Tracking Hub Command Interface

Version 1.9



Viz Virtual Studio

Copyright ©2026 Vizrt. All rights reserved.

No part of this software, documentation or publication may be reproduced, transcribed, stored in a retrieval system, translated into any language, computer language, or transmitted in any form or by any means, electronically, mechanically, magnetically, optically, chemically, photocopied, manually, or otherwise, without prior written permission from Vizrt. Vizrt specifically retains title to all Vizrt software. This software is supplied under a license agreement and may only be installed, used or copied in accordance to that agreement.

Disclaimer

Vizrt provides this publication “as is” without warranty of any kind, either expressed or implied. This publication may contain technical inaccuracies or typographical errors. While every precaution has been taken in the preparation of this document to ensure that it contains accurate and up-to-date information, the publisher and author assume no responsibility for errors or omissions. Nor is any liability assumed for damages resulting from the use of the information contained in this document.

Vizrt’s policy is one of continual development, so the content of this document is periodically subject to be modified without notice. These changes will be incorporated in new editions of the publication. Vizrt may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time. Vizrt may have patents or pending patent applications covering subject matters in this document. The furnishing of this document does not give you any license to these patents.

Antivirus Considerations

Vizrt advises customers to use an AV solution that allows for custom exclusions and granular performance tuning to prevent unnecessary interference with our products. If interference is encountered:

- **Real-Time Scanning:** Keep it enabled, but exclude any performance-sensitive operations involving Vizrt-specific folders, files, and processes. For example:
  - C:\Program Files\[Product Name]
  - C:\ProgramData\[Product Name]
  - Any custom directory where [Product Name] stores data, and any specific process related to [Product Name].
- **Risk Acknowledgment:** Excluding certain folders/processes may improve performance, but also create an attack vector.
- **Scan Scheduling:** Run full system scans during off-peak hours.
- **False Positives:** If behavior-based detection flags a false positive, mark that executable as a trusted application.

Technical Support

For technical support and the latest news of upgrades, documentation, and related products, visit the Vizrt web site at www.vizrt.com.

Created on

2026/11/19

Contents

1 Communication with Tracking Hub 4
1.1 Command Syntax 4
1.2 Tokens 4
1.3 Answer 4
1.4 Long Pattern for Answer Handling 5
1.5 Short Pattern for Answer Handling 5
1.6 Failure Messages 5
1.6.1 Unknown Location 6
1.6.2 Unknown Property 6
1.6.3 Unknown Command 6
2 Commands 7
2.1 MAIN 7
2.2 CONFIG 9
2.3 STUDIO 9
2.4 TEMPLATES 10
2.5 TRACKING SYSTEMS 10
2.6 LATTICES 12
2.7 SERVICES 13
2.8 ROUTERS 16
2.8.1 List Available Router Models 16
2.8.2 Manage Routers 16
2.8.3 Configure Individual Routers 16
2.8.4 Configure Presets of Individual Routers 17
2.8.5 Delays per Router 18
2.9 GPIIO 18
2.10 POST 19
2.10.1 Set the Timecode Sources for Live and Post 19
2.10.2 Session Commands 19
2.10.3 Notifications 20
2.10.4 Recording and Replay Commands 20
2.11 LENSFILE 20

1 Communication with Tracking Hub

Tracking Hub provides a command interface similar to that offered by Viz Engine. The default port is `20000`.

1.1 Command Syntax

```
<token> <location>[*<property>]* <command> [<parameter>]*
```

Where:

- **<token>:** The token tells the receiver how to interpret the message, if an answer is expected etc.
- **<location>:** Reference to an object or a collection of objects.
- **<property>:** Reference property or properties or member objects contained in the root object or collection.
- **<command>:** The command to be executed on the property or object.
- **<parameter>:** Command parameter or parameters.

1.2 Tokens

> 0	Client receives and answers carrying the same token.
-1	Client receives no answer.
-2	(only for message *from* Tracking Hub) Reserved for notifications.
-3	(only for message *from* Tracking Hub) Status message.

1.3 Answer

You always get a string as an answer. It starts with a status code followed by the actual content (for example, XML) or an error message.

```
(0|1) [string|xml|message]
```

0	Failure	Followed by an optional error message.
1	Success	Followed by an optional status message or an answer in a prearranged format.

