



Mount Command Protocol

Software version 3.1.10

Prepared by

Company

Date

Number of pages

Filippo Riccio

COMEC s.n.c.

2022-10-03

79

Index

Introduction.....	2
Communication.....	3
Serial communication.....	3
TCP/IP communication.....	3
Commands.....	3
Alignment Commands.....	4
GPS commands.....	4
Sync Control and Model Building.....	5
Get Information.....	9
Home & Park Commands.....	24
Movement Commands.....	25
Rate Commands.....	30
Set Commands.....	32
Tracking Commands.....	40
Precision Toggle.....	43
Drive configuration.....	43
Dome Control.....	45
Axis angular position commands.....	48
Dithering Commands.....	50
Weather Station Commands.....	51
Satellite Orbital Elements Commands.....	52
Network Discovery and Web Interface Commands.....	54
Other Commands.....	55
Extended LX200 emulation and precision.....	56
Notes on pointing objects.....	56
Notes on pointing state.....	57
Notes on building an alignment model.....	59
Entering an alignment model.....	60
Assisted polar axis alignment and orthogonality correction.....	60
Polar axis alignment.....	60
Orthogonality axis alignment.....	61
Escaped strings.....	61
Example of a HyperTerminal Session.....	62

Introduction

This paper documents the Command Protocol used to remotely command and control the 10micron mounts, with software version 3.1.10.

The protocol is derived from the Meade LX200™ Serial Command Protocol in order to simplify operation with existing software. Many extensions are used to provide functionality not available with the standard LX200 protocol. Some commands are compatible with the Astro-Physics™ serial command protocol, and a few of them behave differently when used in different emulation modes (see the *Extended LX200 emulation and precision* paragraph).

Communication

The communication can be established using a serial connection or a TCP/IP connection.

Serial communication

The communication requires a serial line at 9600 bps, 8 bits per character, 1 stop bit and no parity. The device controlling the mount must initiate all communications, i.e. the mount will not send anything except as a response to a query.

TCP/IP communication

The mount must be configured with an IP address (see the user's manual). The device controlling the mount must connect to that IP address, port 3490 or 3492. Once the connection is established, the device controlling the mount must initiate all communications, i.e. the mount will not send anything except as a response to a query. Until software version 2.9.9, only one connection on each port at a given time is allowed and the mount will refuse further incoming connections. From 2.9.10 onwards, up to ten independent connections on each port are allowed.

Commands

All commands are case-sensitive. Each command is a string of characters beginning with “:” and ending with “#”, with the only exception of the <ACK> command. A single “#” character clears the receiving buffer.

The commands evidenced in blue or green are extensions to the standard LX200 protocol. If part of a command's explanation is evidenced in blue or green, that part is an extension to the behaviour of the command with the standard LX200 protocol.

The characters evidenced in red should be replaced with the appropriate values to be communicated to the mount.

The meaning of the asterisk (*) character changes accordingly to the emulation and precision mode set:

Emulation and precision	Meaning of the asterisk (*) character
LX200 emulation, low or high precision	ascii code 223 (0xDF)
Extended emulation, low or high precision	ascii code 42 (0x2A) (ASCII asterisk "*")
Any emulation, ultra precision	Not used in communications from the mount to the controller, both 0xDF and 0x2A accepted in communications from the controller to the mount.

Except when stated otherwise, all equatorial coordinates used in the communication protocol are local topocentric coordinates.

Alignment Commands

In the LX200 protocol, the alignment commands are used to set or query the tracking mode of the mount; the so-called “Land” mode corresponds to no tracking, while “Polar” and “AltAz” mode correspond tracking in one or both axes.

Since the 10micron mounts do not support different mounting modes, these commands are used only to start and stop tracking.

<ACK>	Query of alignment mounting mode, the command string is the single character “0x06”.
	Returns:
L	If tracking is off
P	If tracking is on

:AL#	Stops tracking.
	Returns: nothing

:AP#	Starts tracking.
	Returns: nothing
	Note: from version 2.12.26 to version 2.15.34 this command does not start tracking when the mount has been stopped with the :STOP# command or by pressing the STOP key on the keypad. A workaround to this is to send ":AL#:AP#" in order to start tracking.

