidetector systems).

## 4.6 Detector settings

Advanced settings changing the analog or digital performance of the acquisition. Use them only if you are sure of what you are doing!

**vthreshold** Returns the DAC value of the detector threshold to n.

**vcalibration** Returns the DAC value of the calibration pulse amplitude to n.

**vtrimbit** Returns the DAC value defining the trimbits LSB size to n.

**vpreamp** Returns the DAC value of the preamp feedback to n.

**vshaper1** Returns the DAC value of the shaper1 feedback to n.

**vshaper2** Returns the DAC value of the shaper2 feedback to n.

**vhighvoltage** Returns the DAC value of the high voltage to n.

**vapower** CHIPTEST BOARD ONLY - Returns the DAC value of the analog voltage to n.

**vddpower** CHIPTEST BOARD ONLY - Returns the DAC value of the analog voltage to n.

**vshpower** CHIPTEST BOARD ONLY - Returns the comparator power supply in dac units (0-1024).

**viopower** CHIPTEST BOARD ONLY - Returns the FPGA I/O power supply in dac units (0-1024).

**vref_ds** Returns vrefds

**vcascn_pb** Returns vcascn_pb

**vcascp_pb** Returns vcascp_pb

**vout_cm** Returns vout_cm

**vcasc_out** Returns vcasc_out

**vin_cm** Returns vin_cm

**vref_comp** Returns vref_comp

**ib_test_c** Returns ib_test_c

**vsvp** Returns vsvp

**vsvn** Returns vsvn

**vtr** Returns vtr trim strength (EIGER)

**vrf** Returns vrf preamp gain (EIGER)

**vrs** Returns vrs shaper gain (EIGER)

**vtgstv** Returns vtgstv (EIGER)

**vcmp_ll** Returns vcmp_ll (EIGER) leftmost chip theshold

**vcmp_lr** Returns vcmp_lr (EIGER) second to leftmost chip theshold

**vcmp_rl** Returns vcmp_rl (EIGER) second to rightmost chip theshold

**vcmp_rr** Returns vcmp_rr (EIGER) rightmost chip theshold

**vcall** Returns vcall calibration stength (EIGER)

**rxb_rb** Returns rxb_rb rightmost chip value to decode 0-1 in the readout

**rxb_lb** Returns rxb_lb leftmost chip value to decode 0-1 in the readout

**vcp** Returns vcp cascode p value (EIGER)

**vcn** Returns vcn cascode n value (EIGER)

**vis** Returns vis shaper current (EIGER)

**iodelay** Returns iodelay

**temp_adc** Returns the temperature of the ADCs

**temp_fpga** Returns the temperature of the FPGA.

**temp_fpgaext** Returns the temperature close to the fpga (EIGER).

**temp_10ge** Returns the temperature close to the 10GE (EIGER).

**temp_dcdc** Returns the temperature close to the dc dc converter (EIGER).

**temp_sodl** Returns the temperature close to the left so-dimm memory (EIGER).

**temp_sodr** Returns the temperature close to the right so-dimm memory (EIGER).

**temp_fpgafl** Returns the temperature of the left front end board fpga (EIGER).

**temp_fpgafr** Returns the temperature of the right front end board fpga (EIGER).

**reg a** Write to register of address a the data d

**clkdivider** Returns the clock divider for the readout.

**setlength** Returns the length of the set/reset signals in the acquisition.

**waitstates** Returns the wait states for CPU/FPGA communication.

**totdivider** Returns the tot clock divider.

**totdutycycle** Returns the tot duty cycle.

**setup** Dumps all settings to file (config, parameters, trimbits etc.).

**trimbits fn** Dumps the trimbits to the file files fn.snxxx

## 5 Usage

### 5.1 Mandatory setup

First, your detector should always be configured for each PC that you might want to use for controlling the detector. To do that:

```
sls_detector_put config mydetector.config
```

Refer to sample configuration files to produce the appropriate one for your detector.

One can configure all the detector settings in a parameter file `setup.det`, which is loaded by doing:

```
sls_detector_put parameters setup.det
```

In the case of EIGER, the parameter file (`setup.det` needs to setup the proper bias voltage of the sensor, i.e. needs to contain the line `vhighvoltage 150`.

## 5.2 Standard acquisition

You will then need to setup the detector threshold and settings, the exposure time, the number of real time frames and eventually how many real time frames should be acquired:

```
sls_detector_put settings standard
sls_detector_put threshold 6000
sls_detector_put exptime 1.
sls_detector_put frames 10
```

In this case 10 consecutive 1s frames will be acquired.

You need to setup where the files will be written to

```
sls_detector_put outdir /scratch
sls_detector_put fname run
sls_detector_put index 0
```

this way your files will all be named /scratch/run_fj_i.dat where j goes between 0 and 9 and is relative to the frame number, i starts from 0 and is automatically incremented. The next acquisition it will be 1.

To acquire simply type

```
sls_detector_acquire
```

You can poll the detector status using

```
sls_detector_get status
```

## 5.3 Data processing

Flat field and rate corrections can be applied directly by simply selecting:

```
sls_detector_put flatield myflatfield.raw
sls_detector_put ratecorr -1
```