Copyright © 2026 Vizrt

1.4 Long Pattern for Answer Handling

```
if (answer != null) {
    // Handle error answer
    if (CommandMessage.IsFailure(answer)) {
        if (CommandMessage.IsErrorMessage(answer)) {
            Log.Error("{0} returned an error: {1}", command,
CommandMessage.ExtractMessage(answer));
            return;
        }
    }

    // Handle success answer
    if (CommandMessage.IsSuccess(answer)) {
        // HANDLE VALID ANSWER HERE
        return;
    }
    Log.Error("{0} returned an invalid answer", command);
} else {
    // No answer at all
    Log.Error("{0} returned no answer", command);
}

This pattern is implemented in: CommandMessage.IsSuccess(string answer, string
command)
```

1.5 Short Pattern for Answer Handling

```
if (CommandMessage.IsFailure(answer)) {
    // REPORT ERROR HERE
    return; // then bail out
}
```

1.6 Failure Messages

⚠ **Note:** Any command message may fail and return an error. It is because of the command interface having changed with a new version of Tracking Hub.

The most common generic error messages are:

Copyright © 2026 Vizrt

1.6.1 Unknown Location

```
0 Unknown location: <location>
```

Returned if the top level location of the command message is unknown.

1.6.2 Unknown Property

```
0 Unknown property: <location>[*<property>]*
```

Returned if command message starts with a known location, but contains an unknown property in the following property path.

1.6.3 Unknown Command

```
0 Unknown command: <location>[*<property>]* <command>
```

Returned if command message starts with a known location and valid property path, but ends on an unknown command.

Copyright © 2026 Vizrt

2 Commands

2.1 MAIN

Tracking Hub matching, available ports, IPs, etc. (MAIN*):

```
MAIN*COMPORTS GET = <portname>[,<portname>]*
MAIN*PARAMETERLIST GET = <parametername>[,<parametername>]*
MAIN*PROTOCOLS GET = <protocolname>[,<protocolname>]*

MAIN*LOCAL_IPS GET = <ip>[,<ip>]*
MAIN*VIZ_IP SET <ip> or <adaptername>
            GET = <adaptername>
MAIN*TRACKING_IP SET <ip> or <adaptername>
                 GET = <adaptername>

MAIN*CONFIGURATIONS GET = [<name>[,<name>]*]?

//MAIN START not used
//MAIN STOP  not used
//MAIN*RUNNING GET = <bool> not used

MAIN*WRITE_LOG_FILE SET <bool>
                    GET = <bool>
MAIN*FORWARD_LOG SET <bool>
                 GET = <bool>

MAIN*LICENSE SET <key>
             GET = <name>,<number of cameras>,<days left> (name is either Cameras or
Dongle, -1 means unlimited cameras/days)

MAIN*DEBUGLEVEL SET <int>   (range is  1 ... 10, 1 only emergency, 10 all)
                GET = <int>

MAIN*DEBUGLEVEL++
MAIN*DEBUGLEVEL--

MAIN*TH_VERSION GET = <version>
Tracking Hub version as major.minor.patch, for example 1.99.0.

MAIN*ROLE SET <int>
GET = <int>
Backup role of this Tracking Hub: 0 none, 1 Master, 2 Slave. Setting it applies the
backup rule immediately – see Backup Configuration.

MAIN*NDI_SOURCES GET = <name>[,<name>]*
NDI sources currently visible to Tracking Hub.

MAIN*TCPROTOCOLS GET = <protocolname>[,<protocolname>]*
```

Copyright © 2026 Vizrt

```
Available timecode protocols, as MAIN*PROTOCOLS does for tracking protocols.

MAIN*LIC_TYPE SET <int>
GET = <int>
License system, matching the LIC_Type setting in the user interface.

MAIN*LIC_LOC SET <int>
GET = <int>
Where a license is acquired, matching LIC_LOC in the user interface: 0 local, 1
 network, 2 local + network.

MAIN*LIC_CAM SET <int>
GET = <int>
Number of licensed cameras.

MAIN*3RDPLICENSE GET = <string>
Third party license information. Read only.

MAIN*PARAMETERSERVICEPORT SET <port>
GET = <port>
Port of the parameter service.

MAIN*WEBSOCKETPORT SET <port>
GET = <port>
Port of the websocket interface. Setting it reopens the websocket.

MAIN*RINGBUFFERSIZE SET <int>
GET = <int>

MAIN*AI_PATH SET <path>
Path used by the AI components. Sending the command without a value clears the path.
There is no GET.

MAIN*CSINTERPOL SET <int>
GET = <int>
Center shift interpolation, the same setting stored as CentershiftInterpolation in
BaseConfig.xml.

MAIN*USEWINSOCK2 SET <int>
GET = <int>
Stored as Usewinsock2 in BaseConfig.xml.

MAIN*BACKUP_IP SET <ip>
GET = <ip>
Address of the backup Tracking Hub. Setting an empty or too short value clears the
address and deactivates the backup partner.

MAIN*SYNCHRONIZE SET
Runs the synchronization from the Master to the Slave – see Backup Configuration for
 what is transferred.

MAIN*POST_AUTOLOAD SET <0|1>
GET = <0|1>
```

