
# TCI

## Telescope Control Interface

*Specifications of the Interface to the Telescope Hardware's Control Software (TCS)*

Version 1.4

**Document Id:**

4PI-DOC-03-008-02-2

Michael Ruder, Daniel Plasa, Peter Kroll

Revision 464 as of 19th November 2003 (michael)

**Revision History:**

| | |
|---|---|
| 29.11.2002 | Initial Version (alpha) (dp) |
| 05.12.2002 | Prefinal Version (dp, pk) |
| 06.12.2002 | Finalized Version 1.1 (dp, rs:fe) |
| 25.03.2003 | English Version 1.2 with TPL specific parts removed (mr) |
| 14.07.2003 | Updated to 1.3, included DIMM interface, pointing and refraction (dp) |
| 08.11.2003 | Updated to 1.4, added sensors, RS errors, POINTING now undocumented (mr) |



$Id: TCI.tex 464 2003-11-19 15:39:51Z michael$

# Contents

# 1 Preface

This document covers the interface to the telescope control software (TCS) running on the computer in the control cabinet. For the communication the protocol "TPL2" is used (refer to [1]).

# 2 Basics

- The TCI will include a list of modules which will give an ordered, hierarchical access to all important functions.
- For debugging and problem analysis there will also be modules which give access to internal settings, like controller settings, cabinet switches etc.
- In addition to the TCI core modules there may be one or more additional modules covering your ordered software. These modules will be covered in a separate document.

# 3 List of Modules

## 3.1 Core Modules

The following modules are provided by the current version of the TCS:

| Name | Function |
|---|---|
| `AXIS[]` | Control over the telescope axes. Usually an array of two (for azimuth and zenith distance) |
| `CABINET` | Everything that is related to the control cabinet |
| `CONFIG` | Configuration parameters and information (like the telescope mount type etc.) |
| `DEROTATOR` | Control over the image derotator |
| `FOCUS` | Control over the focus motor |
| `GPS` | Access to the GPS receiver for time and location data |
| `SENSOR[]` | Any sensors (temperatures, pressure) |
| `TRAJECTORY` | Module for transferring and running trajectories |

## 3.2 Additional Modules

The following modules can be provided as an option.

| Name | Function |
|---|---|

| Name | Function |
|---|---|
| CAMERA[] | Access to camera(s) [only as an option] |
| COVER[] | Any covers that can be controlled (like mirror covers) |
| DEROTATOR[] | Control over image derotator(s) (i.e. support for multiple derotators) |
| DOME | Control over the dome |
| FILTER[] | Control over the filter wheel(s) (i.e. support for multiple wheels) |
| FOCUS[] | Control over more than one focus motor |
| MIRROR | Mirror parameter (e.g., control of a flip mirror) |
| SWITCH[] | Any switches (pumps etc.) |
| WEATHER[] | Access to weather station(s) |

Some of the modules are defined as an array (shown as []) to allow multiple instances. If only one instance is available, the module will not be defined as an array.

## 3.3 Module AXIS

With this module array (typically with 2 elements, azimuth and zenith distance) a low level control to the telescope axes is provided. It is mainly for monitoring and maintenance tasks, as it will not provide functions for precise tracking. For this task, the module TRAJECTORY (see section 3.12) should be used which allows a synchronized movement of both telescope axes and the derotator axis.
The following functions are provided in this module:

| Name | Type | Access | Description |
|---|---|---|---|
| TYPE | INT | RO | Type identifier, see below |
| VERSION | STRING | RO | Hardware version. Newer version may introduce new functionality |
| REVISION | STRING | RO | Hardware revision. Newer revision will not introduce new functionality |
| DEVICEID | STRING | RO | Hardware device identifier, unique for all TCI devices |
| POWER | INT | RW | Switch power for this component (0=off, 1=on). The main power (CABINET.POWER) has to be turned on to enable or disable power for single components, otherwise this command will fail |

| Name | Type | Access | Description |
|---|---|---|---|
| POWER_STATE | INT | RO | Power state of the component (0=off, 1=on) |
| REFERENCED | INT | RO | Referenced state of this component (0=not referenced, 1=referenced) |
| REALPOS | FLOAT | RO | True current position in degrees. For azimuth, zero will be in the south (northern hemisphere) resp. north (southern hemisphere). The degrees are counted south to west. For zenith distance, zero will be in the zenith, 90 at the horizon (looking to the south if the azimuth is zero), -90 in the opposite direction (if the telescope supports movement through the zenith). (See appendix A for an explanation of the coordinate systems). With the properties !MIN and !MAX the accessible range of positions can be inquired. |
| CURRPOS | FLOAT | RO | Current position in degrees, corrected by the given OFFSET (in the same unit as REALPOS. |
| TARGETPOS | FLOAT | RW | The position the telescope is currently moving to (in the same unit as REALPOS). While a trajectory is executed, the variable will constantly be updated to the current position on the trajectory. By writing to this variable, the telescope will switch into direct positioning mode, cancelling a possibly running trajectory (see section 3.12). The callback function will return after the position has been reached so the execution of this command may take a while. |
| TARGETDISTANCE | FLOAT | RO | The distance between CURRPOS and TARGETPOS. |

| Name | Type | Access | Description |
|---|---|---|---|
| OFFSET | FLOAT | RW | Additional offset (in the same unit as REALPOS) that will be added to *all* positioning requests for that component if CABINET.OFFSET_MODE is not 0. This can be used for a guiding system. It will also be used to implement the manual movement with an optional bottle if CABINET.OFFSET_MODE is 2. |
| CURRSPEED | FLOAT | RO | The current speed of the axis in degree per second |
| CURRACC | FLOAT | RO | The current acceleration in degree per second squared. |

The following axis types are currently defined:

| Type no | Description |
|---|---|
| 0 | Not connected (since a telescope needs at least two axes, this should not occur) |
| 1 | Standard TCI Axis |
| 2 | Standard TCI Axis, V2, which includes direct trajectory support (planned) |

## 3.4 Module CABINET

This module allows direct access to the control cabinet and provides functions for turning on and off main power as well as a lot of diagnostic functions.
The following functions are provided in this module:

| Name | Type | Access | Description |
|---|---|---|---|
| TYPE | INT | RO | Type identifier, see below. |
| VERSION | STRING | RO | Hardware version. Newer version may introduce new functionality. |
| REVISION | STRING | RO | Hardware revision. Newer revision will not introduce new functionality. |

| Name | Type | Access | Description |
|---|---|---|---|
| POWER | INT | RW | Switch main power on or off (0=off, 1=on). Individual components can be turned on and off as long as main power has been turned on with the respective POWER variable of the component. After turning on main power all components will be powered on automatically. |
| POWER_STATE | INT | RO | Main power state (0=off, 1=on, −1=emergency stop). The power state of individual components can be checked with the respective POWER_STATE variable of the component. |
| REFERENCED | INT | RO | All powered drives referenced (0=not all referenced, 1=all referenced, −1=referencing in progress). The referenced state of individual components can be checked with the respective REFERENCED variable of the component. |
| OFFSET_MODE | INT | RW | Consideration of the additional offset for all components (0=offset will be ignored, 1=offset will be used, 2=offset will be used and can be modified with the optional bottle) |
| TIME | FLOAT | RO | Precision time source of the real time core. This is the time used for executing trajectories. The format is seconds since January 1st, 1970 UTC in millisecond resolution. |

The following axis types are currently defined:

| Type no | Description |
|---|---|
| 0 | No type (this should not occur) |
| 1 | RS Style Control Cabinet |
| 2 | RP Style Control Cabinet |

There are several other internal variables available, mostly for identifying the hardware or debugging and maintenance tasks. These are however strongly TYPE specific.