GPS commands

:gT#	Updates the clock, latitude, longitude and elevation data from the GPS. Until firmware version 2.12.5, this command may take several minutes to complete. During the process, the mount does not respond to other commands. From version 2.12.6 onwards, this command is asynchronous
	Returns:
0	if the user interrupts the process, or the GPS is unconnected, or some other error occurs
1	if the process has been completed successfully
	From firmware version 2.12.6 onwards, this command configures the mount to use data from a GPS module and returns immediately.
	Returns:
0	if the GPS module hasn't communicated all data yet
1	if the GPS module has communicated all data and the mount is synchronized

:gps#	Returns the last string from the GPS in standard NMEA format, if available, followed by a '#' character.
-------	--

:gtg#	Checks if the mount clock is being kept synchronized to the clock of the connected GPS module.
	Returns:

- "0#" if the mount clock is not being kept synchronized to the GPS clock.
 "1#" if the mount clock is being kept synchronized to the GPS clock.

Sync Control and Model Building

:CMS# Adds a alignment point to the current model, by synchronizing the position of the mount with the coordinates of the currently selected target and recalculating the alignment model.

Returns:
 the string "V#" if successful
 the string "E#" if the model can't be refined
 Available from version 2.8.15.

:CM# Synchronizes the position of the mount with the coordinates of the currently selected target.

Starting with version 2.8.15, this command has two possible behaviours depending on the value passed to the last :CMCFGn# command. By default after startup, or after the :CMCFG0# command has been given, the synchronization works by offsetting the axis angles. If the :CMCFG1# command has been given, it works like the :CMS# command, but returning the strings below.

Returns:
 the string "Coordinates matched #" if the coordinates have been synchronized
 the string "Match fail: dist. too large#" if the coordinates have not been synchronized

:CMR# The same as :CM#.

:CMCFGn#

Configures the behaviour of the :CM# and :CMR# commands depending on the value of n. If n=0, the commands :CM# and :CMR# work in the default mode, i.e. they synchronize the position of the mount with the coordinates of the currently selected target by correcting the axis offset values. If n=1, the commands :CM# and :CMR# work by using the synchronization position as an additional alignment star for refining the alignment model.

Returns:
 the string "0#" if the value 0 has been passed
 the string "1#" if the value 1 has been passed
 Available from version 2.8.15.

:getalst# Gets the number of alignment stars used in the current alignment model.

Returns:
 the number of alignment stars terminated by '#'.
 Available from version 2.8.15.

:delalig# Deletes the current alignment model and stars.

Returns: an empty string terminated by '#'.
 Available from version 2.8.15.

:getaliN# Gets the alignment information for star number N in the alignment model, where N goes from 1 to the number returned by :getalst#.

Returns:

the string "E#" if N is out of range

otherwise a string formatted as follows

"HH:MM:SS.SS,+dd*mm:ss.s,eeee.e#" where HH:MM:SS.SS is the hour angle of the alignment star in hours, minutes, seconds and hundredths of second, +dd*mm:ss.s is the declination of the alignment star in degrees, arcminutes, arcseconds and tenths of arcsecond, eeee.e is the error between the star and the alignment model in arcseconds.

Available from version 2.8.15.

:getain# Gets alignment information about the current alignment model.

Returns:

The string "E#" if there are less than two stars in the alignment; otherwiser, a string formatted as follows:

"ZZZ.ZZZZ,+AA.AAAA,EE.EEEE,PPP.PP,+OO.OOOO,+aa.aa,+bb.bb,NN,RRRRR.R#" where ZZZ.ZZZZ and +AA.AAAA are the azimuth and altitude of the direction pointed at by the right ascension axis, EE.EEEE is the polar align error in degrees and decimals, PPP.PP is the position angle of the right ascension axis with respect to the celestial pole in degrees and decimals, +OO.OOOO is the orthogonality error between the optical axis of the telescope and the declination axis in degrees and decimals, +aa.aa is the number of turns of the azimuth adjustment knobs (positive if the polar axis must be moved to the left), +bb.bb is the number of turns of the altitude adjustment knob (positive if the polar axis must be moved downwards), N is the number of terms used in the modeling, RRRRR.R is the expected RMS error in arcseconds.

Some values can be omitted and replaced with the string "E" if they aren't calculated: namely, the RMS and number of terms in case of less than 4 stars or QCI mounts, the orthogonality error in case of less than 3 stars, the polar align error and position angle in case of altazimuth mounts, the number of turns of the knobs in case of altazimuth mounts or latitude greater than 80 degrees.

Available form version 2.14.20.

