# MyRailIO©

## CLI reference guide

Version: 0.0.1

Jonas Bjurel
8-10-2024

# MyRailIO© CLI reference guide

A model railway controller backend for signal masts-, sensors-, light effects-, actuators and more.

## Content:

## License and copyright

## Resources

All MyRailIO resources can be found at https://www.myrail.io/

## Introduction

MyRailIO (maɪreɪlio) provides an Input/Output peripheral backend to model railway controllers such as JMRI, RockRail, etc. (at current only JMRI is supported).

MyRailIO provides flexible and configurable capabilities for the model railway controller to throughout the layout capture diverse types of sensors-, control multiple types of signaling masts-, control diverse types of light-effects-, and maneuver actuators such as turnouts-, servos-, and solenoids.

The myRailIO decoders run on multiple cheap ESP32 micro controllers, each connected to multiple cheap I/O satellite FPGAs; All managed and supervised by a centralized management software which in principle is OS independent (currently only evaluated with Windows 11). The decoder provides a Command Line Interface (CLI) for debugging and visibility. It is not intended for production configuration of MyRailIO, infact most of the configurations performed in the CLI are volatile and does not persist a decoder reboot. This document describes the decoder CLI.

## MyRailIO CLI Principles

The MyRailIO Decoder provides a CLI through Telnet. The CLI provides capabilities to view and modify objects through-out the MyRailIO managed object tree - which we refer to as CLI contexts. At any given point in time, the CLI points/operates on a particular context which provides the CLI capabilities for that specific CLI context/Managed object tree instance.
There are three different categories of CLI commands:

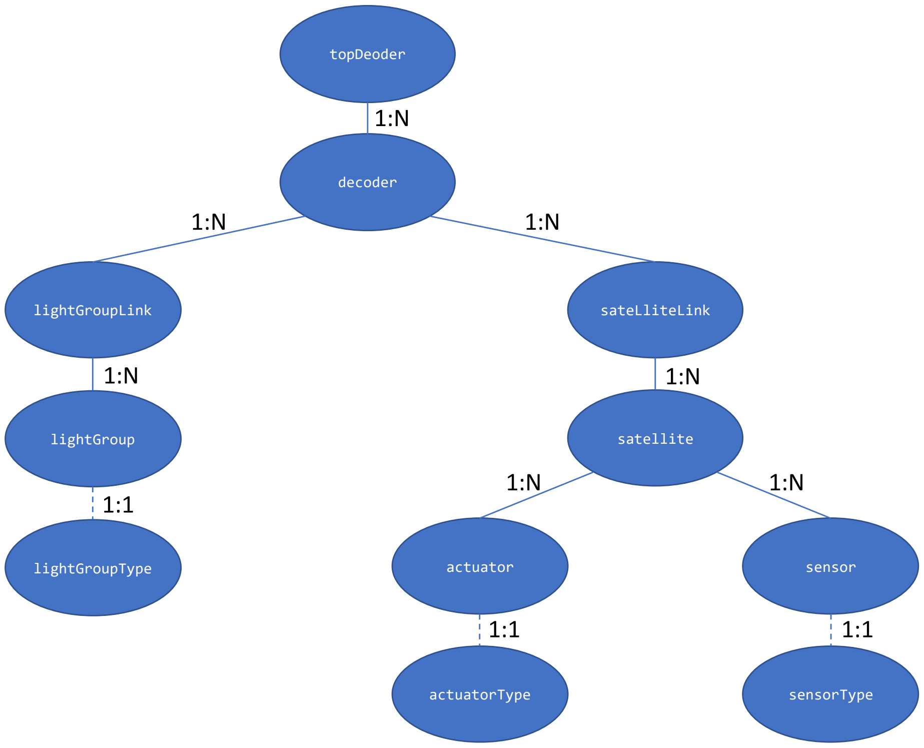
- **Global CLI commands**: These CLI commands are available in all CLI contexts, but does not operate on the specific context, but on the global context. Example: "reboot"…
- **Common CLI commands**: These CLI commands are available in all CLI contexts and provide capabilities that are common for all CLI contexts, it operates on the CLI context (not the global context). Example: "get opstate" or "set debug"…
- **Context unique CLI commands**: These CLI commands are unique to a CLI context. Example: "get aspect" or "set aspect"…

There is a well-defined set of main CLI command methods across all CLI contexts which is globally defined, these cannot be changed or extended by CLI contexts. Examples: help, show, get, set….

The concept of CLI context path provides the capability to navigate across various CLI contexts/myRailIO management object model tree junctions. The CLI context can be changed by explicitly set the context by providing an absolute context path: "set context [/absoluteContextPath]", or by relative context definition: "set context [rellativeContextPath]".

CLI commands can operate on other CLI contexts than the current CLI context by providing context routing information to the CLI command - "command [contextPath]firstArg [secondArg] [-nonPosflags][-nonPosArgs=][][]"… Example: "get ../lglink-1/opstate" or "get ./../lglink-1/opstate" or "get /decoder-0/lglink-1/opstate".
The MyRailIO CLI contextPath concept in large follows the syntax and semantics of Posix/Unix file definitions.

The CLI context model in large follows the myRailIO class/object model which is shown below:



The decoder CLI is case sensitive.
The current implementation does not support CLI history, nor autocompletion.