### 3.4.1 RS Style specific Cabinet features

| Name | Type | Access | Description |
|---|---|---|---|
| MODE | INT | RO | Control cabinet mode (0=local, no remote (TPL) access, 1=remote, TPL controlled, -1=no operation possible). |
| MANUAL_CONTROL | STRUCT | — | Information about the optional bottle for manual control (see below) |
| VERSION_TEXT | STRING | RO | Concatenated string of all version identifiers of the used components |
| HW_ID | STRING | RO | Unique hardware ID of the cabinet |
| SW_ID | STRING | RO | Unique ID of the low level driver software running on the cabinet computer |
| SW_VERSION | STRING | RO | Version of the low level driver software |
| CONFIG_ID | STRING | RO | Unique hardware ID of the control components mounted on the telescope |
| PARAM_ID | STRING | RO | Unique ID for the current set of parameters for the low level driver software |
| PARAM_VERSION | STRING | RO | Version of the current parameter set |
| PARAM_NAME | STRING | RO | Name of the current parameter set |
| LOG[] | STRUCT | — | Structure array with contents of driver log files (see below) |
| STATUS | STRUCT | — | Structure allowing access to driver status/error flags (see below) |
| RTCORE | MODULE | — | Module allowing access to internal driver variables (see below) |
| DEBUG | STRUCT | — | Structure for realtime debugging information (see below) |

The CABINET.MANUAL_CONTROL structure gives access to informations and settings of the optional bottle.

| Name | Type | Access | Description |
|---|---|---|---|
| MODE | INT | RO | Bottle state (0=turned off, 1=turned on) |
| BUTTONLIGHT | FLOAT | RW | Brightness of bottle buttons (0=lowest, 1=brightest) |

The CABINET.LOG[] structure array allows the readout of the logfiles created by the realtime core.

| Name | Type | Access | Description |
|---|---|---|---|
| NAME | STRING | RO | Name of the log file |
| FILE | STRING | RO | The entire log file contents as one string |

The CABINET.STATUS structure allows a readout of the driver status and error flags. These flags can also be reset.

| Name | Type | Access | Description |
|---|---|---|---|
| STATUS | INT | RW | Current status of the system (0=everything ok, 1=warning, 2=stop, 3=panic) / Reset all errors on write |
| LATCH | STRING | RO | Error latch bit field (transmitted as hex string). If an error occurs, even if it vanishes before readout of the CURRENT variable, the corresponding bit will stay set in this variable until it is reset by writing to STATUS |
| LATCH_COUNT | INT | RO | Number of set bits in the LATCH register |
| CURRENT | STRING | RO | Bit field (transmitted as hex string) showing all currently active errors |
| SYMBOLS[] | STRING | RO | Contains a symbol for each bit in the LATCH and CURRENT registers. The bits are numbered (starting with 0) in the hex strings from right (the lowest bits first) to left. For an explanation of all possible errors, refer to appendix B. |

The CABINET.RTCORE module gives read-only access to the internal states of the realtime core.

| Name | Type | Access | Description |
|---|---|---|---|
| DLG_STATE | INT | RO | Internal state of the dialog process |
| RTM_STATE | INT | RO | Internal state of the realtime process |
| RSVARS[] | STRUCT | — | Access to all internal parameter variables (see below) |

The CABINET.RTCORE.RSVARS[] structure array allows read-only access to all internal variables of the realtime core.

| Name | Type | Access | Description |
|---|---|---|---|
| NAME | STRING | RO | Name of the parameter |
| VALUE | STRING | RO | Value of the parameter |

| Name | Type | Access | Description |
|---|---|---|---|
| DESCRIPTION | STRING | RO | Description of the parameter |
| UNIT | STRING | RO | Unit of the parameter |

The CABINET.DEBUG structure allows recording of realtime debugging information

| Name | Type | Access | Description |
|---|---|---|---|
| DO_TRACE | INT | RW | Starts the recording of the given number of traces. Each trace has 4000 points and every point is TRACE_DELAY milliseconds apart. The traces are written to the flash disk of the TCC. By reading this variable, the remaining number of traces can be acquired. |
| TRACE_DELAY | INT | RW | The delay in milliseconds between two points of a trace. |
| TRACE_SPACE | INT | RO | The free space in kilobytes on the flash disk of the TCC. |
| OSZI_CHANNEL | INT | RW | Allows the selection of the channel for the oszi functionality. Each channel represents a fixed group of variables that are to be monitored. |

### 3.4.2 RP Style specific Cabinet features

| Name | Type | Access | Description |
|---|---|---|---|
| INIT | INT | WO | Writing a value of 1 forces the telescope to re-initialize all axes. A value of 0 re-initializes only axes that are not initialized (possibly due to a emergency stop). |
| VERSION_TEXT | STRING | RO | Concatenated string of all version identifiers of the used components |
| HW_ID | STRING | RO | Unique hardware ID of the cabinet |
| SW_ID | STRING | RO | Unique ID of the low level driver software running on the cabinet computer |
| SW_VERSION | STRING | RO | Version of the low level driver software |
| CONFIG_ID | STRING | RO | Unique hardware ID of the control components mounted on the telescope |
| PARAM_ID | STRING | RO | Unique ID for the current set of parameters for the low level driver software |

| Name | Type | Access | Description |
|---|---|---|---|
| PARAM_VERSION | STRING | RO | Version of the current parameter set |
| PARAM_NAME | STRING | RO | Name of the current parameter set |
| STATUS | MODULE | — | Module allowing access to driver status/error flags (see below) |
| CORE | MODULE | — | Module allowing access to internal driver variables (see below) |

The CABINET.STATUS module allows a readout of the driver status and error flags. These flags can also be reset.

| Name | Type | Access | Description |
|---|---|---|---|
| STATUS | INT | RW | Current status of the system (0=everything ok, 1=warning, 2=stop, 3=panic) / Reset all errors on write |
| LATCH | STRING | RO | A string with all occurred errors numbers separated by blanks. |

The CABINET.CORE module gives read-only access to the internal states of the realtime core and the parameter set.

| Name | Type | Access | Description |
|---|---|---|---|
| UPTIME | INT | RO | Uptime of the core control in seconds |
| EW[] | STRUCT | — | Access to all internal parameter variables |
| EW[].NAME | STRING | RO | Name of the parameter |
| EW[].VALUE | STRING | RO | Value of the parameter |

## 3.5 Module CONFIG

This module is used to gather configuration information about the telescope in general. Currently only the telescope mount type can be inquired.
The following functions are provided in this module:

| Name | Type | Access | Description |
|---|---|---|---|
| TYPE | INT | RO | Type identifier, see below |

The following telescope (mount) types are currently defined:

| Type no | Description |
|---|---|
| 0 | No telescope (very unusual...) |
| 1 | Alt-Azimuth mount, capable of moving through the zenith |

| Type no | Description |
|---|---|
| 2 | Alt-Azimuth mount, not capable of moving through the zenith |

