Reference Manual

Generated by Doxygen 1.6.1

Mon Mar 12 14:18:05 2018

Contents

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.

The commands are sudivided into different pages depending on their functionalities:

- Acquisition: commands to start/stop the acquisition and retrieve data
- Configuration: commands to configure the detector
- Timing: commands to configure the detector timing
- Data postprocessing: commands to process the data - mainly for MYTHEN except for rate corrections.

- Settings: commands to define detector settings/threshold.
- Output: commands to define output file destination and format
- Actions: commands to define scripts to be executed during the acquisition flow
- Network: commands to setup the network between client, detector and receiver
- Receiver: commands to configure the receiver
- Chiptest board: commands specific for the new chiptest board as pattern generator
- Developer: commands to be used only for software debugging. Avoid using them!

2 Developer

Commands to be used only for software debugging. Avoid using them!

- **test** returns an error
- **help** Returns a list of possible commands.
- **exitserver** Shuts down all the detector servers. Don't use it!!!!
- **exitreceiver** Shuts down all the receivers. Don't use it!!!!
- **flippeddatay [i]** enables/disables data being flipped across y axis. 1 enables, 0 disables. Not implemented.
- **digitest [i]** will perform test which will plot the unique channel identifier, instead of data. Only get!
- **bustest** performs test of the bus interface between FPGA and embedded Linux system. Can last up to a few minutes. Cannot set! Used for Mythen only. Only get!
- **digibittest:[i]** performs digital test of the module i. Returns 0 if succeeded, otherwise error mask. Only put!
- **reg [addr] [val]** ??? writes to an register `addr` with `value` in hexadecimal format.
- **adcreg [addr] [val]** ??? writes to an adc register `addr` with `value` in hexadecimal format. Only put!
- **setbit** ??? Only put!

- **clearbit** ??? Only put!
- **getbit** ??? Only get!
- **r_compression [i]** sets/gets compression in receiver. 1 sets, 0 unsets. Not implemented.

3 Acquition commands

Commands to control the acquisition

- **acquire** blocking acquisition (like calling sls_detector_acquire). Starts receiver and detector, writes and processes the data, stops detector. Only get! `Returns` (string)`"acquire unsuccessful"` if fails, else "" for MYTHEN, `"Acquired (int)"` for others, where int is number of frames caught.
- **busy i** sets/gets acquiring flag. `1` the acquisition is active, `0` otherwise. Acquire command will set this flag to 1 at the beginning and to 0 at the end. Use this to clear flag if acquisition terminated unexpectedly. `Returns` (int)
- **status [s]** starts or stops acquisition in detector in non blocking mode. When using stop acquisition and if acquisition is done, it will restream the stop packet from receiver (if data streaming in receiver is on). `s:` [`start`, `stop`]. `Returns` the detector status: [`running`, `error`, `transmitting`, `finished`, `waiting`, `idle`]. `Returns` (string)
- **data** gets all data from the detector (if any) processes them and writes them to file according to the preferences already setup (MYTHEN only). Only get!
- **frame** gets a single frame from the detector (if any) processes it and writes it to file according to the preferences already setup (MYTHEN only). Only get!
- **readctr** Reads the counters from the detector memory (analog detector returning values translated into number of photons - only GOTTHARD). Cannot put.
- **resetctr i** Resets counter in detector, restarts acquisition if i=1(analog detector returning values translated into number of photons - only GOTTHARD). Cannot put.
- **resmat i** sets/resets counter bit in detector.gets the counter bit in detector ????

4 Configuration commands

Commands to configure the detector. these commands are often left to the configuration file.

- Data Structure: commands to configure detector data structure
- Status: commands to configure detector status
- Data Size: commands to configure detector data size
- Flags: commands to configure detector flags
- Chip: commands to configure chip of the detector
- Versions: commands to check version of each subsytem
- Speed: commands to configure speed of detector
- Detector Parameters: commands to configure/retrieve configuration of detector

4.1 Data Structure

commands to configure detector data structure

- **free** Free shared memory on the control PC
- **add** Adds a detector at the end of the multi-detector structure. `put` argument is the hostname or IP adress. Returns the chained list of detector hostnames.
- **remove i** Removes controller `i` from the multi-detector structure. Can be used for partial readout of the detector.
- **type** Sets/gets detector type. `Returns` (string). Normally not used. Using hostname is enough.
- **hostname** `put` adds the hostname (ot IP adress) at the end of the multi-detector structure. If used for a single controlled (i:) replaces the current hostname. Returns the list of the hostnames of the multi-detector structure. `Returns` (string)
- **id[:i]** Returns the id of the detector structure. i is the detector position in a multi detector system. If used a `put`, 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.
- **master i** `put` sets the position of the master of the acquisition (-1 if none). Returns the position of the master of the detector structure (-1 if none).
- **sync** Sets/gets the synchronization mode of the detectors in the multi-detector structure. Can be: `none`, `gating`, `trigger`, `complementary`. Mainly used by MYTHEN/GOTTHARD.