Note: a bug present up to version 2.15.17 can prevent this command from returning the correct information when "Align Info" hasn't been shown on the keypad. You can use the following workaround:

- send the :newalig# command; this will not clear the current alignment, only start a new sequence of alignment points – the mount answers with V#;
- send the :endalig# command; this will not finalize the new sequence of alignment points, since none has been added, but it will force recalculation of the correct information for :getain# - the mount answers with E#;

At this point :getain and :getalpt return the correct values.

The bug has been corrected in version 2.15.18.

:getalpN# Gets the alignment information for star number N in the alignment model, where N goes from 1 to the number returned by :getalst#, including the polar angle of the measured star coordinates with respect to the modeled star coordinates.

Returns:

the string "E#" if **N** is out of range

otherwise a string formatted as follows

"HH:MM:SS.SS,+dd*mm:ss.s,eeee.e,ppp#" where HH:MM:SS.SS is the hour angle of the alignment star in hours, minutes, seconds and hundredths of second (from 0h to 23h59m59.99s), +dd*mm:ss.s is the declination of the alignment star in degrees, arcminutes, arcseconds and tenths of arcsecond, eeee.e is the error between the star and the alignment model in arcseconds, ppp is the polar angle of the measured star with respect to the modeled star in the equatorial system in degrees from 0 to 359 (0 towards the north pole, 90 towards east).

Available from version 2.8.15.

Note: a bug present up to version 2.15.17 can prevent this command from returning the correct information when "Align Info" hasn't been shown on the keypad. You can use the following workaround:

- send the :newalig# command; this will not clear the current alignment, only start a new sequence of alignment points – the mount answers with V#;
- send the :endalig# command; this will not finalize the new sequence of alignment points, since none has been added, but it will force recalculation of the correct information for :getain# - the mount answers with E#;

At this point :getain and :getalpt return the correct values.

The bug has been corrected in version 2.15.18.

:delalst**N**# Deletes the alignment star **N**, where **N** goes from 1 to the number returned by :getalst#, and recalculates the alignment model.

Returns:

the string "0#" if the procedure has failed (this can happen due to **N** being out of range, or if the model can't be recalculated correctly with the remaining stars).

the string "1#" if the procedure has succeeded.

Available from version 2.8.15.

Note that the first two stars cannot be deleted in firmware version < 2.10.6.

:newalig# Start creating a new alignment specification, that will be entered with the :newalpt command.

Returns:

the string "V#" (this is always successful).

See also the paragraph *Entering an alignment model*.

Available from version 2.8.15.

:newalpt**MRA,MDEC,MSIDE,PRA,PDEC,SIDTIME**#

Add a new point to the alignment specification. The parameters are:

MRA – the mount-reported right ascension, expressed as **HH:MM:SS.S**

MDEC – the mount-reported declination, expressed as **sDD:MM:SS**

MSIDE – the mount-reported pier side (the letter 'E' or 'W', as reported by the :pS# command)

PRA – the plate-solved right ascension (i.e. the right ascension the telescope was effectively pointing to), expressed as **HH:MM:SS.S**

PDEC – the plate-solved declination (i.e. the declination the telescope was effectively pointing to), expressed as **sDD:MM:SS**

SIDTIME – the local sidereal time at the time of the measurement of the point, expressed as **HH:MM:SS.S**

Returns:

the string "**nnn#**" if the point is valid, where **nnn** is the current number of points in the alignment specification (including this one)

the string "**E#**" if the point is not valid

See also the paragraph *Entering an alignment model*.

Available from version 2.8.15.

:endalig# Completes the alignment specification and computes a new alignment from the given alignment points.

Returns:

the string "**V#**" if the alignment has been computed successfully

the string "**E#**" if the alignment couldn't be computed successfully with the current alignment specification. In this case the previous alignment is retained.

See also the paragraph *Entering an alignment model*.

Available from version 2.8.15.

:modelcnt# Returns the number of user alignment models saved in the mount.

Returns:

the string "**nnn#**", where **nnn** is the number of models available.

Available from version 2.13.3.

:modelnamN#

Returns the name of model **N** saved in the mount, from 1 to the number returned by command **:modelcnt#**.

Returns:

the string "**#**" if **N** is not valid;

the name of model **N**, terminated by the character "**#**".

The model names are escaped as explained at page 76. Model names are case-sensitive and have a maximum length of 15 characters. Spaces at the end are ignored.

Available from version 2.13.3.

:modelld0NAME#

Loads the model with the given **NAME** from the alignment models saved in the mount. The model names are escaped as explained at page 76. Model names are case-sensitive and have a maximum length of 15 characters. Spaces at the end are ignored.