## MyRailIO CLI verbs

**myRailIO provides the following CLI verbs, a MyRailIO always starts with a CLI verb:**

| **Cmd:** | **CmdCategory:** |
|---|---|
| help | Global CLI command |
| reboot | Global CLI command |
| show | Global-, common-, and context unique |
| get | Global-, common-, and context unique |
| set | Global-, common-, and context unique |
| unset | Global-, common-, and context unique |
| add | Global-, common-, and context unique |
| delete | Global-, common-, and context unique |
| copy | Not implemented |
| paste | Not implemented |
| move | Not implemented |
| start | Global-, common-, and context unique |
| stop | Global-, common-, and context unique |
| restart | Not implemented |

# MyRailIO global commands

## Help commands

### `help [{command} [{sub-MO}]]`

Provides a CLI help, if command is omitted a generic CLI help is provided, otherwise a help text for the provided command [sub-command] is provided.

*Examples:*

>>> *help*

>>> *help reboot*

>>> *help get network*

---

### `help cli`

Provides a generic CLI help, describing principles and main commands.

---

### `show commands [-all] [-help]`

Provides a list of all available commands, if “-all” is provided all commands no matter the current CLI-context are listed, otherwise only those available in the current CLI-context. If “-help is provided, a brief description of each command is provided.

## Context commands

### `get context`

Prints current CLI context.

---

### `set context {context-path}`

Sets current context path. If { context-path } begins with "/" it is considered to be an absolute path, otherwise it is considered to be a relative path. ".." is used to ascend one level in the CLI context topology. "." is a NULL separator which does not change (ascend or descend) the CLI context path, i.e. "./././path represents the same path as "path"

---

### `show topology [-childs]`

Shows the CLI context topology tree.
Current active context is highlighted with an "<<<" indication on the right side of the table.  - If the "-childs" flag is given, only the CLI context topology starting from the current CLI context junction and below is shown.

## Runtime commands

### `Reboot [-panic {panic-message}] [-exception]`

Reboots the decoder. If none of “-panic” or “-exception” is provided, the decoder will be rebooted through a normal software reset. If “-panic {panic message}” is provided, an application panic with a panic message “panic-message” will cause the decoder to reboot. If “-exception” is provided, the decoder will be rebooted as a consequence of an illegal “division-by-zero” machine exception.

---

### `get uptime`

Prints the time in seconds since the previous boot.

---

### `show coredump`

Prints the previously stored core-dump.

---

### `start cpu [-stats]`

Starts a CPU activity.

- If the “-stats” flag is provided, the collection of CPU- and Memory- statistics is started, CPU- and Memory- collection is a prerequisite for some of the statistics that can be provided by “get cpu [-flags]”, “show cpu”, “get memory [-flags]”, and “show memory”. CPU and Memory collection of statistics can have a significant negative impact on the system performance".

---

### `stop cpu [-stats]`

Stops an ongoing CPU activity.

- If the “-stats” flag is provided - ongoing collection of CPU- and Memory- statistics is stopped.

---

## get memory [-internal] [-total] [-available] [-used] [-watermark] [-average {period_s}] [-trend {period_s}] [-maxblock]

Provides heap memory status and statistics:

- If no flags are provided-, a summary of the global (Internal + SPIRAM) memory status is provided.
- If the “-internal” flag is provided, the status of the internal on-chip RAM status is provided, else the total combined memory statistics is shown.
- “-total” : Provides the total installed memory capacity in bytes.
- “-available”: Prints currently available memory capacity in bytes.
- “-used”: Provides currently used memory capacity in bytes.
- “-watermark”: Prints the lowest available memory watermark in bytes.
- “-average {period_s }”: Prints the average memory usage in bytes over “{period_s}” seconds, the CPU and memory statistics collection function needs to be active - see “start cpu -stats”.
- “-trend { period_s }”: Prints the memory usage trend in bytes over “{period_s}” seconds, the CPU and memory statistics collection function needs to be active - see “start cpu -stats”.
- “-maxblock” : Prints the maximum block of heap memory in bytes that can be allocated.

---

## show memory

Shows a summary of the heap status and statistics, it is identical to “get memory”.

*Example:*

```
>>> show memory
Total memory statistics
| FreeMem(B)  | UsedMem(B) | TotalMem(B) | HighMemWatermark(B) | MemUsage(%) | MemUsage10s(deltaB) | MemUsage30s(deltaB) | MemUsage1m(deltaB) | MemUsage10m(deltaB) |
| 4020327     | 471128     | 4491455     | 3931019             | 11.00       | -                   | -                   | -                  | -                   |
```

---

## start memory -allocate{bytes} [-internal|-external|-default]

Starts a heap memory allocation activity.

- The “-allocate {bytes}” flag determines the test-buffer size to be allocated.
- “-internal” Operates on internal memory segments.
- “-external” Operates on external SPI-RAM memory segments.
- “-default” Operates on memory segments as defined by the default OS memory allocation policy.

---

## `stop memory [-allocate]`

Stops earlier started heap memory activities.

- If the “-allocate” flag is provided, memory earlier allocated by "start memory -allocate {bytes}" will be freed.

---

## `set wdt {wdt-id} [-debug] [-timeout{ timeout_ms}] [-action{action}] [-active{value}]`

Sets various parameters of a Watchdog with ID: “{wdt-id}” given by the provided flags:

Available flags:

- “-debug”: Sets the debug flag, allowing intrusive Watchdog CLI commands.
- “-id {watchdog_id}”: The watchdog identity to be altered.
- “-timeout {timeout_ms}”: Sets the watchdog timeout in ms.
- “-action {LOC0|LOC1|LOC2|GLB0|GLBFSF|REBOOT|ESC_GAP}”: Sets the Watchdog escalation ladder actions:
    - "LOC0": Watchdog owner first local action.
    - "LOC1": Watchdog owner second local action.
    - "LOC2" Watchdog owner third local action.
    - "GLB1": Watchdog global action.
    - "GLBFSF": Watchdog global failsafe action.
    - "REBOOT": Watchdog global reboot action.
    - "ESC_GAP": Escalation gap, each escalation action is inter-gaped by one watchdog timer tick.