4.2 Status

commands to configure detector status

- **online [i]** sets the detector in online (1) or offline (0) mode. `Returns` (int)
- **checkonline** returns the hostnames of all detectors without connecting to them. `Returns` (string) "All online" or "[list of offline hostnames] : Not online".
- **activate** Activates/Deactivates the detector. Deactivated detector does not send data. Used for EIGER only. `Returns` (int)

4.3 Data Size

commands to configure detector data size

- **nmod [i]** sets/gets the number of modules of the detector. Used for MYTHEN only. `Returns` (int)
- **maxmod** Gets the maximum number of modules of the detector. Used for MYTHEN only. Cannot put! `Returns` (int)
- **dr [i]** sets/gets the dynamic range of detector. Mythen [4,8,16,24]. Eiger [4,8,16,32]. Others cannot put! `Returns` (int)
- **roi [i] [xmin] [xmax] [ymin] [ymax]** sets region of interest of the detector, where i is number of rois;i=0 to clear rois. Used for GOTTHARD only. `Returns` (int)
- **detsizechan [xmax] [ymax]** sets the maximum number of channels in each dimension for complete detector set; -1 is no limit. Use for multi-detector system as first command in config file. `Returns` ("int int")
- **roimask [i]** ?? `Returns` (int) in hexadecimal
- **flippeddatax [i]** enables/disables data being flipped across x axis. 1 enables, 0 disables. Used for EIGER only. 1 for bottom half-module, 0 for top-half module. `Returns` (int)
- **tengiga [i]** enables/disables 10GbE in system (detector & receiver). 1 enabled 10GbE, 0 enables 1GbE. Used in EIGER only. `Returns` (int)

4.4 Flags

commands to configure detector flags

- **flags [flag]** sets/gets the readout flags to mode. Options: none, storeinram, tot, continous, parallel, nonparallel, safe, digital, analog_digital, unknown. Used for MYTHEN and EIGER only. `Returns` (string). put takes one string and `returns` concatenation of all active flags separated by spaces.

- **extsig:[i] [flag]** sets/gets the mode of the external signal i. Options: `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`.

  Used in MYTHEN, GOTTHARD, PROPIX only. `Returns` (string)

- **programfpga [file]** programs the FPGA with file f (with .pof extension). Used for JUNGFRAU, MOENCH only. Only put! `Returns` ("successful", "unsuccessful")

- **resetfpga [f]** resets FPGA, where f can be any value. Used for JUNGFRAU only. Only put! `Returns` ("successful", "unsuccessful")

4.5 Chip

commands to configure chip of the detector

- **powerchip [i]** Powers on/off the chip. 1 powers on, 0 powers off. Can also get the power status. Used for JUNGFRAU only. `Returns` (int)

- **led [i]** sets/gets the led status. 1 on, 0 off. Used for MOENCH only ?? `Returns` (int)

- **auto_comp_disable i** Currently not implemented. this mode disables the on-chip gain switching comparator automatically after 93.75% of exposure time (only for longer than 100us). 1 enables mode, 0 disables mode. By default, mode is disabled (comparator is enabled throughout). (JUNGFRAU only). `Returns` (int)

- **pulse [n] [x] [y]** pulses pixel at coordinates (x,y) n number of times. Used in EIGER only. Only put! `Returns` ("successful", "unsuccessful")

- **pulsenmove [n] [x] [y]** pulses pixel n number of times and moves relatively by x value (x axis) and y value(y axis). Used in EIGER only. Only put! `Returns` ("successful", "unsuccessful")

- **pulsechip [n]**pulses chip n number of times, while n=-1 will reset it to normal mode. Used in EIGER only. Only put! `Returns` ("successful", "unsuccessful")

4.6 Versions

Commands to check versions of each subsystem

- **moduleversion:[i]** Gets the firmware version of module i. Used for MYTHEN only. Only get! `Returns` (long int) in hexadecimal or "undefined module number"
- **detectornumber** Gets the serial number or MAC of detector. Only get! `Returns` (long int) in hexadecimal
- **modulenumber:[i]** Gets the serial number of module i. Used for MYTHEN only. Only get! `Returns` (long int) in hexadecimal or "undefined module number"
- **detectorversion** Gets the firmware version of detector. Only get! `Returns` (long int) in hexadecimal
- **softwareversion** Gets the software version of detector server. Only get! `Returns` (long int) in hexadecimal
- **thisversion** Gets the software version of this client software. Only get! `Returns` (long int) in hexadecimal
- **receiverversion** Gets the software version of receiver. Only get! `Returns` (long int) in hexadecimal
- **framesl** gets number of frames left. Used in MYTHEN, GOTTHARD only. Only get! `Returns` (double with 9 decimal digits)