Returns:

the string "**1#**" if the model has been loaded correctly;

the string "**0#**" if there was an error.

Available from version 2.13.3.

:modelsv0NAME#

Saves in the mount the current model with the given **NAME**.

The model names are escaped as explained at page 76. Model names are case-sensitive and have a maximum length of 15 characters. Spaces at the end are ignored.

Returns:

the string "1#" if the model has been saved correctly;
the string "0#" if there was an error.
Available from version 2.13.3.

:modeldel0NAME#

Deletes the model with the given NAME from the alignment models saved in the mount.

The model names are escaped as explained at page 76. Model names are case-sensitive and have a maximum length of 15 characters. Spaces at the end are ignored.

Returns:

the string "1#" if the model has been deleted;

the string "0#" if there was an error.

Available from version 2.13.3.

Get Information

:GA# Get telescope altitude. Returns the current telescope altitude above the horizon formatted as follows:

Emulation and precision	Return value
Any emulation, low precision	sDD*MM# (degrees, arcminutes)
Any emulation, high precision	sDD*MM:SS# (degrees, arcminutes, arcseconds)
Any emulation, ultra precision	sDD:MM:SS.S# (degrees, arcminutes, arcseconds and tenths of arcsecond)

:Ga# Get current target altitude. Returns the target altitude above the horizon formatted as follows:

Emulation and precision	Return value
Any emulation, low precision	sDD*MM# (degrees, arcminutes)
Any emulation, high precision	sDD*MM:SS# (degrees, arcminutes, arcseconds)
Any emulation, ultra precision	sDD:MM:SS.S# (degrees, arcminutes, arcseconds and tenths of arcsecond)

If the target position has been set using the :Sr and :Sd commands, the return value is the altitude of the target computed from its equatorial coordinates at the time the :Ga command is received. Otherwise, it is the value set using the :Sa command. If neither an equatorial target position nor a target altitude has been set, the return value is undefined.

From version 2.12.14 onwards, when the mount is commanded to slew to a target from the keypad / virtual keypad interface, the target altitude is updated with the altitude of the target set by the keypad / virtual keypad interface.

:GC# Get current date. Returns the current date formatted as follows:

Emulation and precision	Return value
LX200 emulation, low and high precision	MM/DD/YY# (month, day, year)
Extended emulation, low and high precision	MM:DD:YY# (month, day, year) – note that the separator character is ':' instead of '/'.
Any emulation, ultra precision	YYYY-MM-DD# (year, month, day) – note that the separator character is '-' instead of '/'.

:GD# Get telescope declination. Returns the current telescope declination formatted as follows:

Emulation and precision	Return value
LX200 emulation, low and high precision	sDD*MM# (degrees, arcminutes)
Extended emulation, low and high precision	sDD*MM:SS# (degrees, arcminutes, arcseconds)
Any emulation, ultra precision	sDD:MM:SS.S# (degrees, arcminutes, arcseconds and tenths of arcsecond)

:Gd# Get current target declination. Returns the target declination formatted as follows:

Emulation and precision	Return value
LX200 emulation, low and high precision	sDD*MM# (degrees, arcminutes)
Extended emulation, low and high precision	sDD*MM:SS# (degrees, arcminutes, arcseconds)
Any emulation, ultra precision	sDD:MM:SS.S# (degrees, arcminutes, arcseconds and tenths of arcsecond)

If the target position has been set using the :Sa and :Sz commands, the return value is the declination of the target computed from its altazimuth coordinates at the time the :Gd command is received. Otherwise, it is the value set using the :Sd command. If neither an altazimuth target position nor a target declination has been set, the return value is undefined.

From version 2.12.14 onwards, when the mount is commanded to slew to a target from the keypad / virtual keypad interface, the target declination is updated with the declination of the target set by the keypad / virtual keypad interface.

:Gev# Gets the current site elevation.

Returns: sXXXX.X#

The current site elevation expressed in metres.

Available from version 2.9.9.