    The escalation ladder actions can be combined in any way by separating them with “|”, but the escalation ladder order always remains the same - from "LOC0" towards "REBOOT".

- “-active {“True” | ”False”}”: activates or inactivates the Watchdog.

---

## `unset wdt [-debug]`

Un-sets various parameters of WDT given by the provided flags:

Available flags:

- "-debug": Un-sets the debug flag, dis-allowing intrusive Watchdog CLI commands.

---

## `get wdt [-id {wd-id} [-description] | [-active] | [-inhibited] | [-timeout] | [-action] | [-expiries] | [-closesthit]]`

Prints Watchdog parameters and statistics. "get wdt" without flags is identical to "show wdt

"get wdt -id {wd_id}" without any other flags shows all parameters and statistics for that specific Watchdog id

Available flags:

- “-id”: Specifies the WDT id for which information is requested.
- “-description”: Prints the WDT description.
- “-active”: Shows if the WDT is active or not.
- “-inhibited”: Shows if the WDT is inhibited or not - I.e. from a "stop wdt" command.
- “-timeout”: Prints the WDT expiration value [ms]
- “-actions”: Shows requested and existing expiration actions: {FAULTACTION_ESCALATE_INTERGAP | FAULTACTION_LOCAL0 | FAULTACTION_LOCAL1 | FAULTACTION_LOCAL2 | FAULTACTION_GLOBAL_FAILSAFE | FAULTACTION_GLOBAL_REBOOT (See the MyRailIO architecture description for more information.
- “-expiries”: Number of WDT expires.
- “-closesthit”: Closest time to a WDT expiry [ms].

---

## `show wdt`

Shows a summary of the WDT information. Is identical to "get wdt" without flags.

*Example:*

>>> *show wdt*

```
| id: | Description:      | Active: | Inhibited: | Timeout [ms]: | Actions:                    | Expiries: | ClosesThit [ms]: |
| 1   | logJob            | True    | False      | 30000         | ESC_GAP|GLBFSF|REBOOT       | 0         | 21900            |
| 2   | sysStateJob-0     | True    | False      | 10000         | ESC_GAP|GLBFSF|REBOOT       | 0         | 6600             |
| 3   | sysStateJob-1     | True    | False      | 10000         | ESC_GAP|GLBFSF|REBOOT       | 0         | 6600             |
| 4   | sysStateJob-2     | True    | False      | 10000         | ESC_GAP|GLBFSF|REBOOT       | 0         | 6600             |
| 5   | sysStateJob-3     | True    | False      | 10000         | ESC_GAP|GLBFSF|REBOOT       | 0         | 6600             |
| 6   | satLinkWdt-0      | True    | False      | 3000          | ESC_GAP|GLBFSF|REBOOT       | 0         | 2900             |
| 7   | sysStateJob-4     | True    | False      | 10000         | ESC_GAP|GLBFSF|REBOOT       | 0         | 6600             |
.......
```

---

## `stop wdt [-id {wd_id}]`

Stops the reception of WDT feeds, ultimately resulting in a WDT expiry and escalation ladder, this command requires the WDT debug flag to be set - see "set wdt -debug"

Available flags:

- "-id": Specifies the WDT id for which the feeding should be stopped, if "-id" is omitted - all WDTs will be starved

---

## `start wdt [-id {wd_id}]`

Starts the reception of WDT feeds which was earlier stopped, this command requires the WDT debug flag to be set - see "set wdt -debug"

Available flags:

- "-id": Specifies the WDT id for which the feeding should start, if "-id" is omitted - all WDTs will be started.

---

```
clear wdt [-id {wd_id}]|-allstats|-expires|-closesthit
```

Clears certain WDT statistics

- "-id": Specifies the WDT id for which a statistics object should be cleared, if omitted the below given statistics object for all WDTs will be cleared
- "-allstats": Clears all statistics
- "-expires": The WDT expires counter will be cleared
- "-closesthit": The WDT closest hit statistics will be cleared

---

```
set job [-debug] | [-id {job_Id} [-priority {priority 1-24}]]
```

Sets various parameters of a Job queue given by the provided flags:

Available flags:

- "-debug": Sets the debug flag, allowing intrusive Job CLI commands
- "-id {job_Id}"": The Job identity to be altered
- "-priority {jobTaskPriority 0-24}": Sets the job task priority

For mor information about job queues, see the  MyRailIO architecture description.

---

```
unset job [-debug]
```

Un-sets various parameters of Job given by the provided flags:

Available flags:

- "-debug": Un-sets the debug flag, dis-allowing intrusive Job CLI commands.