## 3.6 Module DEROTATOR

This module (or, optionally, module array) allows direct control of the derotator(s). This is mainly for monitoring and maintenance tasks, as it will not provide functions for precise movement during tracking. For this task, the module `TRAJECTORY` (see section 3.12) should be used which allows a synchronized movement of the derotator axes with both telescope axes.
The following functions are provided in this module:

| Name | Type | Access | Description |
|---|---|---|---|
| `TYPE` | INT | RO | Type identifier, see below |
| `VERSION` | STRING | RO | Hardware version. Newer version may introduce new functionality |
| `REVISION` | STRING | RO | Hardware revision. Newer revision will not introduce new functionality |
| `DEVICEID` | STRING | RO | Hardware device identifier, unique for all TCI devices |
| `POWER` | INT | RW | Switch power for this component (0=off, 1=on). The main power (`CABINET.POWER`) has to be turned on to enable or disable power for single components, otherwise this command will fail |
| `POWER_STATE` | INT | RO | Power state of the component (0=off, 1=on) |
| `REFERENCED` | INT | RO | Referenced state of this component (0=not referenced, 1=referenced) |

| Name | Type | Access | Description |
|---|---|---|---|
| REALPOS | FLOAT | RO | True current position in degrees. Zero will be on the upper side (when looking toward the flange.) The degrees are counted in the same direction as for the zenith distance (Type 1 and 3) or in the same direction as for the azimuth (Type 2 and 4) or in mathematical positive direction (Type 5) Refer to appendix A for an explanation of the coordinate systems and possible derotator locations). With the properties !MIN and !MAX the allowed range of positions can be inquired. |
| CURRPOS | FLOAT | RO | Current position in degrees, corrected by the given OFFSET (in the same unit as REALPOS. |
| TARGETPOS | FLOAT | RW | The position the derotator is currently moving to (in the same unit as REALPOS). While a trajectory is executed, the variable will constantly be updated to the current position on the trajectory. By writing to this variable, the telescope will switch into direct positioning mode, cancelling a possibly running trajectory (see section 3.12). The callback function will return after the position has been reached so the execution of this command may take a while. |
| TARGETDISTANCE | FLOAT | RO | The distance between CURRPOS and TARGETPOS. |
| OFFSET | FLOAT | RW | Additional offset (in the same unit as REALPOS) that will be added to *all* positioning requests for that component if CABINET.OFFSET_MODE is not 0. This can be used for a guiding system. It will also be used to implement the manual movement with the supplied paddle if CABINET.OFFSET_MODE is 2 |
| CURRSPEED | FLOAT | RO | The current speed of the derotator in degree per second |

| Name | Type | Access | Description |
|---|---|---|---|
| CURRACC | FLOAT | RO | The current acceleration in degree per second squared. |

The following derotator types are currently defined:

| Type no | Description |
|---|---|
| 0 | Not connected |
| 1 | Standard TCI derotator on focus 1 |
| 2 | Standard TCI derotator on focus 2 |
| 3 | Standard TCI derotator on focus 3 |
| 4 | Standard TCI derotator on focus 4 |
| 5 | Standard TCI derotator on focus 5 |
| 6 | Standard TCI derotator on focus 1, V2, which includes direct trajectory support (planned) |
| 7 | Standard TCI derotator on focus 2, V2, which includes direct trajectory support (planned) |
| 8 | Standard TCI derotator on focus 3, V2, which includes direct trajectory support (planned) |
| 9 | Standard TCI derotator on focus 4, V2, which includes direct trajectory support (planned) |
| 10 | Standard TCI derotator on focus 5, V2, which includes direct trajectory support (planned) |

## 3.7 Module DOME

| Name | Type | Access | Description |
|---|---|---|---|
| TYPE | INT | RO | Type identifier, see below |
| VERSION | STRING | RO | Hardware version. Newer version may introduce new functionality |
| REVISION | STRING | RO | Hardware revision. Newer revision will not introduce new functionality |
| DEVICEID | STRING | RO | Hardware device identifier, unique for all TCI devices |
| POWER | INT | RW | Switch power for this component (0=off, 1=on). The main power (CABINET.POWER) has to be turned on to enable or disable power for single components, otherwise this command will fail |

| Name | Type | Access | Description |
|---|---|---|---|
| POWER_STATE | INT | RO | Power state of the component (0=off, 1=on) |
| REFERENCED | INT | RO | Referenced state of this component (0=not referenced, 1=referenced) |

The following dome types are currently defined:

| Type no | Description |
|---|---|
| 0 | Not connected |
| 1 | Classic dome with slit position. |
| 2 | Fold enclusure. Either open or closed. |

Depending on the TYPE the following functions are defined:

### 3.7.1 Classic DOME features

| Name | Type | Access | Description |
|---|---|---|---|
| REALPOS | FLOAT | RO | True current position of the slit in degrees. Same meaning as with the azimuth axis. With the properties !MIN and !MAX the allowed range of positions can be inquired. |
| CURRPOS | FLOAT | RO | Current position in degrees, corrected by the given OFFSET. |
| TARGETPOS | FLOAT | RW | The position slit is currently moving to (in the same unit as REALPOS). By writing to the variable, the slit will start positioning. The callback function will return after the position has been reached so the execution of this command may take a while. |
| TARGETDISTANCE | FLOAT | RO | The distance between CURRPOS and TARGETPOS. |
| OFFSET | FLOAT | RW | Additional offset (in the same unit as CURRPOS) that will be added to *all* positioning requests for that component if CABINET.OFFSET_MODE is not 0. This can be used for a guiding system. It will also be used to implement the manual movement with the supplied paddle if CABINET.OFFSET_MODE is 2 |
| CURRSPEED | FLOAT | RO | The current speed in degree per second |

| Name | Type | Access | Description |
|---|---|---|---|
| CURRACC | FLOAT | RO | The current acceleration in degree per second squared. |
| ENCLOSURE_STATE | FLOAT | RO | Current position of the slit enclosure. Since the Slit does not have an encoder, this will be 0 if it is open, 1 if it is closed and 0.5 if somewhere in between. |
| ENCLOSURE | INT | RW | The position the slit enclosure is currently moving to (as defined with ENCLOSURE_STATE). By writing to the variable, the slit will open or close. |

### 3.7.2 Fold DOME features

| Name | Type | Access | Description |
|---|---|---|---|
| ENCLOSURE_STATE | FLOAT | RO | Current position. Since the dome does not have an encoder, this will be 0 if it is open, 1 if it is closed and 0.5 if somewhere in between. |
| ENCLOSURE | INT | RW | The position the dome is currently moving to (as defined with ENCLOSURE_STATE). By writing to the variable, the dome will open or close. |

## 3.8 Module FILTER

This module (or, optionally, module array) provides functions to move the filter wheel(s) and select the filters by index.
The following functions are provided in this module:

| Name | Type | Access | Description |
|---|---|---|---|
| TYPE | INT | RO | Type identifier, see below |
| VERSION | STRING | RO | Hardware version. Newer version may introduce new functionality |
| REVISION | STRING | RO | Hardware revision. Newer revision will not introduce new functionality |
| DEVICEID | STRING | RO | Hardware device identifier, unique for all TCI devices |

| Name | Type | Access | Description |
|---|---|---|---|
| POWER | INT | RW | Switch power for this component (0=off, 1=on). The main power (CABINET.POWER) has to be turned on to enable or disable power for single components, otherwise this command will fail |
| POWER_STATE | INT | RO | Power state of the component (0=off, 1=on) |
| REFERENCED | INT | RO | Referenced state of this component (0=not referenced, 1=referenced) |

The following filter types are currently defined:

| Type no | Description |
|---|---|
| 0 | Not connected |
| 1 | Filter wheel with discrete index numbers |
| 2 | Filter wheel with continuous positions |

### 3.8.1 Discrete index filters

For filters with discrete positions, the following functions will additionally be provided:

| Name | Type | Access | Description |
|---|---|---|---|
| CURRINDEX | INT | RO | Currently selected filter number. With the properties !MIN and !MAX the allowed range of positions can be inquired. |
| TARGETINDEX | INT | RW | The filter number the wheel is currently turning to. By writing to the variable, the filter wheel will start to move towards the selected filter. The command will be completed once the position has been reached. |
| TARGETDISTANCE | INT | RO | The distance between CURRINDEX and TARGETINDEX. |