Copyright © 2026 Vizrt

```
With 1, the current post session is stored as the session to load on the next start.

MAIN*TIMING SET <value>
Toggles the send timing insertion. The value is required but not evaluated.

MAIN*CANBUSTEST SET <value>
Toggles the CAN bus test. The value is required but not evaluated.
```

2.2 CONFIG

Store configuration on the Tracking Hub machine:

```
CONFIG EXISTS = 0|1
CONFIG SAVE
CONFIG LOAD
CONFIG SET = <name>
CONFIG GET = <name>

CONFIG RESET

CONFIG*BASE SAVE
CONFIG*BASE DISMISS
CONFIG SAVE_AS <name>
Saves the current configuration under a new name and makes it the active one. This
writes the whole set: the studio configuration file, the router configuration, the
cyc and the video wall. The name may contain spaces. Without a name the answer is “No
filename given!”.

CONFIG SEND = <xml>
Returns the current configuration as XML without writing anything to disk.
```

2.3 STUDIO

Base studio configuration (STUDIO*):

```
STUDIO GET = <xml>

STUDIO*MODE SET FREE|AV|VIZ
            GET = SET
STUDIO*FREQUENCY SET 50|60|59.94
                 GET = SET
STUDIO*SHAPE SET LSHAPE_LEFT|LSHAPE_RIGHT|USHAPE|WALL|HOLE
             GET = SET
STUDIO*CYC SET <xml>
           GET = SET
STUDIO*TIMING GET = <bool>
```

Copyright © 2026 Vizrt

```
STUDIO*SENDDELAY GET = <int> [in ms]
                 SET   <int> [in ms]
STUDIO*BUSYLOOP  GET = <int> [type]
                 SET   <int> [type]
STUDIO*CYC DISTRIBUTE
Pushes the current cyc to all connected Viz Engines. Add this to the existing
STUDIO*CYC entry.

STUDIO*VW SET <xml>
GET = <xml>
DISTRIBUTE
The video wall, handled exactly like the cyc: SET replaces it, GET returns it,
DISTRIBUTE pushes the current one to all connected Viz Engines. Sending SET with a
value shorter than three characters deletes the video wall.

STUDIO*BASECFG GET = <xml>
Returns BaseConfig.xml from the configuration folder. The matching SET exists but is
handled on the large-file path rather than as a normal command.
```

2.4 TEMPLATES

Templates stored on the Tracking Hub machine (TEMPLATES*):

```
TEMPLATES*COUNT GET = <amount of templates>
TEMPLATES LIST = <name1, name2, . . .>
TEMPLATES*<name> GET = <xml>
TEMPLATES*<name> SET <xml>
TEMPLATES*<name> DELETE
TEMPLATES*<name> LOAD = <xml>
TEMPLATES*<name> USE <name>
TEMPLATES*<name> SAVE <ts_name1 ts_name2 . . . rig_name1 rig_name2 . . . 1 2>
```

2.5 TRACKING SYSTEMS

Tracking Systems set up on the Tracking Hub machine (TRACKING_SYSTEMS*)

```
TRACKING_SYSTEMS*COUNT GET = <number of tracking systems>

TRACKING_SYSTEMS CREATE <name>
TRACKING_SYSTEMS DELETE <name>
TRACKING_SYSTEMS XMLFILES GET = <string>
TRACKING_SYSTEMS*<idx/name> GET = <xml>
TRACKING_SYSTEMS*<idx/name> CONNECT
TRACKING_SYSTEMS*<idx/name> DISCONNECT
TRACKING_SYSTEMS*<idx/name>*INDEX GET = <index>
```