:GG# Get UTC offset time. Returns the number of hours to add to local time to convert it to UTC. The daylight savings setting in effect is factored into the returned value. The return value is formatted as follows:

Emulation and precision	Return value
LX200 emulation, low or high precision	sHH.H# (sign, hours and tenths of hours)
Extended emulation, low precision	sHH:MM.M# (sign, hours, minutes and tenths of minute)
Extended emulation, high precision	sHH:MM:SS.S# (sign, hours, minutes, seconds and tenths of second)
Any emulation, ultra precision	sHH:MM:SS.S# (sign, hours, minutes, seconds and tenths of second)

:Gg# Get current site longitude. **Note: East Longitudes are expressed as negative.** Returns the current site longitude formatted as follows:

Emulation and precision	Return value
Any emulation, low precision or LX200 emulation, high precision	sDDD*MM# (sign, degrees, arcminutes)
Extended emulation, high precision	sDDD*MM:SS# (sign, degrees, arcminutes, arcseconds)
Any emulation, ultra precision	sDDD:MM:SS.S# (sign, degrees, arcminutes, arcseconds, tenths of arcsecond)

:Gh# Get high limit. Returns the highest altitude above the horizon that the mount will be allowed to slew to without reporting an error message, formatted as follows:

Emulation and precision	Return value
Any emulation, low precision or high precision	sDD*# (sign, degrees)
Any emulation, ultra precision	sDD# (sign, degrees) (note the absence of '*')

:Ginfo# Get multiple information. Returns a string where multiple data are encoded, separated by commas ',', and terminated by '#'. Data are recognized by their position in the string. The data are the following:

Position	Datum
1	The telescope right ascension in hours and decimals (from 000.000000 to 23.999999), true equinox and equator of date of observation (i.e. Jnow).
2	The telescope declination in degrees and decimals (from -90.000000 to +90.000000), true equinox and equator of date of observation (i.e. Jnow).

- 3 A flag indicating the pointing state on which the telescope is currently positioned ("E" or "W"). This was formerly called "side of the pier". Please see the detailed explanation on page 72.
- 4 The telescope azimuth in degrees and decimals (from 000.00000 to 359.99999).
- 5 The telescope altitude in degrees and decimals (from -90.00000 to +90.00000).
- 6 The julian date (JJJJJJ.JJJJJJ), UTC, with leap second flag (see command :GJD2# for the description of this datum).
- 7 A number encoding the status of the mount as in the :Gstat command.
- 8 A number returning the slew status (0 if :D# would return no slew, 1 otherwise).

The string is terminated by '#'. Other parameters may be added in future at the end of the string: do not assume that the number of parameters will stay the same. Modified in version 2.16.12 with an added decimal digit for right ascension, declination, azimuth and altitude.

Available from version 2.14.9 (previous versions may have this command but it was experimental and possibly with a different format).

:GINQ# Get the type of connection.

Returns:

- 0# if the connection is over a serial RS-232 port
- 1# if the connection is over a GPS or GPS/RS-232 port
- 2# if the connection is over a cabled LAN port
- 3# if the connection is over a wireless LAN

Available from version 2.10.

:GIP# Get the IP address of the mount.

Returns: nnn.nnn.nnn.nnn,mmm.mmm.mmm.mmm,ggg.ggg.ggg.ggg,c#

A string containing the IP address (nnn.nnn.nnn.nnn), the subnet mask (mmm.mmm.mmm.mmm), the gateway (ggg.ggg.ggg.ggg) and a character (c) that is set to "D" if the address is obtained from a DHCP server, or "M" if the address is configured manually.

:GIPW# Get the wireless IP address of the mount.

Returns: nnn.nnn.nnn.nnn,mmm.mmm.mmm.mmm,ggg.ggg.ggg.ggg,c#

A string containing the IP address (nnn.nnn.nnn.nnn), the subnet mask (mmm.mmm.mmm.mmm), the gateway (ggg.ggg.ggg.ggg) and a character (c) that is set to "D" if the address is obtained from a DHCP server, or "M" if the address is configured manually.

Available from version 2.9.8.

:GMAC# Returns the MAC address of the Ethernet interface.

Returns: a string

"NN:NN:NN:NN:NN:NN#" the MAC address (or the string "#" if no Ethernet interface is available).

Available from version 2.14.11.

:GMACW#

Returns the MAC address of the Wireless interface.

Returns: a string

"NN:NN:NN:NN:NN:NN#" the MAC address (or the string "#" if no wireless interface is available).

Available from version 2.14.11.

:GJD# Get the current Julian Date.

Returns: JJJJJJ.JJJJ#

The current Julian Date for the mount.

Note

The Julian Date is computed from the UTC time. During leap seconds, the value of the Julian Date should be considered invalid.

:GJD1# Get the current Julian Date with extended precision.

Returns: JJJJJJ.JJJJJJ#

The current Julian Date for the mount in extended precision (8 decimal places).