### 3.8.2 Continuous position filters

For filters with continuous positioning (e.g., polarization filters) instead the following functions are provided:

| Name | Type | Access | Description |
|---|---|---|---|
| REALPOS | FLOAT | RO | True current filter wheel position in degrees. With the properties !MIN and !MAX the allowed range of positions can be inquired. |
| CURRPOS | FLOAT | RO | The current filter wheel position in degrees, corrected by the given OFFSET. |
| TARGETPOS | FLOAT | RW | The position the wheel is currently turning to (in the same unit as REALPOS). |
| TARGETDISTANCE | FLOAT | RO | The distance between REALPOS and TARGETPOS. |
| OFFSET | FLOAT | RW | Additional offset (in the same unit as REALPOS) that will be added to *all* positioning requests for that component if CABINET.OFFSET_MODE is not 0. This can be used for a guiding system. It will also be used to implement the manual movement with the supplied paddle if CABINET.OFFSET_MODE is 2. |

## 3.9 Module FOCUS

This module (or, optionally, module array) provides functions to move the focus motor(s).
The following functions are provided in this module:

| Name | Type | Access | Description |
|---|---|---|---|
| TYPE | INT | RO | Type identifier, see below |
| VERSION | STRING | RO | Hardware version. Newer version may introduce new functionality |
| REVISION | STRING | RO | Hardware revision. Newer revision will not introduce new functionality |
| POWER | INT | RW | Switch power for this component (0=off, 1=on). The main power (CABINET.POWER) has to be turned on to enable or disable power for single components, otherwise this command will fail. If the power is turned off and the focus motor has a brake, it will automatically come on. |
| POWER_STATE | INT | RO | Power state of the component (0=off, 1=on) |

| Name | Type | Access | Description |
|---|---|---|---|
| REFERENCED | INT | RO | Referenced state of this component (0=not referenced, 1=referenced) |
| REALPOS | FLOAT | RO | True current position in millimeter. Smaller values mean a smaller distance between primary and secondary mirror. Zero is the hardware limited, smallest distance between these two mirrors. With the properties !MIN and !MAX the allowed range of positions can be inquired. |
| CURRPOS | FLOAT | RO | Current position in millimeter. |
| TARGETPOS | FLOAT | RW | The position the focus motor is currently moving to (in the same unit as REALPOS). By writing to the variable, the focus motor will start to move towards this position. The command will be completed once the position has been reached. |
| TARGETDISTANCE | FLOAT | RO | The distance between CURRPOS and TARGETPOS. |
| OFFSET | FLOAT | RW | Additional offset (in the same unit as REALPOS) that will be added to *all* positioning requests for that component if CABINET.OFFSET_MODE is not 0. It will be used to implement the manual movement with the supplied paddle if CABINET.OFFSET_MODE is 2 |
| CURRSPEED | FLOAT | RO | The current speed of the focus motor in millimeter per second |

The following axis types are currently defined:

| Type no | Description |
|---|---|
| 0 | Not connected |
| 1 | Focus motor with brake system |
| 2 | Focus motor without brake system |

## 3.10 Module GPS

With this module, access to the GPS receiver is possible. Time and position information can be acquired. The following functions are provided in this module:

| Name | Type | Access | Description |
|---|---|---|---|
| TYPE | INT | RO | Type identifier, see below |
| VERSION | STRING | RO | Hardware version. Newer version may introduce new functionality |
| REVISION | STRING | RO | Hardware revision. Newer revision will not introduce new functionality |
| TIME | STRING | RO | Date and time as Unix formatted text string (as received by the GPS receiver, second resolution only) |
| POS | STRUCT | — | Structure with the GPS Position (see below) |

The following GPS types are currently defined:

| Type no | Description |
|---|---|
| 0 | Not connected |
| 1 | HOPF GPS system (no height information available, will always be zero) |

The GPS.POS structure gives access to the GPS position from the receiver.

| Name | Type | Access | Description |
|---|---|---|---|
| LAT | FLOAT | RO | Latitude in degrees |
| LON | FLOAT | RO | Longitude in degrees |
| HEIGHT | FLOAT | RO | Height in meters (if provided by the receiver, otherwise zero) |

## 3.11 Module SENSOR

This module array provides access to all temperature, pressure etc. sensors of the telescope.

| Name | Type | Access | Description |
|---|---|---|---|
| TYPE | INT | RO | Type identifier, see below |
| VERSION | STRING | RO | Hardware version. Newer version may introduce new functionality |
| REVISION | STRING | RO | Hardware revision. Newer revision will not introduce new functionality |
| VALUE | INT | RO | Sensor value in the unit specified for the type (see below) |

The following sensor types are currently defined:

| Type no | Description |
|---|---|
| 0 | Not connected |
| 1 | Temperature, unit °C |
| 2 | Pressure, unit Pa |
| 3 | Flow, unit $\frac{m^3}{s}$ |

## 3.12 Module TRAJECTORY

This module allows movement of both telescope axes and the derotator axis. In contrast to the AXIS modules (section 3.3) and the DEROTATOR module(s) (section 3.6) all axes will be moved synchronously and will reach the given positions exactly at the given times. Therefore a highly precise tracking is possible. Furthermore, a spline interpolation is done between the given sample points to make the movement smoother.
If the CABINET.OFFSET_MODE is not 0, the offsets set for both axes and the derotator will be applied to the trajectory points. It is possible to change the offsets during trajectory execution, allowing f.e. to center a tracked star.
The following functions are provided in this module:

| Name | Type | Access | Description |
|---|---|---|---|
| TYPE | INT | RO | Type identifier, see below |
| VERSION | STRING | RO | Spline engine version. Newer version may introduce new functionality |
| REVISION | STRING | RO | Spline engine revision. Newer revision will not introduce new functionality |
| BUFFER[] | STRUCT | — | Struct array which is used as buffer for new sample points (see below) |
| FREEBUFFER | INT | RO | Depending on the TYPE this is the number of entire buffers that can be added by GENERATE without blocking (TYPE 1) or the number of sample points in the buffer that can be added without blocking (TYPE 2). |

| Name | Type | Access | Description |
|---|---|---|---|
| GENERATE | INT | WO | Generates a spline of the first $n$ sample point entries of the BUFFER, where $n$ is the number written to the variable and adds this new spline to the internal trajectory buffer. The Callback will only return if all sample points have been written to the trajectory buffer or fail if no buffers are available. Depending on the TYPE this will almost happen instantaneous with TYPE 1 and may take some time with TYPE 2 (see below). |
| EXECUTE | INT | RW | By writing 1 to this variable, the execution of the precalculated splines is started. This function will only return if the internal spline buffer is empty or the spline execution was aborted. During the execution, more splines can be generated from new sample points (until the spline buffer is full, see below) to allow continuous tracking over longer periods of time. By setting the variable to 0, the execution is aborted. To check if a trajectory is executing, the variable can also be read, returning 1 or 0. |
| CLEAR | INT | WO | Clear the entire sample point buffer on writing 1. |
| RESET | INT | WO | Discard all sample points and splines and stop execution on writing 1. |
| INTERPOLATION_MODE | INT | RW | Method which is used for the interpolation between sample points (see below). |
| INTERPOLATION_ORDER | INT | RW | In case of spline interpolation, the maximal order of the used polynoms. |