Copyright © 2026 Vizrt

```
TRACKING_SYSTEMS*<idx/name>*NAME SET <name>
                                 GET = <name>
TRACKING_SYSTEMS*<idx/name>*SLOTINDEX SET <slotindex>
                                   GET = <slotindex>
TRACKING_SYSTEMS*<idx/name>*PROTOCOL SET <protocolname>
                                  GET = <protocolname>
TRACKING_SYSTEMS*<idx/name>*NETUSE SET <netuse>
                                GET = <netuse>
TRACKING_SYSTEMS*<idx/name>*COMPORT SET <name>
                                 GET = <name>
TRACKING_SYSTEMS*<idx/name>*BAUDRATE SET <baud>
                                  GET = <baud>
TRACKING_SYSTEMS*<idx/name>*PARITY SET <baud>
                                GET = <baud>
TRACKING_SYSTEMS*<idx/name>*STOPBITS SET <baud>
                                  GET = <baud>
TRACKING_SYSTEMS*<idx/name>*DATASIZE SET <baud>
                                  GET = <baud>
TRACKING_SYSTEMS*<idx/name>*HOST SET <ip>
                                 GET = <ip>
TRACKING_SYSTEMS*<idx/name>*PORT SET <port>
                                 GET = <port>
TRACKING_SYSTEMS*<idx/name>*XMLFILE SET <string>
                                  GET = <string>

TRACKING_SYSTEMS*<idx/name>*STATUS GET = DISCONNECTED|NOT_RECEIVING|BAD_TIMING|
GOOD_TIMING

//only in running mode
TRACKING_SYSTEMS*<idx/name>*PARAMETERS GET = <parametername>[,<parametername>]*

TRACKING_SYSTEMS*<idx/name>*SENDRAWDATA SET <bool>

TRACKING_SYSTEMS*<idx/name>*RAWDATA_OFFSET <parametername> ACT

TRACKING_SYSTEMS*<idx/name>*RAWDATA_OFFSET <parametername> SET <int>

// not in use any longer
//TRACKING_SYSTEMS*<idx/name>*SENDDELAY GET = <ms>
//                                    SET <ms>

//this command switches hexdumplogging
TRACKING_SYSTEMS*<idx/name>*HEXLOG SET <bool>
                                 GET = <bool>

TRACKING_SYSTEMS*<idx/name>*TICKCOUNT SET <parametername, value>[,<parametername,
value>]
                                    GET = <parametername, value>[,<parametername,
value>]

TRACKING_SYSTEMS*<idx/name>*RCVTIMECORR SET <bool>
                                      GET = <bool>
TRACKING_SYSTEMS*<idx/name>*NDI_COMPRESSED SET <0|1>
```

Copyright © 2026 Vizrt

```
GET = <0|1>
Receive Compressed. Only the metadata of the NDI stream is read, which keeps GPU and
CPU load down. On by default.

TRACKING_SYSTEMS*<idx/name>*NDI_AVSYNC SET <0|1>
GET = <0|1>
Use AVsync. The NDI library tries to harmonize the timing of incoming and outgoing
frames. Off by default and not recommended.

TRACKING_SYSTEMS*<idx/name>*NDI_TIMING SET <0|1|2>
GET = <0|1|2>
What the tracker paces its work on: 0 VIZ, 1 NDI, 2 NDI av. The default is 2.

TRACKING_SYSTEMS*<idx/name>*LONGITUDE SET <string>
GET = <string>

TRACKING_SYSTEMS*<idx/name>*LATITUDE SET <string>
GET = <string>

TRACKING_SYSTEMS*<idx/name>*ALTITUDE SET <string>
GET = <string>

Longitude, latitude and altitude define the GPS zero point of the studio. They are
used by the HD Skycam protocol, which converts them to UTM coordinates as the origin
of its tracking data. Other protocols ignore them.
```

2.6 LATTICES

Lattices set up on the Tracking Hub machine (LATTICES*):

```
LATTICES*COUNT GET = <zero-based index>
LATTICES CREATE <name>
LATTICES DELETE <name>
LATTICES LENSFILES GET = <string>
LATTICES LENSFILES RELOAD
LATTICES*<idx/name> CREATE PARENT|CHILD <childname>
LATTICES*<idx/name> GET = <xml>

LATTICES*<idx/name>*INDEX GET = <index>
LATTICES*<idx/name>*NAME SET <string>
                         GET = <string>
LATTICES*<idx/name>*FILTER_ZOOM SET <bool>
                                GET = <bool>

LATTICES*<idx/name>*TYPE SET SIMPLE_CAM|OBJECT|LATTICE
                         GET = SIMPLE_CAM|OBJECT|LATTICE

LATTICES*<idx/name>*SLOTINDEX SET <slotindex>
                              GET = <slotindex>
```