4.7 Speed

commands to configure speed of detector

- **clkdivider [i]** sets/gets the readout clock divider. EIGER, JUNGFRAU [0(fast speed), 1(half speed), 2(quarter speed)]. MYTHEN[???]. `Returns` (int)
- **setlength [i]** sets/gets length of set/reset signals (in clock cycles). Used in MYTHEN only. `Returns` (int)
- **waitstates [i]** sets/gets waitstates of the bus interface (in clock cycles). Used in MYTHEN only. `Returns` (int)
- **totdivider [i]** sets/gets clock divider in tot mode. Used in MYTHEN only. `Returns` (int)

- **totdutycycle [i]** sets/gets duty cycle of the tot clock. Used in MYTHEN only. `Returns` (int)

- **phasestep [i]** Only put for gotthard. Moves the phase of the ADC clock.`Returns` (int)

- **oversampling [i]** Sets/gets the number of adcsamples per clock. For the new chiptestboard.`Returns` (int)

- **adcclk [i]** sets/gets the ADC clock frequency in MHz. For the new chiptestboard!`Returns` (int)

- **adcphase [i]** Sets/gets the ADC clock frequency in MHz. For the new chiptestboard!`Returns` (int)

- **adcpipeline [i]** Sets/gets the pipeline of the ADC. For the new chiptestbaord!`Returns` (int)

- **dbitclk [i]** Sets/gets the clock frequency of the latching of the digital bits in MHz. For the new chiptestboard!`Returns` (int)

- **dbitphase [i]** Sets/gets the phase of the clock for latching of the digital bits. For the new chiptestboard!?`Returns` (int)

- **dbitpipeline [i]** Sets/gets the pipeline of the latching of the digital bits. For the new chiptestbaord!`Returns` (int)

4.8 Detector Parameters

commands to configure/retrieve configuration of detector

- **config [fname]** sets/saves detector/receiver to configuration contained in fname. Same as executing sls_detector_put for every line. Normally a one time operation. `Returns` (string) fname

- **rx_printconfig** prints the receiver configuration. Only get! `Returns` (string)

- **parameters [fname]** sets/saves detector parameters contained in fname. Normally once per different measurement. `Returns` (string) fname

- **setup [fname]** sets/saves detector complete setup contained in fname (extensions automatically generated), including trimfiles, ff coefficients etc. `Returns` (string) fname

5 Timing commands

Commands to setup the timing

- **timing [mode]** sets/gets synchronization mode of the detector. Mode: auto, trigger, ro_trigger, gating, triggered_gating (string)
- **exptime [i]** sets/gets exposure time in s. `Returns` (double with 9 decimal digits)
- **subexptime [i]** sets/gets sub exposure time in s. Used in EIGER only in 32 bit mode. `Returns` (double with 9 decimal digits)
- **period [i]** sets/gets frame period in s. `Returns` (double with 9 decimal digits)
- **delay [i]** sets/gets delay in s. Used in MYTHEN, GOTTHARD only. `Returns` (double with 9 decimal digits)
- **gates [i]** sets/gets number of gates. Used in MYTHEN, GOTTHARD only. `Returns` (long long int)
- **frames [i]** sets/gets number of frames. If `timing` is not `auto`, then it is the number of frames per cycle/trigger. `Returns` (long long int)
- **cycles [i]** sets/gets number of triggers. Timing mode should be set appropriately. `Returns` (long long int)
- **probes [i]** sets/gets number of probes to accumulate. When setting, max 3! cycles should be set to 1, frames to the number of pump-probe events. Used in MYTHEN only. `Returns` (long long int)
- **measurements [i]** sets/gets number of measurements. `Returns` (long long int)
- **samples [i]** sets/gets number of samples expected from the jctb. Used in CHIP TEST BOARD only. `Returns` (long long int)
- **exptimel** gets exposure time left. Used in MYTHEN, GOTTHARD only. Only get! `Returns` (double with 9 decimal digits)
- **periodl** gets frame period left. Used in MYTHEN, GOTTHARD only. Only get! `Returns` (double with 9 decimal digits)
- **delayl** gets delay left. Used in MYTHEN, GOTTHARD only. Only get! `Returns` (double with 9 decimal digits)
- **gatesl** gets number of gates left. Used in MYTHEN, GOTTHARD only. Only get! `Returns` (double with 9 decimal digits)

- **cyclesl** gets number of cylces left. Used in MYTHEN, GOTTHARD only. Only get! `Returns` (double with 9 decimal digits)
- **probesl** gets number of probes left. Used in MYTHEN, GOTTHARD only. Only get! `Returns` (double with 9 decimal digits)
- **now** Actual time of the detector. Only get!
- **timestamp** Last frame timestamp for MYTHEN. Only get!
- **nframes** ??? Only get!