Available from version 2.10.

Note

The Julian Date is computed from the UTC time. During leap seconds, the value of the Julian Date should be considered invalid.

:GJD2# Get the current Julian Date with extended precision and leap second flag.

Returns: JJJJJJ.JJJJJJ# or JJJJJJ.JJJJJJL#

The current Julian Date for the mount in extended precision (8 decimal places), with an optional "L" appended at the end to signal that we are in a leap second.

Available from version 2.13.2.

Note

The Julian Date is computed from the UTC time. During leap seconds, the value of the Julian Date continues to increase, with the "L" flag set. So, for example, you will have, around the time of the leap second of 2015 June 30:

Date	Time (UTC)	Result of :GJD2#
2015-06-30	23:59:59.0	2457204.49998843#
2015-06-30	23:59:59.5	2457204.49999421#
2015-06-30	23:59:60.0	2457204.50000000L#
2015-06-30	23:59:60.5	2457204.50000579L#
2015-07-01	00:00:00.0	2457204.50000000#
2015-07-01	00:00:00.5	2457204.50000579#

:GL# Get local time. Returns the local time in 24-hour format, formatted as follows:

Emulation and precision	Return value
LX200 emulation, low or high precision	HH:MM:SS# (hours, minutes, seconds)
Extended emulation, low precision	HH:MM.T# (hours, minutes and tenths of minutes)
Extended emulation, high precision	HH:MM:SS.S# (hours, minutes, seconds)
Any emulation, ultra precision	HH:MM:SS.SS# (hours, minutes, seconds, hundredths of second)

:GLDT# Get local date and time. Returns the local date and time formatted as follows:
 <date>,<time>#
 where <date> is formatted as in the :GC# command, and <time> is formatted as in the :GL# command.
 Available from version 2.12.26.

:GUDT# Get UTC date and time. Returns the UTC date and time formatted as follows:
 <date>,<time>#
 where <date> is formatted as in the :GC# command, and <time> is formatted as in the :GL# command.
 Available from version 2.12.26.

:GDUT# Get the current UT1 – UTC difference in seconds and decimals.
 Returns:
 XXX.XX# the UT1 – UTC difference in seconds and decimals (with sign).
 Available from version 2.13.1.
 Note: this was incorrectly specified as UTC – UT1 in past versions of this document.

:GDUTV# Gets the current status of the ΔT flag and the expiration date for the ΔT – UTC data.
 Returns a string formatted as follows: "F,XXXX-XX-XX#", where:
 XXXX-XX-XX the expiration date of the ΔT – UTC data (UTC);
 F a flag which is "V" if data are valid, "E" if expired.
 Available from version 2.15.

:GDGPS# Get the current GPS – UTC difference in seconds.
 Returns:
 XX# the current GPS – UTC difference in seconds.
 Available from version 2.13.1.

:GULEAP#
 Gets the date of the next leap second that will be accounted for.
 Returns:
 XXXX-XX-XX# the date (UTC) of the next leap second, if available;
 E# if no leap second is due according to the data loaded in the mount.
 Available from version 2.13.1.

:GMF# Get meridian side behaviour.

Returns: n#

A number with the following interpretation:

- 1 both sides of the meridian allowed;
- 2 only objects west of the meridian allowed (plus slew flip tolerance) – all slews are done so that after the slew the ":pS#" command would return "East#";
- 3 only objects east of the meridian allowed (plus slew flip tolerance) – all slews are done so that after the slew the ":pS#" command would return "West#".

Available from version 2.7.7.

:Go# Get lower limit. Returns the lowest altitude above the horizon that the mount will be allowed to slew to without reporting an error message, formatted as follows:

Emulation and precision	Return value
Any emulation, low precision or high precision	sDD*# (sign, degrees)
Any emulation, ultra precision	sDD# (sign, degrees) (note the absence of '*')

:GPA# Get parallactic angle with respect to actual mount orientation. This computation accounts for the actual alignment of the mount with the celestial pole, so it should be used to control an actual field derotator connected to the mount.

Returns: +AAA.AAAAA#

The parallactic angle with respect to actual mount orientation in degrees and decimals. This

Available from version 2.15.14.

:GPAS# Get parallactic speed with respect to actual mount orientation. This computation accounts for the actual alignment of the mount with the celestial pole, so it should be used to control an actual field derotator connected to the mount.

Returns: +AAA.AAAAA#

The parallactic speed (the rate of change of the parallactic angle) with respect to actual mount orientation in arcseconds per second.