Copyright © 2026 Vizrt

```
LATTICES*<idx/name>*<parameter>*NAME SET <string>
                                      GET = <string>
                                *OFFSET SET <float>
                                        GET = <float>
                                *INVERT SET <bool>
                                        GET <bool>
                                *DELAY SET <float>
                                       GET <float>
LATTICES*<idx/name>*TRACKINGDELAY SET <frames(float)>
LATTICES*<idx/name>*VISUAL_XML SET <xml>
                                GET = <xml>
LATTICES*<idx/name>*CALIBRATION SET <bool>
                                 GET = <bool>
LATTICES*<idx/name>*CALIBRATIONDONE GET = <bool>
LATTICES*<idx/name>*CALIBRATION_RANGE GET = <zoom_min> <zoom_max> <zoom> <focus_min>
<focus_max> <focus>
LATTICES*<idx/name>*LENSRANGE SET <lensrange_min> <lensrange_max>
                              GET = <lensrange_min> <lensrange_max>

LATTICES*<idx/name>*SCALEVAUES SET <ScXNear, ScXWide, ScYNear, ScYWide>
                               GET = <ScXNear, ScXWide, ScYNear, ScYWide>

LATTICES*<idx/name>* LENSFILE SET <string>
                              GET = <string>

LATTICES*<idx/name>* LENSFILE_LENSEXT SET <string>
                                      GET = <string>
LATTICES*<idx/name>*LENSVALUES SET <0|1> = <stream_id>
Returns the id of the lens value stream for this rig, which a client can use to
receive lens values over the websocket interface. The argument is required but is
currently not evaluated – see MAIN*WEBSOCKETPORT for the port the stream runs on.
```

2.7 SERVICES

Services set up on the Tracking Hub machine (SERVICES*):

```
SERVICES*COUNT GET = <zero-based index>
SERVICES*BY_INDEX*<idx> GET = <xml>
                        START
                        STOP
SERVICES*BY_INDEX*<idx>*SLOTINDEX GET = <slotindex>
                                  SET <slotindex>
SERVICES*BY_INDEX*<idx>*RUNNING GET = <bool>

SERVICES*BY_ID*<service_id> GET = <xml>
                            START
                            STOP
SERVICES*BY_ID*<service_id>*SLOTINDEX GET = <slotindex>
                                      SET <slotindex>
SERVICES*BY_ID*<service_id>*RUNNING GET = <bool>
```

Copyright © 2026 Vizrt

```
SERVICES ADD PARAMETER ALL                          <ip> <port> = <service_id>
SERVICES ADD PARAMETER             <lattice_name>   <ip> <port> = <service_id>
SERVICES ADD TRACKING_TIMING       <ts_name>        <ip> <port> = <service_id>
SERVICES ADD COMMUNICATION_TIMING  <service_id>     <ip> <port> = <service_id>
SERVICES ADD CAMERA                <lattice_name>   <ip> <port> <cameranumber> =
<service_id>
SERVICES ADD OBJECT                <lattice_name>   <ip> <port> = <service_id>
SERVICES ADD TIMECODE              <timcode>        <ip> <port> = <service_id>

SERVICES REPLACE <service_id> (TRACKING_TIMING|COMMUNICATION_TIMING) (<ts_name>|
<service_id>)    <ip> <port>                = <service_id>
SERVICES REPLACE <service_id> (PARAMETER|OBJECT|TIMCODE)             (<lattice_name>|
<timcode>)  <ip> <port>                = <service_id>
SERVICES REPLACE <service_id> CAMERA
<lattice_name>             <ip> <port> <cameranumber> = <service_id>
SERVICES REMOVE <service_id>
SERVICES REMOVE_ADDR <ip>
SERVICES REMOVE_ALL

SERVICES ADD STATE <objectname> <ip> <port> = <service_id>

SERVICES ADD TRFLOW <objectname> <ip> <port> = <service_id>

SERVICES ADD TRFLOWRAW <objectname> <ip> <port> = <service_id>

SERVICES ADD MULTIOBJECT <lattice_name> <ip> <port> <cameranumber> = <service_id>

SERVICES ADD MOCAP <lattice_name> <ip> <port> <cameranumber> = <service_id>

SERVICES ADD VIDEOWALL <lattice_name> <ip> <port> <cameranumber> = <service_id>

SERVICES ADD TCCAMERA <lattice_name> <ip> <port> <cameranumber> = <service_id>

SERVICES ADD NDI <lattice_name> <ip> <port> <cameranumber> = <service_id>

SERVICES ADD TARGET <lattice_name> <ip> <port> = <service_id>
SERVICES ADD TARGET <lattice_name> <ip> <port> <mapprefix> = <service_id>
The second form sets the shared memory map prefix.

The following properties are addressed as SERVICES*BY_ID*<service_id>*<property>.

*NAME SET <string>
GET = <string>

*NETUSE SET <netuse>
GET = <netuse>

*COMPORT SET <name>
GET = <name>

*BAUD SET <baud>
GET = <baud>
```