---

```
get job [-id {job_Id} [-description] | [-maxjobs] | [-currentjobs] |
        [-peakjobs] | [-averagejobs] | [-peaklatency] |
        [-averagelatency] | [-peakexecution] | [-averageexecution] |
        [-priority] | [-priority] | [-overloaded] | [-overloadcnt] |
        [-wdtid] | [-tasksort]]
```

Prints Job queue parameters and statistics. "get job" without flags is identical to "show job"

"get job -id {job_Id}" without any other flags shows all parameters and statistic s for that specific Job id

Available flags:

- “-id”: Specifies the Job id for which information is requested.
- “-description”: Prints the Job description, same as the job task name.
- “-maxjobs”: Shows the total maximum of Job slots/queue depth available.
- “-currentjobs”: Shows currently queued/pending Jobs.
- “-peakjobs”: Shows the peak/maximum number of jobs that have ever been pending in the job queue.
- “-averagejobs”: Shows the average number of jobs that have been pending in the job queue for the past 10 job invocations.
- “-peaklatency”: Shows the peak/maximum latency(uS) that a job has been in queue until start of execution.
- “-averagelatency”: Shows the average latency time(uS) that jobs have been in queue until start of execution for the past 10 job invocations.
- “-peakexecution”: Shows the peak/maximum job execution time(uS) from when a job was dequeued from the job buffer until finished.
- “-averageexecution”: Shows the average job execution time(uS) from when a job was dequeued from the job buffer until finished for the past 10 job invocations.
- “-priority”: Shows the Job task base priority.
- “-overloaded”: Shows the current job queue overload status.
- “-overloadcnt”: Shows the number of job queue overload instances.
- “-wdtid”: Shows related Watchdog Id for the job supervision.
- “-tasksort”: Tasksorting - when task sorting is disabled ("False") strict first-in first-served global policy applies, if not, a sorting policy keeping jobs enqueued from same tasks is applied: I.e. strict first-in first-served per job enqueueing origin task .

## `show job`

Shows a summary of the Job queue information. Is identical to “get job” without flags.

*Example:*

```
>>> show job
| id:  | Desc:          | Max Jobs: | Curr Jobs: | Peak Jobs: | Avr Jobs: | Peak lat(us): | Avr lat(us): | Peak ex(us): | Avr ex(us): | Prio: | O-loaded: | O-load Cnt: | WdtId: | Task sort: |
| 1    | logJob         | 10        | 0          | 11         | 1         | 146575        | 164          | 80061        | 10694       | 5     | No        | 12          | 1      | False      |
| 2    | sysStateJob-0  | 32        | 0          | 1          | 1         | 910           | 295          | 95           | 64          | 10    | No        | 0           | 2      | True       |
| 3    | sysStateJob-1  | 32        | 0          | 0          | 0         | 0             | 0            | 0            | 0           | 10    | No        | 0           | 3      | True       |
| 4    | sysStateJob-2  | 32        | 0          | 1          | 1         | 88            | 88           | 1771         | 1771        | 10    | No        | 0           | 4      | True       |
...
```

---

## `clear job [-id {job_Id}] [-allStats] | [-peakjobs] | [-averagejobs] | [-peaklatency] | [-averagelatency] | [-peakexecution] | [-averageexecution] | [-overloadcnt]]`

Clears certain Job statistics.

Available flags:

- "-id": Specifies the Job id for which a statistics object should be cleared, if omitted the statistics object will be cleared for all Jobs.
- "-allstats": Clears all statistics
- "-peakjobs": Clears stats of the peak/maximum number of jobs that has ever been pending in the job queue
- "-averagejobs": Clears stats of the average number of jobs that has been pending in the job queue for the past 10 job invocations
- "-peaklatency": Clears stats of the peak/maximum latency(uS) that a job has been in queue until start of execution
- "-averagelatency": Clears stats of the average latency time(uS) that jobs have been in queue until start of execution for the past 10 job invocations
- "-peakexecution": Clears stats of the peak/maximum job execution time(uS) from when a job was dequeued from the job buffer until finished
- "-averageexecution": Clears stats of the average job execution time(uS) from when a job was dequeued from the job buffer until finished for the past 10 job invocations
- "-overloadcnt": Clears stats of the number of job queue overload instances

## Network commands

```
set network [-ssid{ssid}] [-pass{pass}] [-hostname{host_name}]
            [-address{ip_address}] [-mask{ip_mask}] [-gw{gw_ip}]
            [-dns{dns_ip}] [-dhcp] [-persist]
```

Sets IP and WIFI networking parameters:

Available flags:

- "-ssid {ssid}": Sets the WiFi SSID.
- "-pass {pass}": Sets the WiFi password.
- "-hostname {host_name}": Sets the host name.
- "-address {ip_address}": Sets the host IPv4 network address.
- "-mask {ip_mask}": Sets the IPv4 network mask.
- "-gw {gw_ip}": Sets the IPv4 network default gateway address.
- "-dns {dns_ip}": Sets the IPv4 DNS address.
- "-dhcp": Sets DHCP operation where the network parameters are assigned from a DHCP server
- "-persist": Persists the network configuration

If any of "-ssid"-, or "pass"- are provided, the WiFi SSID and password will be statically reprovisioned, if one of the two parameters is not provided it will assume the same value as in the previous configuration.

If any of "address"-, "mask"-, "gw"-, or "dns"- are provided, a static IP address configuration will be set for all of those parameters, if any of those are not provided, that parameter will assume a static value from what it previously was set to be (from previously set static configuration, or from previously DHCP assigned configuration).

If "dhcp" is set, all of "address"-, "mask"-, "gw"-, and "dns" will be assigned from the DHCP server.

```
get network [-ssid] [-bssid] [-channel] [-auth] [-rssi] [-mac]
            [-hostname] [-address] [-mask] [-gw] [-dns] [-opstate]
            [-scanap]
```

Prints IP and WIFI networking parameters and statistics. "get network" without flags is identical to "show network".