The following trajectory module types are currently defined:

| Type no | Description |
|---|---|
| 0 | No trajectory support |
| 1 | Trajectory support for azimuth, zenith distance and derotator. Buffer data must overlap by at least INTERPOLATION_ORDER data points. |

| Type no | Description |
|---|---|
| 2 | Trajectory support for azimuth, zenith distance and derotator. Buffer need not to overlap. |

The struct array for the trajectory buffer has the following entries:

| Name | Type | Access | Description |
|---|---|---|---|
| TIME | FLOAT | RW | Time (Seconds since 01.01.1970 with fractions after the decimal point) |
| AZ | FLOAT | RW | Azimuth position (in degrees, see section 3.3 and A) |
| ZD | FLOAT | RW | Zenith distance (in degrees, see section 3.3 and A) |
| DEROT | FLOAT | RW | Derotator position (in degrees, see section 3.6 and A) |

The following interpolation modes are currently defined:

| Type no | Description |
|---|---|
| 0 | No interpolation available. |
| 1 | Spline interpolation, INTERPOLATION_ORDER can be used to configure the maximum order of the used polynoms. |

# A Coordinate Systems of the Telescope Axes

An overview of the coordinate systems on all telescope axes is given in fig. 1.
The top left image shows the telescope from above. The azimuth zero position is in the south (northern hemisphere) or in the north (southern hemisphere). The telescope will look in that direction for a zenith distance of +90 degrees. The azimuth angle is counted clockwise on both hemispheres.
In the top right image one can see a side view of the telescope. To distinguish the two directions of the zenith distance (in case of a telescope capable of moving through the zenith) the telescope chain is used. If the telescope is looking in the same direction as the chain, the zenith distance is defined positive. Zero is always defined to be in the zenith.
The two images in the middle show the derotator as seen when looking toward the flange on the fork. The angle for the derotator is counted in the same direction as for the zenith distance, therefore the direction differs on the both sides. Zero is always defined to be on the top.
The lower image shows the coordinate system for the focus motor. Zero is here

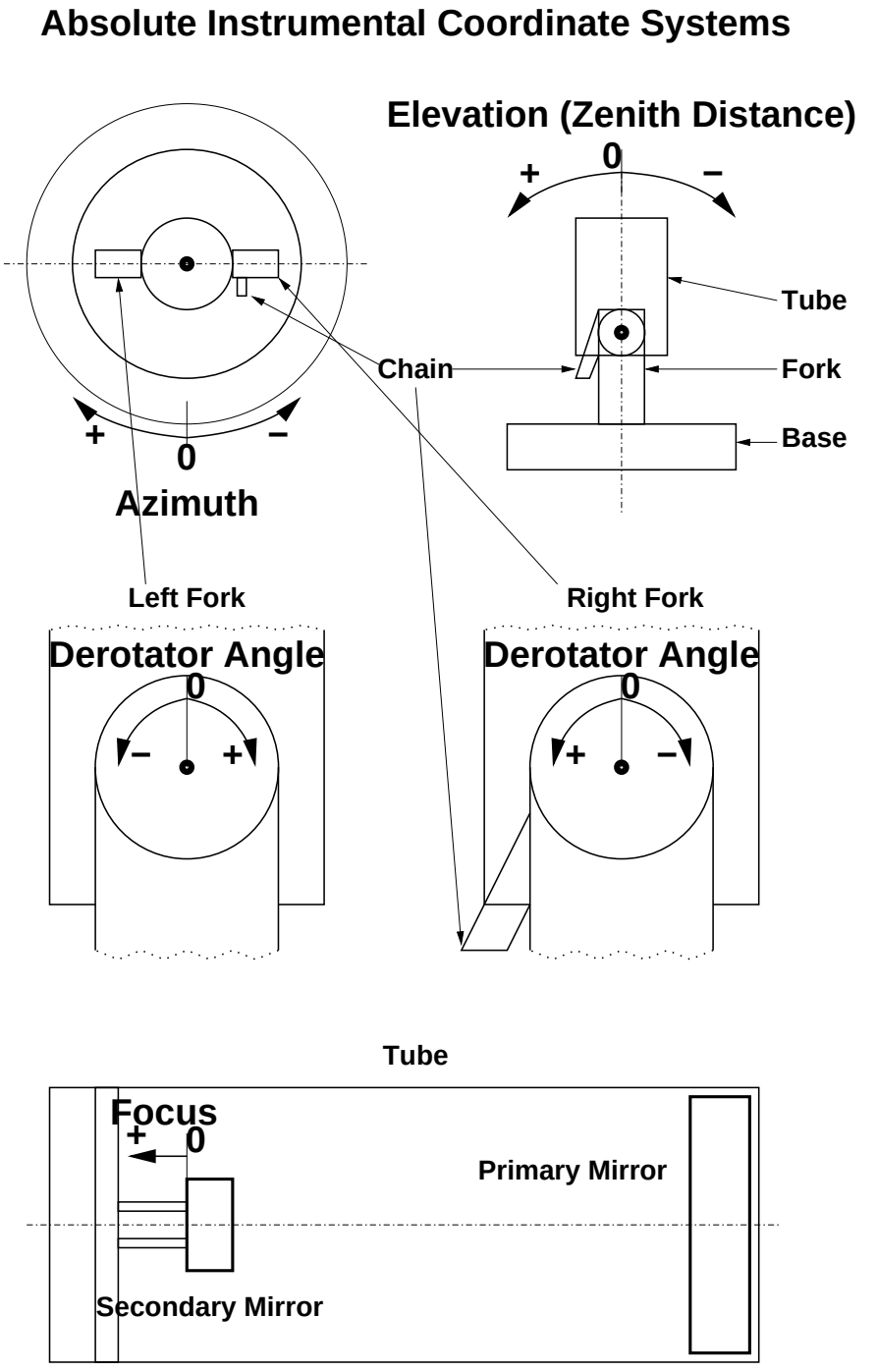


***Figure 1:*** *Overview of the coordinate systems on all telescope axes*

defined to be the smallest, by hardware restraints, possible distance between both mirrors. The focus position is then always positive, giving the additional distance in millimeters.

# B List of RS Error Symbols

The cabinet software uses a bit field to record errors that have occurred. This bit field can be retrieved using `CABINET.STATUS.LATCH` and `CABINET.STATUS.CURRENT` (see section 3.4). The bits are numbered from right (lowest bit) to left. By reading the corresponding array element from `CABINET.STATUS.SYMBOLS[]` it is possible to get a symbolic name of the error. A list of all possible errors together with their number and their meaning is given in the remainder of this section. (To identify an error, only the symbolic name should be used as the number may change in future releases!)
For most errors there is a classification of the severity, as well as a recommendation of what to do, if the error occurs:

| Class | Description |
|---|---|
| [I] | Information message, usually no user intervention necessary. |
| [W] | Warning message, usually the system will continue to operate, but with some limitations. |
| [E] | Error message, the whole system is halted and put in a safe condition. |
| [P] | Panic message, the whole system is halted and put in a safe condition. |

| Short | Recommendation |
|---|---|
| (C) | Clear the error, after the reason for the error (f.e. the too large coordinate in case of a soft limit condition) has been resolved. |
| (R) | If possible, resolve the reason for the error and restart the entire system afterwards. If the error occurs again, ask for support. |
| (S) | Ask for support, as this error is not supposed to occur in a release version. |

Some errors (like the limit switch errors for azimuth and elevation) should never occur during normal operation, and have therefore a "(R)" recommendation. However, when the telescope is moved by hand while the power is off, this error can be cleared without a system reboot.