Available from version 2.15.14.

:GPAZ# Get parallactic angle with respect to zenith.

Returns: +AAA.AAAAA#

The parallactic angle in degrees and decimals.

Available from version 2.15.19.

:GPASZ# Get parallactic speed with respect to zenith.

Returns: +AAA.AAAAA#

The parallactic speed (the rate of change of the parallactic angle) in arcseconds per second.

Available from version 2.15.19.

:Gpgc# Get guiding status.

Returns: n#

A number with the following interpretation:

- 0 the mount is not guiding;
- 1 the mount is guiding in right ascension / azimuth;
- 2 the mount is guiding in declination / altitude;
- 3 the mount is guiding in both axes.

Available from version 2.9.9.

Note: a bug prevents this command from working up to version 2.9.20. Use it only from version 2.9.21 upwards.

:GR# Get telescope right ascension. Returns the current telescope right ascension formatted as follows:

Emulation and precision	Return value
Any emulation, low precision	HH:MM.M# (hours, minutes and tenths of minutes)
LX200 emulation, high precision	HH:MM:SS# (hours, minutes, seconds)
Extended emulation, high precision	HH:MM:SS.S# (hours, minutes, seconds and tenths of seconds)
Any emulation, ultra precision	HH:MM:SS.SS# (hours, minutes, seconds and hundredths of seconds)

:Gr# Get current target RA. Returns the target right ascension formatted as follows:

Emulation and precision	Return value
Any emulation, low precision	HH:MM.M# (hours, minutes and tenths of minutes)
LX200 emulation, high precision	HH:MM:SS# (hours, minutes, seconds)
Extended emulation, high precision	HH:MM:SS.S# (hours, minutes, seconds and tenths of seconds)
Any emulation, ultra precision	HH:MM:SS.SS# (hours, minutes, seconds and hundredths of seconds)

If the target position has been set using the :Sa and :Sz commands, the return value is the right ascension of the target computed from its altazimuth coordinates at the time the :Gr command is received. Otherwise, it is the value set using the :Sr command. If neither an altazimuth target position nor a target right ascension has been set, the return value is undefined.

From version 2.12.14 onwards, when the mount is commanded to slew to a target from the keypad / virtual keypad interface, the target right ascension is updated with the right ascension of the target set by the keypad / virtual keypad interface.

:GRLYn# Get the status of relay n, where n is an ASCII digit (1...9) with the following interpretation:

- 1 User relay 1
- 2 User relay 2
- 3 User relay 3
- 4 User relay 4
- 5 User relay 5
- 6 User relay 6
- 7 Right Ascension/Azimuth motor heater
- 8 Declination/Altitude motor heater

Returns: 0 or 1 depending on the state of the relay, 1 means the relay is closed.
Available only for special purpose mounts with external relay control.

Note

The relays are in an undefined state during startup, then default at open (zero).

:GRPRS# Get the atmospheric pressure used in the refraction model. Note that this is the pressure at the location of the telescope, and not the pressure at sea level.

Returns: PPPP.P#
The required pressure in hPa.
Available from version 2.3.0.

:GRTMP# Get the temperature used in the refraction model

Returns: +TTT.T#
The required temperature in degrees Celsius (°C).
Available from version 2.3.0.

:GS# Get the sidereal time. Returns the local sidereal time formatted as follows:

Emulation and precision	Return value
Any emulation, low precision	HH:MM.M# (hours, minutes and tenths of minutes)
LX200 emulation, high precision	HH:MM:SS# (hours, minutes, seconds)
Extended emulation, high precision	HH:MM:SS.S# (hours, minutes, seconds and tenths of seconds)
Any emulation, ultra precision	HH:MM:SS.SS# (hours, minutes, seconds and hundredths of seconds)

Note

The result may be incorrect in firmware version <= 2.14.3.

:GREF# Gets the current status of the refraction correction.

Returns:
0 Refraction correction inactive
1 Refraction correction active

Available from version 2.10 (previous versions have refraction always active).

:GSC# Gets the current status of the speed correction flag

Returns:

- 0 Speed correction inactive
- 1 Speed correction active

Note

When the speed correction is active, the speed of any movement in the RA/azimuth axis is multiplied by $\cos(\text{dec})^{-1}$ or $\cos(\text{altitude})^{-1}$. In this way the angular speed on the sky is always constant. This is useful when autoguiding, since the relation between the duration of the correction pulses and the offsets in the focal plane of the telescope becomes independent from the declination (or altitude) of the mount.