Available flags:

- “-ssid”: Prints the AP SSID connected to.
- “-bssid”: Prints the AP BSSID connected to.
- “-channel”: Prints the AP WiFi channel connected to.
- “-auth”: Prints current WiFi Autentication/Encryption method.
- “-rssi”: Prints current SNR/RSSI WiFi signal quality.
- “-mac”: Prints decoder WiFi MAC address.
- “-hostname”: Prints decoder host name.
- “-address”: Prints the host IPv4 network address.
- “-mask”: Prints the IPv4 network mask.
- “-gw”: Prints the IPv4 network default gateway address.
- “-dns”: Prints the IPv4 DNS address.
- “-opstate”: Prints current network operational state.
- “-scanap”: Scans available APs and prints the information about them.

*“-scanap” example:*

```
>>> get network -scanap
| SSID:              | BSSID:             | Channel:   | RSSI:      | Encryption:   |
| BREVIK_INTRANET    | B4:FB:E4:2D:F9:28  | 1          | -48        | WPA2_PSK      |
| BREVIK_OPEN        | B6:FB:E4:2D:F9:28  | 1          | -49        | OPEN          |
| BREVIK_EXTRANET    | DE:B3:70:AA:29:5D  | 1          | -70        | WPA2_PSK      |
| BREVIK_2_4_G_LAB   | E6:B3:70:AA:29:5D  | 1          | -71        | WPA2_PSK      |
| BREVIK_INTRANET    | D8:B3:70:AA:29:5D  | 1          | -71        | WPA2_PSK      |
```

## `show network`

Shows a summary of network information. Identical to "get network" without flags.

*Example:*

```
>>> show network
| SSID:              | BSSID:             | Channel: | Encryption: | RSSI: | MAC:               | Host-name:     | DHCP: | IP-Address:   | IP-Mask        | Gateway:     | DNS        |
| BREVIK_2_4_G_LAB   | E6:B3:70:AA:29:5D  | 1        | WPA2_PSK    | -70   | E0:E2:E6:4C:13:30  | esp32-4C1330   | True  | 192.168.2.134 | 255.255.255.0  | 192.168.2.1  | 8.8.8.8    |
```

## Time commands

```
add time [-ntpserver {ntp_server_uri | ntp_server_IPv4Address}]
         [-ntpport{ntp_server_port}]
```

Adds an NTP server.

Available flags:

- “-ntpserver {ntp_server_uri | ntp_server_IPv4Address}”: NTP server URI or IPv4 address.
- “-ntpport {ntp_server_port}”: NTP server port, if a port is not given, the default NTP port:123 will be used.

---

```
delete time [-ntpserver {ntp_server_uri | ntp_server_IPv4Address}]
```

Deletes an NTP server.

Available flags:

- “-ntpserver {ntp_server_uri | ntp_server_IPv4Address}”: Deletes a previously provisioned NTP server with URI : “ntp_server_uri” or IPv4 IP address: “ntp_server_IPv4Address”.

---

```
start time {-ntpclient [-ntpdhcp]}
```

Starts the time services.

Available flags:

- "-ntpclient": Starts the NTP client , if "-ntpdhcp" is provided, the ntp server information is taken from DHCP option 042.

---

```
stop time {-ntpclient}
```

Stops time services.

Available flags:

- “-ntpclient”: Stops the NTP client.

---

```
set time [[-timeofday {timeOfDay}] | [-tod {timeOfDay}] |
          [-epochtime {value}]] [-timezone {tz}]
```

Sets the system time.

Available flags:

- “-timeofday {timeOfDay} | -tod {timeOfDay}”: Sets time of day in UTC, “timeOfDay” format: “YYYY-MM-DDTHH:MM:SS”
- “-epochtime {epochTime_s}”: Sets Epoch time, “epochTime_s” format: NNNNNN - seconds since Jan 1 1970 UTC
- “-timezone {timeZone_h}”: Sets the timezone, “timeZone_h” format (-)NN houres, NN <= 12, E.g. "CET+1".

---

```
get time [[-timeofday] | [-tod]] [[-utc] | [-epochtime]] [-timezone]
         [-daylightsaving] [-ntpdhcp] [-ntpservers] [-ntpsyncstatus]
         [-ntpsyncmode] [-ntpopstate]
```

Prints the system time. "get time", without flags it is identical to “show time”.

Available flags:

- "-timeofday | -tod [-utc]": Prints the local or universal time.
- "-epochtime": Prints the epoch time - seconds since Jan 1 1970 UTC.
- "-timezone": Prints current time-zone.
- "-daylightsaving": Prints daylight-saving status.
- “-ntpdhcp: Prints the NTP DHCP Option 042 status.
- "-ntpservers": Prints the current status of all provisioned NTP servers.
- "-ntpsyncstatus": Prints the current NTP client sync status.
- "-ntpsyncmode": Prints the current NTP client sync mode.
- "-ntpopstate": Prints the current NTP client operational state.

---

## `show time`

Shows a summary of the time services. Identical to "get time" without flags.

*Example:*

```
>>> show time
Time:
| UTC:                                  | Local:                                    | Epoch time:   |
| Fri Jul 19 2024 - 17:55:04 - UTC time | Fri Jul 19 2024 - 19:55:04 - Local time   | 1721411704    |

NTP-Client:
| OpState:        | Sync status:              | Sync mode           | No of servers:  | Time zone:   | Daylight saving: |
| SYNCHRONIZING   | SYNC_STATUS_IN_PROGRESS   | SYNC_MODE_SMOOTH    | 1               | CEST+0200    | True             |

NTP-Servers:
| Index: | NTP Server URI:       | NTP Server IP Address: | Port: | Stratum:   | Reachability: |
| 0      | pool.ntp.org          | 0.0.0.0                | 123   | Unknown    | 0xFF          |
```

## MQTT commands

```
set mqtt [-uri {mqtt_broker_uri | mqtt_broker_IPv4Address}]
         [-port {mqtt_broker_port}] [-clientid {mqtt_client_id}]
         [-qos {mqtt_qos}] [-keepalive {mqtt_keepalive_period_s}]
         [-ping {mqtt_e2e_ping_period_s}] [-persist]
```

Sets MQTT parameters.