Copyright © 2026 Vizrt

```
*PARITY SET <int>
GET = <int>

*STOPBITS SET <int>
GET = <int>

*DATASIZE SET <int>
GET = <int>

*UDPPORT SET <port>
GET = <port>

*TCPROTOCOL SET <protocolname>
GET = <protocolname>
Timecode protocol of the service.

*ADDITIONALDELAY SET <float>
GET = <float>
Delay added on top of the rig delay, in fields.

*LUTFILE SET <string>
GET = <string>
Look-up table used by the service.

*WALLGROUP SET <int>
GET = <int>
Video wall group the service belongs to.

*USECANBUS SET <0|1>
GET = <0|1>

*TOGGLECS SET <0|1>
GET = <0|1>
Center shift toggle.

*NDI_SOURCE SET <string>
NDI stream the service sends into.

*NDI_COMPRESSED SET <0|1>
GET = <0|1>
Receive Compressed, on by default.

*NDI_AVSYNC SET <0|1>
GET = <0|1>
Use AVsync, off by default.

*NDI_TIMING SET <0|1|2>
GET = <0|1|2>
What the service paces its output on: 0 VIZ, 1 NDI, 2 NDI av. The default is 2.

*NDIQS SET <int>
GET = <int>
```

Copyright © 2026 Vizrt

```
NDI queue size, which shifts the delay of the whole NDI signal.

A service can carry several rig/camera pairs. These commands manage them, addressed
as SERVICES*BY_ID*<service_id>.

ADD <slot> <lattice_name> <cameranumber>
Adds a rig/camera pair to the service.

DELETE <slot> <lattice_name> <cameranumber>
Removes a rig/camera pair.

CHANGEOBJ <slot> <lattice_name> <cameranumber>
Replaces the rig of a pair.

CHANGECAM <slot> <lattice_name> <cameranumber>
Replaces the camera number of a pair.

EXTSTART
EXTSTOP
Start and stop the service externally. Also available under SERVICES*BY_INDEX*<idx>.
```

2.8 ROUTERS

Configure routers controlled by Tracking Hub (ROUTERS*):

2.8.1 List Available Router Models

```
ROUTERS*MODEL_LIST GET = <model>[,<model>]*
```

2.8.2 Manage Routers

```
ROUTERS ADD <model> <name> = <index>
ROUTERS REMOVE <name> | <index>
ROUTERS*COUNT GET = <count>
```

2.8.3 Configure Individual Routers

```
ROUTERS*<idx/name> GET = <xml>
ROUTERS*<idx/name>*INDEX GET = <index>
ROUTERS*<idx/name>*NAME SET <string>
                        GET = <string>
ROUTERS*<idx/name>*MODEL SET <model>
                         GET = <model>
```

Copyright © 2026 Vizrt

```
ROUTERS*<idx/name>*AB_MODE SET <mode>
                           GET = <mode>
ROUTERS*<idx/name>*NETUSE SET <netuse>
                          GET = <netuse>
ROUTERS*<idx/name>*COMPORT SET <name>
                           GET = <name>
ROUTERS*<idx/name>*BAUDRATE SET <baud>
                            GET = <baud>
ROUTERS*<idx/name>*PARITY SET <baud>
                          GET = <baud>
ROUTERS*<idx/name>*STOPBITS SET <baud>
                            GET = <baud>
ROUTERS*<idx/name>*DATASIZE SET <baud>
                            GET = <baud>
ROUTERS*<idx/name>*HOST SET <ip>
                        GET = <ip>
ROUTERS*<idx/name>*PORT SET <port>
                        GET = <port>
ROUTERS*<idx/name> CONNECT
ROUTERS*<idx/name> DISCONNECT
ROUTERS*<idx/name>*CONNECTED GET = <boolean>
ROUTERS*<idx/name>*INPUTS*<idx>*NAME SET <string>
                                     GET = <string>
ROUTERS*<idx/name>*OUTPUTS*<idx>*NAME SET <string>
                                      GET = <string>
ROUTERS*<idx/name>*CURRENT_PRESET SET <string>
                                  GET = <string>
```