6 Data processing commands

Commands to setup the data processing (mainly MYTHEN related)

- **flatfield [fn]** `put` sets flatfield file to `fn` (relative to `ffdir`). returns the flatfield file name relative to `ffdir` (string). If
- **ffdir [d]** Sets/gets the directory in which the flat field file is located. `Returns` (string) ffdir
- **ratecorr [ns]** Returns the dead time used for rate correections in ns (int). `put` sets the deadtime correction constant in ns, -1 will set it to default tau of settings (0 unset). `Returns` (double with 9 decimal digit precision)
- **badchannels [fn]** `put` sets the badchannels file to `fn` . returns the bad channels file name. If
- **angconv [fn]** `put` sets the angular conversion file to `fn` . returns the angular conversion file name. If
- **globaloff [f]** Sets/gets the beamline angular global offset (float).
- **fineoff [f]** Sets/gets the angular fine offset of the measurement (float).
- **binsize [f]** Sets/gets the bin size used for the angular conversion (float).
- **angdir [i]** Sets/gets the angular direction. 1 means increasing channels number as increasing angle, -1 increasing channel number decreasing angle.
- **moveflag [i]** Sets/gets the flag for physically moving the detector during the acquisition of several positions. 1 sets (moves), 0 unsets.

- **samplex [f]** Sets/gets the sample displacement in th direction parallel to the beam in um. Unused!
- **sampley [f]** Sets/gets the sample displacement in th direction orthogonal to the beam in um. Unused!
- **threaded [i]** Sets/gets the data processing threaded flag. 1 is threaded, 0 unthreaded.
- **darkimage fn** Loads the dark image to the detector from file fn (pedestal image). Cannot get.
- **gainimage fn** Loads the gain image to the detector from file fn (gain map for translation into number of photons of an analog detector). Cannot get.

7 Detector settings commands

Commands to setup the settings of the detector

- Settings, trim & cal Directories: commands to setup settings/trim/cal directories
- Settings and Threshold: commands to configure settings and threshold of detector
- DACs: commands to configure DACs of detector
- ADCs: commands to readout ADCs of detector
- Temp Control: commands to monitor and handle temperature overshoot (only JUNGFRAU)

7.1 Settings, trim & cal Directories

commands to setup settings/trim/cal directories

- **settingsdir [dir]** Sets/gets the directory where the settings files are located. `Returns` (string) dir
- **trimdir [dir]** obsolete `settingsdir`. `Returns` (string) dir
- **caldir [dir]** Sets/gets the directory where the calibration files are located. `Returns` (string) dir
- **trimen [n e0 e1...e(n-1)]** Sets/gets the number of energies n at which the detector has default trim file and their values in eV (int). `Returns` (int int...) n e0 e1...e(n-1)

7.2 Settings and Threshold

commands to configure settings and threshold of detector

- **settings [s]** sets/gets the settings of the detector. Options: `standard`, `fast`, `highgain`, `dynamicgain`, `lowgain`, `mediumgain`, `veryhighgain`, `lownoise`, `dynamichg0`, `fixgain1`, `fixgain2`, `forceswitchg1`, `forceswitchg2`.

  In Eiger, only sets in client shared memory. Use `threshold` or `thresholdnotb` to pass to detector. Gets from detector. `Returns` (string) s

- **threshold [eV] [sett]** sets/gets the detector threshold in eV. sett is optional and if provided also sets the settings. Use this for Eiger instead of `settings`. `Returns` (int)

- **thresholdnotb [eV] [sett]** sets/gets the detector threshold in eV without loading trimbits. sett is optional and if provided also sets the settings. Use this for Eiger instead of `settings`. `Returns` (int)

- **trimbits [fname]** loads/stores the trimbits to/from the detector. If no extension is specified, the serial number of each module will be attached. `Returns` (string) fname

- **trim:[mode] [fname]** trims the detector according to mode and saves resulting trimbits to file. Mode: noise, beam, improve, fix. Used in MYTHEN only. Only put! `Returns` ("done")

- **trimval [i]** sets all trimbits to i. Used in EIGER only. `Returns` (int)

- **pedestal [i]** starts acquisition for i frames, calculates pedestal and writes back to fpga. Used in GOTTHARD only. Only put! `Returns` (int)

7.3 DACs

commands to configure DACs of detector

- **vthreshold [i] [mv]** Sets/gets detector threshold voltage for single photon counters. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vcalibration [i] [mv]** Sets/gets the voltage of the calibration pulses. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vtrimbit [i] [mv]** Sets/gets the voltage to set the width of the trimbits. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vpreamp [i] [mv]** Sets/gets the voltage to define the preamplifier feedback resistance. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vhaper1 [i] [mv]** Sets/gets the voltage to define the feedback resistance of the first shaper. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vshaper2 [i] [mv]** Sets/gets the voltage to define the feedback resistance of the second shaper. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vhighvoltage [i]** Sets/gets the high voltage to the sensor in V. `Returns` (int ["mV"]).