Available flags:

- "-uri {mqtt_broker_uri | mqtt_broker_IPv4Address}": Sets MQTT Broker URI to "mqtt_broker_uri " or "mqtt_broker_IPv4Address".
- "-port {mqtt_broker_port}": Sets MQTT Broker port to "mqtt_broker_port".
- "-clientid {mqtt_client_id}": Sets MQTT client ID to "mqtt_client_id".
- "-qos {mqtt_qos}": Sets MQTT default QoS to "mqtt_qos", i.e. 0,1 or 2.
- "-keepalive {mqtt_keepalive_period_s}": Sets MQTT keepalive period to "mqtt_keepalive_period_s" seconds.
- "-ping {mqtt_e2e_ping_period_s}": Sets MQTT overlay server-client MQTT ping period to "mqtt_e2e_ping_period_s" seconds. This represents the supervision between the server and the decoders.
- "-persist": Persists MQTT configuration of Broker URI and Broker port to the decoders provisioning flash store.

---

```
get mqtt [-uri] [-port] [-clientid] [-qos] [-keepalive] [-ping]
         [-maxlatency] [-meanlatency] [-overruns] [-opstate]
         [-subscriptions]
```

Prints MQTT parameters and statistics. "get mqtt" without flags is identical to "show mqtt"

Available flags:

- "-uri": Print MQTT broker URI or IPv4 address
- "-port": Prints the MQTT broker port
- "-clientid"": Prints the MQTT client identifier
- "-qos": Prints the MQTT client default Quality of Service class
- "-keepalive": Prints the MQTT keep-alive period in seconds
- "-ping": Prints the server-decoder ping period in seconds
- "-maxlatency": Prints the MQTT poll loop max latency in uS
- "-meanlatency": Prints the MQTT poll loop mean latency in uS
- "-overruns": Prints the MQTT poll loop overrun counter
- "-opstate": Prints the MQTT client operational state
- "-subscriptions": Prints current subscriptions and call backs.

---

```
show mqtt
```

Shows a summary of the MQTT information. Is identical to "get mqtt" without flags.

*Example:*

```
>>> show mqtt
| URI:              | Port: | Client-Id:        | QoS:  | Keep-alive: | Ping:  | Max latency: | Mean latency: | Overruns: | OP-State          |
| 192.168.2.129     | 1883  | myRailIODecoder   | 0     | 60.00       | 10.00  | 366350       | 239           | 1         | WO                |
```

---

```
clear mqtt [-maxlatency] [-overruns]
```

Clear MQTT statistics.

Available flags:

- "-maxlatency": Clears MQTT max-latency statistics.
- "-overruns" : Clears MQTT overrun statistics.

---

## LOG commands

set log{fatal|error|notice|verbose}

```
set log [-loglevel{log_level}] [-logdestination{rsys_log_uri}]
        [-logconsole]
```

Sets log properties.

Available flags:

- “-loglevel {log_level}” Sets the global log level, the global log level should be one of: should be one of “DEBUG-SILENT” | ”DEBUG-PANIC” | “DEBUG-ERROR” | “DEBUG-WARN” | “DEBUG-INFO” | “DEBUG-TERSE” | “DEBUG-VERBOSE”.
- “-logdestination {rsys_log_uri}” Sets the RSyslog destination URI and starts logging to it.
- “-logconsole” Enables console logging.

---

```
unset log [-logdestination] [-logconsole]
```

Unsets log properties

Available flags:

- “-logdestination” Removes the RSyslog destination(s) and stops logging to it.
- “-logconsole” Stops logging to the console.

---

```
get log [-loglevel] [-logdestination] [-customlogitems] [-missedlogs]
```

Prints various log parameters and metrics, "get log" without flags is identical to "show log".

Available flags:

- “-loglevel” Prints the current global log level
- “-logdestination” Prints the current Rsyslog log destination
- “-customlogitems” Prints custom log items with their respective log levels
- “-missedlogs” Prints the aggregated sum of missed log entries that have been pruned due to overload or lack of resources, this counter can be reset by “clear log -missedlogs".

---

```
add log [-customlogitem -customloglevel{log_level}
        -customlogfile{custom_src_file_name}
        [-customlogfunc{custom_src_fun_name}]]
```

Adds log items.

Available flags:

- “-customlogitem -customloglevel {custom_log_level} -customlogfile {custom_src_file_name} [-customlogfunc {custom_src_fun_name}]” Adds a custom log item for a specific source file name - and an optional function name, with a custom log level, the log level value should be one of “DEBUG-SILENT” | “DEBUG-PANIC” | “DEBUG-ERROR” | “DEBUG-WARN” | “DEBUG-INFO” | “DEBUG-TERSE” | “DEBUG-VERBOSE”.
  A custom log item allows to alter the log verbosity for certain parts of the source code, enabling fine grained debugging for that specific part without having to increase the global log verbosity, potentially increasing system load, or loosing log items.