| No | Text | Description |
|---|---|---|
| 0 | I_MOD_INIT | (S) Module initialisation, [should never occur] |
| 1 | I_MOD_ERR_OVERFLOW | (S) Not all modules logged in the shared memory area [should never occur] |
| 2 | E_TEST_ERROR | [E](S) Software test error message [should never occur] |
| 3 | W_TEST_WARNUNG | [W](S) Software test warning message [should never occur] |
| 4 | E_TEST_PANIK | [P](S) Software test panic message [should never occur] |
| 5 | ERR_Main_bad_state | [E](R) Internal software error, wrong state detected |
| 6 | ERR_Main_Debug_Mode | [W](R) Internal software error, debug mode |
| 7 | ERR_Dlg_not_running | [P](R) The dialog program is not running |
| 8 | ERR_Power_ON_OFF | [I] The power was turned on or off in the manual mode |
| 9 | ERR_Power | [W](C) The software intentionally turned off the power (CABINET.POWER=0) |
| 10 | ERR_EmergencyOff_Fuse_F161 | [P](R) Fuse F161 for the emergency off circuit failed |
| 11 | ERR_EmergencyOff_A311 | [E](C) The emergency off switch was pressed on either the cabinet or the bottle |
| 12 | ERR_USV_Power_Lost | [P](C) The uninterruptible power source detected a power failure of main power and took over. Some subsystems, f.e. the ETEL drives, will not be powered by the UPS. |
| 13 | ERR_USV_Battery_alarm_G121 | [W](R) The uninterruptible power source reported a battery alarm condition |
| 14 | ERR_Phase_Watcher_Fuse_F101 | [W](R) Fuse F101 of the phase watcher failed |
| 15 | ERR_Phase_Watcher_N101 | [E](R) The direction of the rotary current is wrong or there was over- or undervoltage detected |
| 16 | ERR_Fuse_F122 | [P](R) Fuse F122 failed |
| 17 | ERR_Bottel_Fuse_F168 | [W](R) Fuse F168 for the bottle failed |
| 18 | ERR_ETEL_MainPower_Fuse_Q102 | [W](R) Fuse Q102 for the main power to the telescope axes failed |

| No | Text | Description |
|---|---|---|
| 19 | ERR_ETEL_Fan_Fuse_F612 | [W](C) Fuse F612 for the cooling fans to the motor controllers failed |
| 20 | ERR_climate_control_Fuse_F201 | [W](C) Fuse F201 for the cabinet cooling fan failed |
| 21 | ERR_cabinet_heating_Fuse_F204 | [W](C) Fuse F204 for the cabinet heating system failed |
| 22 | ERR_teleskop_socket_Fuse_F205 | [W](R) Fuse F205 failed |
| 23 | ERR_oil_pump_Fuse_Q411 | [W](R) Fuse Q411 for the oil pump failed |
| 24 | ERR_oil_extractionpump_Fuse_Q431 | [W](C) Fuse Q431 for the oil extraction pump failed |
| 25 | ERR_oil_heating_Fuse_Q412 | [W](C) Fuse Q412 for the oil heater failed |
| 26 | ERR_oil_bad_state | [P](R) Internal software error, wrong state detected |
| 27 | ERR_oil_Level | [W](C) Oil level is not okay |
| 28 | ERR_oil_Level_low | [W](C) Oil level is too low |
| 29 | ERR_oil_Level_high | [W](C) Oil level is too high |
| 30 | ERR_oil_Temperature_low | [W](C) Oil temperature is too low |
| 31 | ERR_oil_Temperature_high | [W](C) Oil temperature is too high |
| 32 | ERR_oil_Filter | [W](C) Oil filter is clogged |
| 33 | ERR_oil_Filter_status | [W](C) Oil filter is clogged |
| 34 | ERR_oil_Pressure | [W](C) Oil pressure too low during operation |
| 35 | ERR_oil_Pressure_TIMEOUT | [W](R) Oil pressure did not reach correct level within the timeout period (during system power on) |
| 36 | ERR_cover_Fuse_F165 | [W](R) Fuse F165 for the cover motor(s) failed |
| 37 | ERR_cover1_bad_state | [P](R) Internal software error, wrong state detected |
| 38 | ERR_cover1_TIMEOUT | [W](C) Cover motor 1 did not reach the position within the expected time |
| 39 | ERR_cover2_bad_state | [P](R) Internal software error, wrong state detected |
| 40 | ERR_cover2_TIMEOUT | [W](C) Cover motor 2 did not reach the position within the expected time |
| 41 | ERR_cover3_bad_state | [P](R) Internal software error, wrong state detected |
| 42 | ERR_cover3_TIMEOUT | [W](C) Cover motor 3 did not reach the position within the expected time |

| No | Text | Description |
|---|---|---|
| 43 | ERR_astrodome_Fuse_Q451 | [W](R) Fuse Q451 for the control of the dome motor(s) failed |
| 44 | ERR_astrodome1_bad_state | [P](R) Internal software error, wrong state detected |
| 45 | ERR_astrodome1_TIMEOUT | [W](C) Dome motor 1 did not reach the position within the expected time |
| 46 | ERR_astrodome2_bad_state | [P](R) Internal software error, wrong state detected |
| 47 | ERR_astrodome2_TIMEOUT | [W](C) Dome motor 2 did not reach the position within the expected time |
| 48 | ERR_Focus_Fuse_F163 | [W](R) Fuse F163 for the focus motor failed |
| 49 | ERR_BFocus_bad_state | [P](R) Internal software error, wrong state detected |
| 50 | ERR_Focus_bad_state | [P](R) Internal software error, wrong state detected |
| 51 | ERR_Focus_CONTR | [W](R) The focus motor controller returned an error |
| 52 | ERR_Focus_Referenze | [W](R) The reference mark of the focus drive was not found within the expected time |
| 53 | ERR_Focus_LimitSwitch_left | [W](R) The focus drive reached the left hardware limit switch |
| 54 | ERR_Focus_LimitSwitch_right | [W](R) The focus drive reached the right hardware limit switch |
| 55 | ERR_Focus_Soft_Limit_MAX | [W](C) The given target position for the focus was too large |
| 56 | ERR_Focus_Soft_Limit_MIN | [W](C) The given target position for the focus was too small |
| 57 | ERR_Derotator_Fuse_F166 | [W](R) Fuse F166 for the derotator motor failed |
| 58 | ERR_BDerotator_bad_state | [P](R) Internal software error, wrong state detected |
| 59 | ERR_Derotator_bad_state | [P](R) Internal software error, wrong state detected |
| 60 | ERR_Derotator_CONTR | [W](R) The derotator motor controller returned an error |
| 61 | ERR_Derotator_Referenze | [W](R) The reference mark of the derotator drive was not found within the expected time |