- **vapower [i]** Sets/gets the analog power supply for the old chiptest board in DAC units. `Returns` (int ["mV"])

- **vddpower [i]** Sets/gets the digital power supply for the old chiptest board in DAC units. `Returns` (int ["mV"])

- **vshpower [i]** Sets/gets the comparator power supply for the old chiptest board in DAC units. `Returns` (int ["mV"])

- **viopower [i]** Sets/gets the power supply of the FPGA I/Os for the old chiptest board in DAC units. `Returns` (int ["mV"])

- **vrefds [i] [mv]** Sets/gets vrefds. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vcascn_pb [i] [mv]** Sets/gets vcascn_pb. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vcasc_pb [i] [mv]** Sets/gets vcasc_pb. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vout_cm [i] [mv]** Sets/gets vout_cm. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vcasc_out [i] [mv]** Sets/gets vcasc_out. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vin_com [i] [mv]** Sets/gets vin_com. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vref_comp [i] [mv]** Sets/gets vref_comp. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **ib_test_c [i] [mv]** Sets/gets ib_test_c. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **dac[0..7] [i] [mv]** Sets/gets dac[0..7] for MOENCH02. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vsvp [i] [mv]** Sets/gets vsvp. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vsvn [i] [mv]** Sets/gets vsvn. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vtr [i] [mv]** Sets/gets vtr. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vrf [i] [mv]** Sets/gets vrf. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vrs [i] [mv]** Sets/gets vrs. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vtgstv [i] [mv]** Sets/gets vtgstv. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vcmp_ll [i] [mv]** Sets/gets vcmp_ll. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vcmp_lr [i] [mv]** Sets/gets vcmp_lr. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vcal_l [i] [mv]** Sets/gets vcal_l. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vcomp_rl [i] [mv]** Sets/gets vcomp_rl. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vcomp_rr [i] [mv]** Sets/gets vcomp_rr. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **rxb_rb [i] [mv]** Sets/gets rxb_rb. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **rxb_lb [i] [mv]** Sets/gets rxb_lb. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vcp [i] [mv]** Sets/gets vcp. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vcn [i] [mv]** Sets/gets vcn. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **vis [i] [mv]** Sets/gets vis. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **iodelay [i] [mv]** Sets/gets iodelay. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **dac:j [i] [mv]** Sets/gets value for DAC number j for the new chiptestboard. Normally in DAC units unless `mv` is specified at the end of the command line. `Returns` (int ["mV"])

- **adcvpp [i]** Sets/gets the Vpp of the ADC 0 -> 1V ; 1 -> 1.14V ; 2 -> 1.33V ; 3 -> 1.6V ; 4 -> 2V . `Returns` (int ["mV"])

- **v_a [i] mv** Sets/gets value for Va on the new chiptest board. Must be in mV. `Returns` (int ["mV"])

- **v_b [i] mv** Sets/gets value for Vb on the new chiptest board. Must be in mV. `Returns` (int ["mV"])

- **v_c [i] mv** Sets/gets value for Vc on the new chiptest board. Must be in mV. `Returns` (int ["mV"])

- **v_d [i] mv** Sets/gets value for Vd on the new chiptest board. Must be in mV. `Returns` (int ["mV"])

- **v_io [i] mv** Sets/gets value for Vio on the new chiptest board. Must be in mV. `Returns` (int ["mV"])

- **v_chip [i] mv** Sets/gets value for Vchip on the new chiptest board. Must be in mV. `Returns` (int ["mV"]). Normally don't use it!