*Example:*

*add log -customlogitem -customloglevel DEBUG-VERBOSE -customlogfile globalCli.c*
*-customlogfunc* `globalCli::printCustomLogItems`

---

```
delete log [-customlogitem [-all] |
           [-customlogfile{custom_src_file_name}
           [-customlogfunc{custom_src_fun_name}]]
```

Deletes log items

Available flags:

- “-customlogitem [-all]” Deletes all custom log items.
- “-customlogitem -customlogfile {custom_src_file_name} [-customlogfunc {custom_src_fun_name}]" Deletes the custom log item: “custom_src_file_name”::”custom_src_fun_name”.

---

## `show log`

Shows a summary of log information. Identical to “get log” without flags add log.

*Example:*

```
>>> show log
Global log information
| Log-level:        | Log-receiver:      | Log-port  | Missed log entries:  |
| DEBUG-INFO        | 192.168.2.129      | 514       | 3087                 |


Custom log information
| Custom log function:        | Custom log function:                | Custom Log-level:       |
| globalCli.cpp               | globalCli::printCustomLogItems      | DEBUG-VERBOSE           |
```

---

## `clear log [-missedlogs]`

Clears various log metrics

Available flags:

- "-missedlogs" Clears the missed log entries counter.

---

## Other commands

### `set failsafe`

Not supported

---

### `get failsafe`

Not supported

---

## MyRailIO common commands

In difference to MyRailIO Global commands, MyRailIO Common commands operate on objects and parameters related to the current CLI context while Global commands operates on objects and parameters related to a single global objects and parameters. The objects and parameters operated by common commands are standardized and exist for all managed objects/CLI-contexts, while unique context CLI commands operate on objects and parameters unique for a specific Managed object/CLI-context.

### Managed object identifications

```
set systemname {sys_name}
```

Sets the “systemname” of the current Managed Object(MO)/CLI-context. The “debug-flag” of the current Managed Object/CLI-context needs to be active to set the “systemname” - use ”set debug” to enable the “debug-flag”.

**Attention:** The System name of a Managed Object is normally immutable, changing it from the CLI may result in unpredictive consequences and side effects.

---

```
get systemname
```

Prints ”systemname” of the current Managed Object(MO)/CLI-context.

---

```
set username {usr_name}
```

Sets the “username” of the current Managed Object(MO)/CLI-context. The “debug-flag” of the current Managed Object/CLI-context needs to be active to set the “username” - use ”set debug” to enable the “debug-flag”.

**Attention:** The User name of a Managed Object is normally mutable, changing it from the CLI will not however synchronize the new User name with the MyRailIO server and many cause inconsistencies.

---

`get username`

Prints the “username” of the current Managed Object(MO)/CLI-context.

---

`set description {description}`

Sets the “description” of the current Managed Object(MO)/CLI-context. The “debug-flag” of the current Managed Object/CLI-context needs to be active to set the “description” - use ”set debug” to enable the “debug-flag”.

**Attention:** The Description of a Managed Object is normally mutable, changing it from the CLI will not however synchronize the new Description with the MyRailIO server and many cause inconsistencies.

---

`get description`

Prints the “description” of the current Managed Object(MO)/CLI-context.

---

## Managed object states

### `get opstate`

Prints the operational state of the MO:

- "INIT": The managed object is initializing.
- "DISC": The managed object is disconnected (I.e., from WiFi-, MQTT-, or RPC).
- "NOIP": The managed object has not been assigned an IP-address.
- "UDISC": The managed object has not been discovered.
- "UCONF": The managed object has not been configured.
- "DABL": The managed object has been administratively "Disabled" -- see above.
- "SUAVL": The MyRailIO server is missing excessive ping supervision messages from a decoder.
- "CUAVL": A MyRailIO decoder is missing excessive ping supervision messages from the server.
- "ESEC": A Satelite link or a Satelite has experienced a second with extensive errors.
- "ERR": The managed object has experienced a recoverable error.
- "FAIL": The managed object has experienced an unrecoverable error.
- "CBL": The managed object is control blocked due to errors higher up in the managed object hierarchy.
- "UUSED": The managed object is unused.

---

### `set debug`

Sets the debug state of the MO, this enables setting of MO/Sub-MO attributes which may lead to system inconsistency.

A MyRailIO C-Alarm will be raised whenever an MO has the debug state active.

---

### `unset debug`

Un-sets the debug state of the MO.

---

`get debug`

Prints the debug state of the MO.

# MyRailIO context unique commands

## Decoder

-

## Light group link

```
set link {lgLink_number}
```

Sets the lgLink number, the debug-flag needs to be activated to perform this action - see “set debug”.

---

```
get link :
```

Prints the lgLink number.

---

```
get overruns
```

Prints the accumulated number of lgLink over-runs, I.e. for which the lgLink scan had not finished before the next was due.

---

```
clear overruns
```

Clears the counter for accumulated lgLink over-runs.

---

```
get meanlatency
```

Prints the mean latency of the lgLink scan, E.g. how much its start was delayed compared to schedule.

---

```
get maxlatency
```

Prints the maximum latency watermark of the lgLink scan, E.g. how much its start was delayed compared to schedule.

---

`clear maxlatency`

Clears the maximum latency watermark of the lgLink scan.

---

`get meanruntime`

Prints the mean run-time of the lgLink scan, E.g. how long the lgLink scan took.

---

`get maxruntime`

Prints the maximum run-time of the lgLink scan, E.g. how long the lgLink scan took.

---

`clear maxruntime`

Clears the maximum run-time watermark of the lgLink scan.

---

## Light group

`set address {lightGroup_address}`

Sets the LgLink address of the LightGroup, the debug-flag needs to be activated to perform this action.

---

Get address

Prints the LgLink address of the LightGroup.

---

`set ledcnt {ledcnt}`

Sets the number of LEDs/pixels for the LightGroup, the debug-flag needs to be activated to perform this action- see “set debug”..