| No | Text | Description |
|---|---|---|
| 62 | ERR_Derotator_LimitSwitch_left | [W](R) The derotator drive reached the left hardware limit switch |
| 63 | ERR_Derotator_LimitSwitch_right | [W](R) The derotator drive reached the right hardware limit switch |
| 64 | ERR_Derotator_Soft_Limit_MAX | [W](C) The given target position for the derotator was too large |
| 65 | ERR_Derotator_Soft_Limit_MIN | [W](C) The given target position for the derotator was too small |
| 66 | ERR_color_wheel_Fuse_F164 | [W](R) Fuse F164 for the filter wheel failed |
| 67 | ERR_color_wheel_bad_state | [P](R) Internal software error, wrong state detected |
| 68 | ERR_color_wheel_timeout_drive | [W](R) The filter wheel did not pass the next filter index within the expected time |
| 69 | ERR_color_wheel_Referenze | [W](R) The reference mark of the filter wheel was not found within the expected time |
| 70 | ERR_color_wheel_Ref_mismatch | [W](R) Wrong count of index marks counted at the reference mark |
| 71 | ERR_color_wheel_unreachable | [W](C) The selected filter index was too large |
| 72 | ERR_no_Interpolation_Points | [W](C) The trajectory ran out of sample points and therefore, the trajectory execution stopped |
| 73 | ERR_Extrapolation_Points | [W](C) ??? The interpolation engine discovered a gap in the sample points and had to extrapolate points |
| 74 | ERR_mirror_stop_Fuse_F167 | [W](R) ??? Fuse F167 for the mirror stop failed |
| 75 | ERR_mirror_stop_bad_state | [P](R) Internal software error, wrong state detected |
| 76 | ERR_mirror_stop_TIMEOUT | [W](C) ??? The mirror stop did not reach the position within the expected time |
| 77 | ERR_mirror_rotation_Fuse_F168 | [W](R) Fuse F168 for the mirror rotator failed |
| 78 | ERR_mirror_rotation_bad_state | [P](R) Internal software error, wrong state detected |
| 79 | ERR_mirror_rotation_CONTR | [W](R) The mirror rotator controller returned an error |

| No | Text | Description |
|---|---|---|
| 80 | ERR_mirror_rotation_Referenze | [W](R) The reference mark of the mirror rotator was not found within the expected time |
| 81 | ERR_mirror_rotation_LimitSwitch_left | [W](R) The mirror rotator reached the left hardware limit switch |
| 82 | ERR_mirror_rotation_LimitSwitch_right | [W](R) The mirror rotator reached the right hardware limit switch |
| 83 | ERR_mirror_rotation_Soft_Limit_MAX | [W](C) The given target position for the mirror rotator was too large |
| 84 | ERR_mirror_rotation_Soft_Limit_MIN | [W](C) The given target position for the mirror rotator was too small |
| 85 | ERR_GPS_Uart_TX_Error | [W](C) An error occurred while sending data to the GPS receiver |
| 86 | ERR_GPS_Uart_RX_Error | [W](C) An error occurred while reading data from the GPS receiver |
| 87 | ERR_GPS_Error | [W](C) Communication error with the GPS receiver |
| 88 | ERR_GPS_Satellites_too_few | [W](C) The GPS receiver did not receive enough satellites for locking onto the GPS time during initialisation |
| 89 | ERR_GPS_Satellites_lost | [W](C) The GPS receiver lost the lock on the GPS time |
| 90 | ERR_GPS_Leap_second | [W](C) There was a leap second announcement received. However, the leap second will not be applied until system restart |
| 91 | ERR_Clock_Tes_Error | [W](C) Internal clock error |
| 92 | ERR_Clock_Puls_Error | [W](C) Wrong or missing clock pulse from the GPS receiver |
| 93 | ERR_Clock_Regulator_Limit | [W](C) The clock regulator was unable to keep the clock within the defined offset to the GPS time |
| 94 | ERR_Clock_SysClk_Error | [W](C) Internal system clock error |
| 95 | ERR_ETEL_Power_Fuse_F610 | [W](R) The Fuse F610 for the main power to the ETEL drives failed |
| 96 | ERR_ETEL_Control_Fuse_F611 | [W](R) The Fuse F611 for the power to the ETEL motor controllers failed |
| 97 | ERR_ETEL_Move_Power_up | [W](C) One ETEL axis was moving during power up |

| No | Text | Description |
|---|---|---|
| 98 | ERR_ETEL_Profibus_FEHLER | [W](R) The ETEL controller did not answer on the profibus within the expected time |
| 99 | ERR_azm_FAULT | [W](C) The azimuth axis controller raised an error message and the axis stopped moving |
| 100 | ERR_azm_ALARM | [W](C) The azimuth axis controller raised a warning message but the axis will usually continue moving |
| 101 | ERR_azm_TIMEOUT | [W](R) The azimuth axis controller did not respond to a request within the expected time |
| 102 | ERR_azm_Ref_FEHLER | [W](R) An error occurred during the referencing process of the azimuth axis |
| 103 | ERR_azm_Parm_FEHLER | [W](R) An error occurred during the readout of needed parameters from the azimuth axis controller |
| 104 | ERR_azm_Etel_FEHLER | [W](R) The azimuth axis controller is not in the correct state for the requested operation |
| 105 | ERR_azm_bad_state | [P](R) Internal software error, wrong state detected |
| 106 | ERR_azm_LimitSwitch_I | [W](R) The azimuth axis reached the hardware limit switch number one |
| 107 | ERR_azm_LimitSwitch_II | [W](R) The azimuth axis reached the hardware limit switch number two |
| 108 | ERR_azm_Soft_Limit_MAX | [W](C) The given target position for the azimuth axis was too large |
| 109 | ERR_azm_Soft_Limit_MIN | [W](C) The given target position for the azimuth axis was too small |
| 110 | ERR_Azm_bad_state | [P](R) Internal software error, wrong state detected |
| 111 | ERR_BAzm_bad_state | [P](R) Internal software error, wrong state detected |
| 112 | ERR_Azm_TopInstanze | [W](R) Internal state machine error |
| 113 | ERR_Azm_Oil | [W](C) The azimuth axis oil pump is not ready |
| 114 | ERR_elv_FAULT | [W](C) The elevation axis controller raised an error message and the axis stopped moving |

| No | Text | Description |
|---|---|---|
| 115 | ERR_elv_ALARM | [W](C) The elevation axis controller raised a warning message but the axis will usually continue moving |
| 116 | ERR_elv_TIMEOUT | [W](R) The elevation axis controller did not respond to a request within the expected time |
| 117 | ERR_elv_Ref_FEHLER | [W](R) An error occurred during the referencing process of the elevation axis |
| 118 | ERR_elv_Parm_FEHLER | [W](R) An error occurred during the readout of needed parameters from the elevation axis controller |
| 119 | ERR_elv_Etel_FEHLER | [W](R) The elevation axis controller is not in the correct state for the requested operation |
| 120 | ERR_elv_bad_state | [P](R) Internal software error, wrong state detected |
| 121 | ERR_elv_LimitSwitch_I | [W](R) The elevation axis reached the hardware limit switch number one |
| 122 | ERR_elv_LimitSwitch_II | [W](R) The elevation axis reached the hardware limit switch number two |
| 123 | ERR_elv_Soft_Limit_MAX | [W](C) The given target position for the elevation axis was too large |
| 124 | ERR_elv_Soft_Limit_MIN | [W](C) The given target position for the elevation axis was too small |
| 125 | ERR_Elv_bad_state | [P](R) Internal software error, wrong state detected |
| 126 | ERR_BElv_bad_state | [P](R) Internal software error, wrong state detected |
| 127 | ERR_Elv_TopInstanze | [W](R) Internal state machine error |
| 128 | ERR_Elv_Oil | [W](C) The elevation axis oil pump is not ready |
| 129 | ERR_Hilscher_Error | [W](R) The profibus controller could not be initialized correctly or didn't respond |
| 130 | ERR_Hilscher_Profibus | [W](R) The profibus communication failed |
| 131 | ERR_bad_type | [W](R) Internal software error, invalid, not implemented type used |
| 132 | ERR_Break_Error | [W](R) One brake did not work correctly |
| 133 | ERR_Break_Timeout | [W](R) One brake did not respond within the expected time |