2.8.4 Configure Presets of Individual Routers

```
ROUTERS*<idx/name>*PRESETS CREATE <name> = <index>
ROUTERS*<idx/name>*PRESETS DELETE <name>|<index>
ROUTERS*<idx/name>*PRESETS*COUNT GET = <count>
ROUTERS*<idx/name>*PRESETS*<idx/name> GET = <xml>
ROUTERS*<idx/name>*PRESETS*<idx/name>*NAME SET <string>
                                         GET = <string>
ROUTERS*<idx/name>*PRESETS*<idx/name> CONNECT <input-index> <output-index>
ROUTERS*<idx/name>*PRESETS*<idx/name> DISCONNECT <input-index> <output-index>
ROUTERS*<idx/name>*PRESETS*<idx/name>*CAMERAS ADD <ip> <port> <cameranumber> =
<index>
ROUTERS*<idx/name>*PRESETS*<idx/name>*CAMERAS REMOVE <index>
ROUTERS*<idx/name>*PRESETS*<idx/name>*CAMERAS*COUNT GET = <int>
ROUTERS*<idx/name>*PRESETS*<idx/name>*CAMERAS*<idx> GET = <xml>
ROUTERS*<idx/name>*PRESETS*<idx/name>*GPIIO ADD <devicename> <port> <pin> <pressed>
                                          GET  = <devicename> <port> <pin>
<pressed>
ROUTERS*<idx/name>*PRESETS*<idx/name>*GPIIO REMOVE
ROUTERS*<idx/name>*MANUAL_CONTROL GET = <xml>

ROUTERS*<idx/name>*MANUAL_CONTROL*ENABLED SET <boolean>
```

Copyright © 2026 Vizrt

```
                                        GET = <boolean>

ROUTERS*<idx/name>*MANUAL_CONTROL CONNECT <input> <output>
ROUTERS*<idx/name>*MANUAL_CONTROL DISCONNECT <input> <output>
ROUTERS*<idx/name>*IMAGE_MODE SET <0|1|2|3>
GET = <0|1|2|3>
Image mode of the router: 0 off, 1 foreground, 2 background, 3 DVE. A value outside
this range is read back as 0 when the configuration is loaded.

ROUTERS*<idx/name>*DEPLUGIN SET <int>
GET = <int>
```

2.8.5 Delays per Router

```
ROUTERS*<idx/name>*PRESET_DELAY SET <double>
                                GET = <double>

ROUTERS*<idx/name>*ENGINE_DELAY SET <double>
                                GET = <double>
```

2.9 GPIIO

GPIIO used on the Tracking Hub machine (gpiio).

```
GPIIO*COUNT GET = <count>

GPIIO*MONITOR START
GPIIO*MONITOR STOP
GPIIO*MONITOR GET = <boolean>

GPIIO*<idx/name> GET = <devicename> <count inputports> <count outputports>

GPIIO*<idx/name>*INDEX GET = <index>
GPIIO*<idx/<name>*NAME GET = <string>

Notifications (only sent from Tracking Hub) if gpi triggered when monitoring is on
GPIIO*MONITOR PRESSED  <devicename> <port> <pin>
GPIIO*MONITOR RELEASED <devicename> <port> <pin>
GPIIO NEW <devicename> <count inputports> <count outputports>
Registers a GPI IO device that has been found. Tracking Hub adds it to its device
list and re-checks the router configuration for presets bound to that device.

GPIIO CHANGE <devicename> <port> <state>
Sets the input state of one port of a device.

GPIIO TEST START
GPIIO TEST STOP
```

Copyright © 2026 Vizrt

```
Starts and stops the Sealevel 410 worker, for checking a connected device.
```

2.10 POST

Setup of the Tracking Hub post system.

2.10.1 Set the Timecode Sources for Live and Post

```
POST*LIVE_SOURCE SET  <string>
                 GET = <string>
POST*POST_SOURCE SET <string>
                 GET = <string>
POST*TIMECODE_SOURCES GET <string>[,<string>]*

POST*SIM_LIVE SET <int>
Sets the simulated live timecode. SimLive and SimPost allow tracking data to be
recorded without a timecode reader card. For analysis only – never use them for
 production recording.

POST*SIM_POST SET <int>
Sets the simulated post timecode.

POST*SIM_POST_INCREMENT SET <float>
Sets the rate at which the simulated post timecode advances.

POST*DEBUG SET <int>
Sets the timecode debug level.
```