---

`get ledcnt`

Prints the number of LEDs/pixels for the LightGroup.

---

`set ledoffset {led_offset}`

Sets the LightGroup LED offset on the LgLink, the debug-flag needs to be activated to perform this action - see “set debug”..

---

`get ledoffset`

Prints the LightGroup LED offset on the LgLink.

---

## `set property {property_index} {property_value}`

Sets the LightGroup property value according to given property index, the debug-flag needs to be activated to perform this action - see “set debug”..

---

## `get property [{property_index}]`

Prints the LightGroup properties, if "property_index" is provided, the property corresponding to this index is printed, else all properties are printed.

*Example:*

```
>>> get property 1
Lightgroup property 1: Sweden-3HMS:SL-5HL

>>> get property
| Lightgroup property index:          | Lightgroup property value:          |
| 1                                   | Sweden-3HMS:SL-5HL                  |
| 2                                   | FAST                                |
| 3                                   | NORMAL                              |
```

---

## `set showing {aspect}`

Sets the LightGroup’s current aspect to be shown, the debug-flag needs to be activated to perform this action - see “set debug”.

Valid aspects depend on the LightGroup type/sub-type.

---

## `get showin`

Prints the LightGroup’s current showing.

---

## Satellite link

### `set link {satellite_link_no}`

Sets the Satellite-link number, the debug-flag needs to be activated to perform this action - see “set debug”..

---

### `get link`

Prints the Satellite-link number.

---

### `get txunderrun`

Prints number of Satellite-link TX underruns. I.e. number of occurrences the TX link scan could not be served in a timely manner.

---

### `clear txunderrun`

Clears the Satellite-link TX underruns counter.

---

### `get rxoverrun`

Prints number of Satellite-link RX overruns, I.e. number of occurrences the TX link scan could not be served in a timely manner.

---

### `clear rxoverrun`

Clears the Satellite-link RX overruns counter.

---

### `get timingviolation`

Prints number of Satellite-link timingviolations, I.e. link scans which did not finish in time.

---

`clear timingviolation`

Clears the Satellite-link timingviolation counter.

---

`get rxcrcerr`

Prints number of Satellite-link RX CRC errors, I.e. number of CRC errors detected at the RX far end of the link.

---

`clear rxcrcerr`

Clears the Satellite-link RX CRC error counter.

---

`get remotecrcerr`

Prints the number of Satellite-link remote CRC errors, I.e. the aggregate number of CRC errors reported from each Satellite's RX interface.

---

`clear remotecrcerr`

Clears the Satellite-link RX CRC error counter.

---

`get rxsymbolerr`

Prints number of Satellite-link RX symbol errors, I.e. layer-1 symbol errors detected at the RX far end of the link.

---

`clear rxsymbolerr`

Clears the Satellite-link RX symbol error counter.

---

`get rxsizeerr`

Prints number of Satellite-link RX size errors, I.e. occasions where the datagram size has been detected invalid at the RX far end of the link.

---

`clear rxsizeerr`

Clears the Satellite-link RX size error counter.

---

`get wderr`

Prints number of Satellite-link watchdog errors, I.e aggregated watchdog error reported by Satellites on the Satellite-link.

---

`clear wderr`

Clears the Satellite-link watchdog error counter.

---

## Satellite

### `set address {satellite_address}`

Sets the Satellite address, the debug-flag needs to be activated to perform this action - see “set debug”.
**Note** that the Satellite address is counted from the RX far end of the Satellite link, and it is 0-numbered.

---

### `get address`

Prints the Satellite address.

---

### `get rxcrcerr`

Prints Satellite RX CRC errors, I.e. CRC errors encountered for the datagram bound for the Satellite.

---

### `clear rxcrcerr`

Clear the Satellite RX CRC error counter.

---

### `get txcrcerr`

Prints Satellite TX CRC errors, I.e. CRC errors detected on the satelite-link RX far-end, belonging to datagrams supposedly sent from this Satellite.

---

### `clear txcrcerr`

Clears the Satellite TX CRC error counter

---

### `get wderr`

Prints Satellite watchdog errors, I.e. occasions when a satellite scan update hasn’t been received in due time.

---

## `clear wderr`

Clears the Satellite watchdog error counter.

## Sensor

### `set port {sensor_port}`

Sets the Sensor port number, the debug-flag needs to be activated to perform this action - see “set debug”.

---

### `get port`

Prints the Sensor port number.

---

### `get sensing`

Prints current Sensor state.

---

### `set property {property_index} {property_value}`

Sets the Sensor property according to given property index, the debug-flag needs to be activated to perform this action - see “set debug”.

---

### `get property [{property_index}]`

Prints the Sensor properties, if property index is provided the property corresponding to the index is printed, else all properties are printed.

---

## Actuator

### `set port {actuator_port}`

Sets the Actuator port number, the debug-flag needs to be activated to perform this action - see "set debug".

---

### `get port`

Prints the Actuator port.

---

### `set showing`

Sets the actuator showing state, the debug-flag needs to be activated to perform this action - see "set debug".
Showing state is a string that depends on the type of actuator.

---

### `get showing`

Prints the actuator showing state.

---

### `set property {property_index} {property_value}`

Sets the Actuator property according to given property index and value, the debug-flag needs to be activated to perform this action - see "set debug".

---

### `get property [{property_index}]`

Prints the Actuator properties, if property index is provided the property corresponding to the index is printed, else all properties are printed.

---

## References

1. MyRailIO User guide
2. MyRailIO Architecture description