- **v_limit [i] mv** Sets/gets a soft limit for the power supplies and the DACs on the new chiptest board. Must be in mV. `Returns` (int ["mV"])

7.4 ADCs

commands to readout ADCs of detector

- **temp_adc** Gets the ADC temperature. `Returns` `EIGER`,JUNGFRAU(double"°C") Others (int"°C")

- **temp_fpga** Gets the FPGA temperature. `Returns` `EIGER`,JUNGFRAU(double"°C") Others (int"°C")

- **temp_fpgaext** Gets the external FPGA temperature. Used in EIGER only. `Returns EIGER`(double"°C")
- **temp_10ge** Gets the 10Gbe temperature. Used in EIGER only. `Returns EIGER`(double"°C")
- **temp_dcdc** Gets the temperature of the DC/DC converter. Used in EIGER only. `Returns EIGER`(double"°C")
- **temp_sodl** Gets the temperature of the left so-dimm memory . Used in EIGER only. `Returns EIGER`(double"°C")
- **temp_sodr** Gets the temperature of the right so-dimm memory. Used in EIGER only. `Returns EIGER`(double"°C")
- **adc:j** Gets the values of the slow ADC number j for the new chiptest board. `Returns` (int"°C")
- **temp_fpgal** Gets the temperature of the left frontend FPGA. Used in EIGER only. `Returns EIGER`(double"°C")
- **temp_fpgar** Gets the temperature of the right frontend FPGA. Used in EIGER only. `Returns EIGER`(double"°C")
- **i_a** Gets the current of the power supply a on the new chiptest board. `Returns` (int"mV")
- **i_b** Gets the current of the power supply b on the new chiptest board `Returns` (int"mV")
- **i_c** Gets the current of the power supply c on the new chiptest board `Returns` (int"mV")
- **i_d** Gets the current of the power supply d on the new chiptest board `Returns` (int"mV")
- **i_io** Gets the current of the power supply io on the new chiptest board `Returns` (int"mV")
- **vm_a** Gets the measured voltage of the power supply a on the new chiptest board `Returns` (int"mV")
- **vm_b** Gets the measured voltage of the power supply b on the new chiptest board `Returns` (int"mV")
- **vm_c** Gets the measured voltage of the power supply c on the new chiptest board `Returns` (int"mV")

- **vm_d** Gets the measured voltage of the power supply d on the new chiptest board `Returns` (int"mV")

- **vm_io** Gets the measured voltage of the power supply io on the new chiptest board `Returns` (int"mV")

7.5 ADCs

commands to monitor and handle temperature overshoot (only JUNGFRAU)

- **temp_threshold** Sets/gets the threshold temperature. JUNGFRAU ONLY. `Returns` (double"°C")

- **temp_control** Enables/Disables the temperature control. 1 enables, 0 disables. JUNGFRAU ONLY. `Returns int`

- **temp_event** Resets/gets over-temperative event. Put only with option 0 to clear event. Gets 1 if temperature went over threshold and control is enabled, else 0. /Disables the temperature control. JUNGFRAU ONLY. `Returns int`

8 Output settings

Commands to setup the file destination and format

- **outdir [dir]** Sets/gets the file output directory. `Returns` (string)

- **fname [fn]** Sets/gets the root of the output file name `Returns` (string)

- **index [i]** Sets/gets the current file index. `Returns` (int)

- **enablefwrite [i]** Enables/disables file writing. 1 enables, 0 disables. `Returns` (int)

- **overwrite [i]** enables(1) /disables(0) file overwriting. `Returns` (int)

- **currentfname** gets the filename for the data without index and extension. MYTHEN only. `Returns` (string)

- **fileformat** sets/gets the file format for data in receiver. Options: [ascii, binary, hdf5]. Ascii is not implemented in Receiver. `Returns` (string)

9 Actions

Commands to define scripts to be executed during the acquisition flow

- **positions [n [p0..pn-1]]** sets/gets number of angular position and positions to be acquired.. `Returns` (int int..) n [p0..pn-1]
- **startscript [s]** sets/gets the script to be executed at the beginning of the acquisition. `none` unsets. `Returns` (string)
- **startscriptpar [s]** sets/gets a string to be passed as a parameter to the startscript. `Returns` (string)
- **stopscript [s]** sets/gets the script to be executed at the end of the acquisition. `none` unsets. `Returns` (string)
- **stopscriptpar [s]** sets/gets a string to be passed as a parameter to the stopscript. `Returns` (string)
- **scriptbefore [s]** sets/gets the script to be executed before starting the detector every time in the acquisition. `none` unsets. `Returns` (string)
- **scriptbeforepar [s]** sets/gets a string to be passed as a parameter to the scriptbefore. `Returns` (string)
- **scriptafter [s]** sets/gets the script to be executed after the detector has finished every time in the acquisition. `none` unsets. `Returns` (string)
- **scriptafterpar [s]** sets/gets a string to be passed as a parameter to the scriptafter. `Returns` (string)
- **headerafter [s]** sets/gets the script to be executed for logging the detector parameters. `none` unsets. `Returns` (string)
- **headerbefore [s]** sets/gets the script to be executed for logging the detector parameters. `none` unsets. `Returns` (string)
- **headerbeforepar [s]** sets/gets a string to be passed as a parameter to the headerbefore script. `Returns` (string)
- **headerafterpar [s]** sets/gets a string to be passed as a parameter to the headerafter script. `Returns` (string)
- **enacallog [i]** enables/disables logging of the parameters necessary for the energy calibration. 1 sets, 0 unsets. `Returns` (int)

- **angcallog [i]** enables/disables logging of the parameters necessary for the angular calibration. 1 sets, 0 unsets. `Returns` (int)

- **scan0script [s]** sets/gets the script to be executed for the scan 0 level. `none` unsets.