2.10.2 Session Commands

```
POST*SESSION CREATE <string>(create a new session)
             START (starts recording)
             STOP (stops recording)
             GET_LIST = <string>[,<string>]
             LOAD <string>
             DELETE <string>
             INFO_GET = <string>;[,<string>]*
POST*SESSION CLOSE
Closes the loaded session.

POST*SESSION STATUS_GET = <string>
State of the current session.

POST*SESSION GET_CAMERA_CHANNELS = <string>
```

Copyright © 2026 Vizrt

```
Camera channels of the loaded session. These are the channels that can be assigned to
a camera service – see Post System, Camera Channels.
```

2.10.3 Notifications

Loading saved session file progress as a percentage `FILE_PROGRESS <float>`. Loading of saved session file is complete `FILE_FINISH`.

2.10.4 Recording and Replay Commands

```
POST*PARAMETER_RECORDING GET <Parameter Name> = 1 (recording on) | 0 (recording off)
                         SET <Parameter Name> <0|1>
POST*PARAMETER_REPLAY GET <Parameter Name> = 1 (replay on) | 0 (replay off)
                      SET <Parameter Name> <0|1>
                      GET_DELAY <Parameter Name> = <float>
                      SET_DELAY <float>

TRACKING_SYSTEMS*<name>*RECORDING GET = 1 (recording on) | 0 (recording off)
                                  SET <0|1>
TRACKING_SYSTEMS*<name>*REPLAY GET = 1 (replay on) | 0 (replay off)
                               SET <0|1>
TRACKING_SYSTEMS*<index>*RECORDING GET = 1 (recording on) | 0 (recording off)
                                   SET <0|1>
TRACKING_SYSTEMS*<index>*REPLAY GET = 1 (replay on) | 0 (replay off)
                                SET <0|1>
TRACKING_SYSTEMS*<name>*REPLAY GET_DELAY = <float>
                               SET_DELAY <float>
TRACKING_SYSTEMS*<index>*REPLAY GET_DELAY = <float>
                                SET_DELAY <float>

SERVICES*RECORDING GET <service id> = 1 (recording on) | 0 (recording off)
                   SET <service id> <0|1>
SERVICES*REPLAY GET <service id> = 1 (replay on) | 0 (replay off)
                SET <service id> <0|1>
SERVICES*REPLAY GET_DELAY <service id> = <float>
SERVICES*REPLAY SET_DELAY <service id> <delay>
```

2.11 LENSFILE

Lensfile, load, save and change on the Tracking Hub machine (LENSFILE*).

```
LENSFILE*LOGIN <pwd> = <xml_lensfile> ??? correct error handling ???
LENSFILE*NEW
LENSFILE*LOCK <service_id> = <LockID>
LENSFILE*UNLOCK <LockID >
```

Copyright © 2026 Vizrt

```
LENSFILE*XML_GET <LockID > = <xml_lenfsile>
LENSFILE*SELECT <lensfilename> =  lenfile (binary) loaded for edit
LENSFILE*SAVE <LockID,>
LENSFILE*SAVEAS <LockID, name>

LENSFILE*GROUP SELECT <LockID  parameter>
LENSFILE*GROUP NEW <LockID  parameter>
LENSFILE*GROUP ADD < LockID , parameter >
LENSFILE*GROUP DELETE < LockID , parameter>
LENSFILE*GROUP PARAMETERMOVE <LockID , parameter groupID >
LENSFILE*GROUP PARAMETERMOVE <LockID , parameter >   //move into a new group

LENSFILE*PARAMETER<parameter>VALUEADD< LockID , zoom,focus >
LENSFILE*PARAMETER<parameter>VALUEDEL< LockID , zoom,focus >
LENSFILE*PARAMETER<parameter>VALUECHANGE< LockID , zoom,focus,newvalue>

LENSFILE*ACTIVE GET = all active services

LENSFILE*PROXY SEND<LockID,string>
LENSFILE*PROXY REQ < LockID string> = <answer>

LENSFILE*PARAMETER<parameter>GET_TRIANGULATION< LockID > = <string>
Triangulation data of the parameter in the locked lens file.

LENSFILE*PARAMETER<parameter>GET_SAMPLES< LockID > = <string>
Sample points of the parameter in the locked lens file.

LENSFILE*PARAMETER<parameter>SET_INTERPOLATION< LockID , interpolation >
Interpolation used for the parameter. Accepted values are linear, linearmix and
precalc.

All three require the lens file to be locked. If the LockID is unknown, the answer is
“Lensfile is not locked”.
```

Copyright © 2026 Vizrt