| No | Text | Description |
|---|---|---|
| 134 | ERR_Axis_Mechanic_Blocked | [?](?) One axis is not movable due to a mechanic problem |
| 135 | ERR_Motor_has_bad_Polarity | [?](?) One axis motor is connected with the wrong polarity |
| 136 | ERR_Temp_Cabinet_HOT | [W](C) The temperature in the control cabinet is too high |
| 137 | ERR_Temp_Cabinet_COLD | [W](C) The temperature in the control cabinet is too low |
| 138 | W_RTMSTART | [I](C) The realtime module did start |
| 139 | W_RTMSTOP | [I](C) The realtime module did stop |
| 140 | W_DLGSTART | [I](C) The dialog program did start |
| 141 | W_DLGSTOP | [I](C) The dialog program did stop |
| 142 | W_TPLSTART | [I](C) The TCI TPL-Server did start |
| 143 | W_TPLConnectionLost | [W](C) A connection to the TCI TPL-Server was closed unexpectedly |
| 144 | W_TPLNewConnection | [I](C) A new connection was opened to the TCI TPL-Server |
| 145 | W_TPLSTOP | [W](C) The TPL-Server did stop |
| 146 | W_TPLTEST | [W](C) Warning for test purposes only |
| 147 | W_TPLREBOOT | [W](C) A reboot of the system was requested via TPL |
| 148 | W_TPLREBOOT_IGN | [I](C) A reboot of the system that was requested via TPL was ignored |
| 149 | W_DEFVAR | [W](S) The system could not find correct parameters and used a default set |
| 150 | W_BKPPAR | [W](S) The system could not find correct parameters and used a backup set |
| 151 | W_DEFPAR | [P](S) The system could not find correct parameters and used a default set |
| 152 | W_SYS_REBOOT | [P](C) A reboot of the system was initiated |
| 153 | W_SYS_HALT | [P](C) A halt of the system was initiated |
| 154 | ERR_not_def | [?](C) Default error message, not used |

# C Undocumented Modules

## C.1 Module POINTING

This module provides functions to point the telescope to RA / DEC positions and is only available for the DIMM telescopes. It will be superseded by the much more advanced TSI functions and therefore removed in later releases.
The following functions are provided in this module:

| Name | Type | Access | Description |
|---|---|---|---|
| TYPE | INT | RO | Type identifier, see below |
| VERSION | STRING | RO | Hardware version. Newer version may introduce new functionality |
| REVISION | STRING | RO | Hardware revision. Newer revision will not introduce new functionality |
| MODE | INT | RW | Pointing mode (0=internal, 1=use TRAJECTORY internally). |
| TARGET.RA | FLOAT | RW | Target RA. |
| TARGET.DEC | FLOAT | RW | Target DEC. |
| TARGETDISTANCE | FLOAT | RO | RMS Distance of all axes from the target. |
| TRACK | INT | RW | Start or stop tracking on RA/DEC position. Values are 0 to stop tracking; 1 to start tracking with positive zenith distance; 2 to start tracking with dead zone optimization, that is, if the telescope is capable of moving the second axis through the zenith, the software will use either negative or positive zenith distance values to avoid the azimuth position to run into a limit switch; 3 to start tracking with negative zenith distance; 4 to select automatically the zenith distance to minimize slewing time. After writing 2 or 4 the variable will either be 1 or 3 on readout, indicating which range was chosen. |
| POINTINGPARAMS | STRUCT | — | Contains pointing model coefficients (see below) |
| REFRACTION | STRUCT | — | Refraction correction settings. |

The following pointing types are currently defined:

| Type no | Description |
|---|---|
| 0 | Not available |
| 1 | Extra-solar object tracking only. |

### C.1.1 Pointing model parameters

This structure holds all pointing model coefficients. Unless noted otherwise, the units are degrees. The used pointing correction model is the standard geometrical model with 6 coefficients for mounting errors plus one for sagging of the tube due to gravitation. These coefficients can be calculated using standard numerical fitting technics or additional programs (e.g. TPoint) and are set by default to 0.
The following equations are used to calculate correction for azimuth and zenith axis:

$$\begin{aligned} d_{\mathrm{A}} &= \sin A \cdot \cot Z \cdot c_{\mathrm{AN}} \\ &- \cos A \cdot \cot Z \cdot c_{\mathrm{AE}} \\ &+ \cot Z \cdot c_{\mathrm{NPAE}} \\ &- \frac{c_{\mathrm{CA}}}{\sin Z} \\ &+ c_{\mathrm{AOFF}} \\ d_{\mathrm{Z}} &= \cos A \cdot c_{\mathrm{AN}} \\ &+ \sin A \cdot c_{\mathrm{AE}} \\ &+ \sin Z \cdot c_{\mathrm{FLEX}} \\ &+ c_{\mathrm{ZOFF}} \end{aligned}$$

When pointing to RA/DEC values, the `POINTING` module will do the following steps:

1. Convert RA/DEC to AZ/ZD
2. apply refraction (if enabled)
3. apply pointing corrections as described above
4. add axis offsets if enabled by `CABINET.OFFSET_MODE`

The following values can be modified:

| Name | Type | Access | Description |
|---|---|---|---|
| `AOFF` | FLOAT | RW | Absolut azimuth encoder offset. |
| `ZOFF` | FLOAT | RW | Absolut zenith distance encoder offset. |
| `AE` | FLOAT | RW | Azimuth axis tilt of vertical to NORTH. |
| `AN` | FLOAT | RW | Azimuth axis tilt of vertical to EAST. |

| Name | Type | Access | Description |
|---|---|---|---|
| NPAE | FLOAT | RW | Zenith distance axis not perpendicular to azimuth axis. |
| CA | FLOAT | RW | Telescope beam not perpendicular to zenith distance axis. |
| FLEX | FLOAT | RW | Flexure term. |

### C.1.2 Refraction parameters

The following values can be modified to control the refraction compensation:

| Name | Type | Access | Description |
|---|---|---|---|
| MODE | INT | RW | 0 - no refraction correction, 1 - use defaults for refraction (10 °C, 1010 mbar, corrected by height), 2 - use customized settings |
| TEMPERATURE | FLOAT | RW | Outside temperature in Celsius (only for mode 2). |
| PRESSURE | FLOAT | RW | Air pressure in mbar (only for mode 2). |
| HEIGHT | FLOAT | RW | Height above sea level in meters (only for mode 1). |

## C.2 Module SIMPLE_FOCUS

This module is provided as a quick patch for the DIMM telescope to allow unregulated Focus movements and will be removed in the future. The following functions are provided in this module:

| Name | Type | Access | Description |
|---|---|---|---|
| STEP | FLOAT | WO | Switch power ON for the given value(in seconds), then stop the motor by switching the power OFF. If value is negative, the focus will run towards the inner position, else it will run to the outer position. |
| CURRPOS | FLOAT | RO | Since the Focus does not have a encoder, this will be 0 if it is in the innermost position, 1 if it is in the outermost position and 0.5 if somewhere in between. |

# D Abbreviations

| Abbreviation | Description |
|---|---|
| TPL | Transfer Protocol Language |
| TCC | Telescope Control Computer, computer in cabinet, running the TCS |
| TCI | Telescope Control Interface, connection between TCS and TSS |
| TCS | Telescope Control System, software running on TCC |
| TSC | Telescope Software Computer, computer close to the telescope running TSS |
| TSI | Telescope Software Interface, connection between TSS and user applications interface or applications |
| TSS | Telescope Software System, software running on TSC |

# References

[1] M. Ruder and D. Plasa. *TPL2, Transfer Protocol Language, V2 — A protocol for client-server based exchange of data and commands over a TCP/IP network connection.* 4pi systeme GmbH. 4PI-DOC-03-008-01.