- **scan0par [s]** sets/gets a string to be passed as a parameter to the scan0script

- **scan0prec [i]** sets/gets number of digits to be used for the scan0 variable in the file name.

- **scan0steps [i [s0..sn-1]]** sets/gets number of steps (int) of the scan0 level and their values (float).

- **scan0range [smin smax sstep]** sets scan0 min, max and step, returns the number of steps and their values as scan0steps.

- **scan1script [s]** sets/gets the script to be executed for the scan1 level. `none` unsets.

- **scan1par [s]** sets/gets a string to be passed as a parameter to the scan1script

- **scan1prec [i]** sets/gets number of digits to be used for the scan1 variable in the file name.

- **scan1steps [i [s0..sn-1]]** sets/gets number of steps (int) of the scan1 level and their values (float).

- **scan1range [smin smax sstep]** sets scan1 min, max and step, returns the number of steps and their values as scan1steps.

10 Network

Commands to setup the network between client, detector and receiver

- **rx_hostname [s]** sets/gets the receiver hostname or IP address, configures detector mac with all network parameters and updates receiver with acquisition parameters. Normally used for single detectors (Can be multi-detector). `none` disables. If used, use as last network command in configuring detector MAC. `Returns` (string)

- **rx_udpip [ip]** sets/gets the ip address of the receiver UDP interface where the data from the detector will be streamed to. Normally used for single detectors (Can be multi-detector). Used if different from eth0. `Returns` (string)

- **rx_udpmac [mac]** sets/gets the mac address of the receiver UDP interface where the data from the detector will be streamed to. Normally used for single detectors (Can be multi-detector). `Returns` (string)

- **rx_udpport [port]** sets/gets the port of the receiver UDP interface where the data from the detector will be streamed to. Use single-detector command. `Returns` (int)

- **rx_udpport2 [port]** sets/gets the second port of the receiver UDP interface where the data from the second half of the detector will be streamed to. Use single-detector command. Used for EIGER only. `Returns` (int)

- **detectormac [mac]** sets/gets the mac address of the detector UDP interface from where the detector will stream data. Use single-detector command. Normally unused. `Returns` (string)

- **detectorip [ip]** sets/gets the ip address of the detector UDP interface from where the detector will stream data. Use single-detector command. Keep in same subnet as rx_udpip (if rx_udpip specified). `Returns` (string)

- **txndelay_left [delay]** sets/gets the transmission delay of first packet in an image being streamed out from the detector's left UDP port. Use single-detector command. Used for EIGER only. `Returns` (int)

- **txndelay_right [delay]** sets/gets the transmission delay of first packet in an image being streamed out from the detector's right UDP port. Use single-detector command. Used for EIGER only. `Returns` (int)

- **txndelay_frame [delay]** sets/gets the transmission frame period of entire frame being streamed out from the detector for both ports. Use single-detector command. Used for EIGER only. `Returns` (int)

- **flowcontrol_10g [delay]** Enables/disables 10 GbE flow control. 1 enables, 0 disables. Used for EIGER only. `Returns` (int)

- **zmqport [port]** sets/gets the 0MQ (TCP) port of the client to where final data is streamed to (eg. for GUI). Use single-detector command to set individually or multi-detector command to calculate based on `port` for the rest. Must restart zmq client streaming in gui/external gui `Returns` (int)

- **rx_zmqport [port]** sets/gets the 0MQ (TCP) port of the receiver from where data is streamed from (eg. to GUI or another process for further processing). Use single-detector command to set individually or multi-detector command to calculate based on `port` for the rest. put restarts streaming in receiver with new port. `Returns` (int)

- **rx_datastream** enables/disables data streaming from receiver. 1 enables 0MQ data stream from receiver (creates streamer threads), while 0 disables (destroys streamer threads). `Returns` (int)

- **configuremac [i]** configures the MAC of the detector with these parameters: detectorip, detectormac, rx_udpip, rx_udpmac, rx_udpport, rx_udpport2 (if applicable). This command is already included in `rx_hsotname`. Only put!. `Returns` (int)

- **rx_tcpport [port]** sets/gets the port of the client-receiver TCP interface. Use single-detector command. Is different for each detector if same `rx_hostname` used. Must be first command to communicate with receiver. `Returns` (int)

- **port [port]** sets/gets the port of the client-detector control server TCP interface. Use single-detector command. Default value is 1952 for all detectors. Normally not changed. `Returns` (int)

- **stopport [port]** sets/gets the port of the client-detector stop server TCP interface. Use single-detector command. Default value is 1953 for all detectors. Normally not changed. `Returns` (int)

- **lock [i]** Locks/Unlocks the detector to communicate with this client. 1 locks, 0 unlocks. `Returns` (int)

- **lastclient** Gets the last client communicating with the detector. Cannot put!. `Returns` (string)

11 Receiver commands

Commands to configure the receiver. Not used in MYTHEN.

- **receiver [s]** starts/stops the receiver to listen to detector packets. Options: [`start`, `stop`]. `Returns` (string) status of receiver[`idle`, `running`].

- **r_online [i]** sets/gets the receiver in online/offline mode. 1 is online, 0 is offline. Get is from shared memory. `Returns` (int)

- **r_checkonline** Checks the receiver if it is online/offline mode. Only get! `Returns` (string) "All online" or "[list of offline hostnames] : Not online".

- **framescaught** gets the number of frames caught by receiver. Average of all for multi-detector command. Only get! `Returns` (int)

- **resetframescaught [i]** resets the number of frames caught to 0. i can be any number. Use this if using status start, instead of acquire (this command is included). Only put! `Returns` (int)

- **frameindex [i]** gets the current frame index of receiver. Average of all for multi-detector command. Only get! `Returns` (int)

- **r_lock [i]** locks/unlocks the receiver to communicate with only this client. 1 locks, 0 unlocks. `Returns` (int)
- **r_lastclient** gets the last client communicating with the receiver. Only get! `Returns` (int)
- **r_readfreq [i]** sets/gets the stream frequency of data from receiver to client. i $>$ 0 is the nth frame being streamed. 0 sets frequency to a default timer (200ms). `Returns` (int)
- **rx_fifodepth [i]** sets/gets receiver fifo (between Listener and Writer Threads) depth to i number of frames. Can improve listener packet loss (loss due to packet processing time in Listener threads), not if limited by writing. `Returns` (int)
- **r_silent [i]** sets/gets receiver in silent mode, ie. it will not print anything during real time acquisition. 1 sets, 0 unsets. `Returns` (int)

12 Chiptest board

Commands specific for the new chiptest board as pattern generator

- **adcinvert [mask]** Sets/gets ADC inversion mask (8 digits hex format)
- **adcdisable [mask]** Sets/gets ADC disable mask (8 digits hex format)
- **pattern fn** loads binary pattern file fn
- **patword addr [word]** sets/gets 64 bit word at address addr of pattern memory. Both address and word in hex format. Advanced!
- **patioctrl [word]** sets/gets 64 bit mask defining input (0) and output (1) signals. hex format.
- **patclkctrl [word]** sets/gets 64 bit mask defining if output signal is a clock and runs. hex format. Unused at the moment.
- **patlimits [addr1 addr2]** sets/gets the start and stop limits of the pattern to be executed. hex format. Advanced!
- **patloop0 [addr1 addr2]** sets/gets the start and stop limits of the level 0 loop. hex format. Advanced!
- **patnloop0 [n]** sets/gets the number of cyclesof the level 0 loop (int).
- **patwait0 [addr]** sets/gets the address of the level 0 wait point. hex format. Advanced!

- **patwaittime0 [n]** sets/gets the duration of the witing of the 0 waiting point in clock cycles (int).
- **patloop1 [addr1 addr2]** sets/gets the start and stop limits of the level 1 loop. hex format. Advanced!
- **patnloop1 [n]** sets/gets the number of cyclesof the level 1 loop (int).
- **patwait1 [addr]** sets/gets the address of the level 1 wait point. hex format. Advanced!
- **patwaittime1 [n]** sets/gets the duration of the witing of the 1 waiting point in clock cycles (int).
- **patloop2 [addr1 addr2]** sets/gets the start and stop limits of the level 2 loop. hex format. Advanced!
- **patnloop2 [n]** sets/gets the number of cyclesof the level 2 loop (int).
- **patwait2 [addr]** sets/gets the address of the level 2 wait point. hex format. Advanced!
- **patwaittime2 [n]** sets/gets the duration of the waiting of the 2 waiting point in clock cycles (int).
- **dut_clk [i]** sets/gets the signal to be used as a clock for the digital data coming from the device under test. Advanced!

13 Advanced Usage

This page is for advanced users. Make sure you have first read the introduction.