:GSTAT# (deprecated, use :Gstat# instead) Gets the status of the mount.

Returns: a number depending on the current status of the mount. See :Gstat# for the list of codes.

Note: this command is deprecated because it has no '#' terminator.

:Gstat# Gets the status of the mount.

Returns: a number depending on the current status of the mount, terminated by a '#', with the following meaning:

- 0# The mount is tracking.
- 1# The mount is stopped after the pressing of the STOP key, receiving the :STOP# command or completing an homing sequence. Note that up to firmware version 2.12.21 included, the status will be returned as 1# just after the STOP key is pressed or the :STOP# command is received, even if the mount hasn't stopped yet. Since firmware version 2.12.22, this status will be returned only if the motors are actually stopped.
- 2# The mount is slewing to the park position.
- 3# The mount is unparking.
- 4# The mount is slewing to the home position.
- 5# The mount is parked.
- 6# The mount is slewing, or the mount is going to stop (but still moving) after the STOP key has been pressed or the :STOP# command has been received.
- 7# Tracking is off and the mount is not moving.
- 8# The motors are inhibited because of low temperature (only for special-purpose mounts with temperature sensors) , or the motors are subject to pre-heating before startup.
- 9# Tracking is on but the mount is outside tracking limits.
- 10# The mount is following a precalculated satellite trajectory.
- 11# The mount needs an user intervention to authorize movement, due to a suspected inconsistency in data (see :USEROK# command); if this occurs, DO NOT assume anything about the correctness of the mount position or alignment data.

98# Unknown status.
99# Error.
Available from version 2.8.8.

Note

The slew settle time set by the :Sstm command does not affect the status returned by this command.

:Gstm# Returns the slew settle time. After a slew has been completed, the :D# and :GDW# commands will return a slewing status for the time duration set in this command.

Note: the :Gstat# command is not affected by this setting.

Returns: **NNNNN.NNN#**

The mount settle time in seconds.

Available from version 2.9.14.

:GDstm# Returns the dome settle time. After a slew has been completed, the :GDW# and :GDw# commands will return a slewing status for the time duration set in this command.

Returns: **NNNNN.NNN#**

The dome settle time in seconds.

Available from version 2.9.14.

:Glmt# Returns the meridian limit for tracking in degrees.

Returns: **NN#**

The meridian limit for tracking in degrees.

Available from version 2.11.

:Glms# Returns the meridian limit for slews in degrees.

Returns: **NN#**

The meridian limit for slews in degrees.

Available from version 2.11.

:Gmte# Returns the estimated time to tracking end due to horizon / flip limits reached.

Returns: **NNNN#**

The estimated time to tracking end in minutes of time.

Available from version 2.11.

:Guaf# Returns the unattended flip setting.

Returns:

0 disabled

1 enabled

Available from version 2.11.

Note: unattended flip didn't work properly in firmware versions up to 2.13.8 included.

:GT# Get tracking rate.

Returns: TT.T#

This value is computed in order to emulate the corresponding LX200 command. This corresponds to the equivalent frequency expressed in hertz assuming a synchronous motor design where a 60.0 Hz motor clock would produce 1 revolution of the mount in 24 hours. So, in order to obtain the tracking rate in arcseconds per second of time, this value should be divided by four.

:Gt# Get current site latitude. Returns the latitude of the current site formatted as follows:

Emulation and precision	Return value
Any emulation, low precision	sDD*MM# (sign, degrees, minutes)
LX200 emulation, high precision	sDD*MM# (sign, degrees, minutes)
Extended emulation, high precision	sDD*MM:SS# (sign, degrees, arcminutes, arcseconds)
Any emulation, ultra precision	sDD:MM:SS.S# (sign, degrees, arcminutes, arcseconds, tenths of arcsecond)

Positive implies north latitude.

:GTMPn# Get the temperature of element n, where n is an ASCII digit (1...9) with the following interpretation:

- 1 Right Ascension/Azimuth motor driver
- 2 Declination/Altitude motor driver
- 7 Right Ascension/Azimuth motor
- 8 Declination/Altitude motor
- 9 Electronics box temperature sensor
- 11 Keypad (v2) display sensor
- 12 Keypad (v2) PCB sensor
- 13 Keypad (v2) controller sensor

Returns: +TTT.T#

The required temperature in degrees Celsius (°C).

If the required temperature cannot be read, the string "Unavailable#" is returned.

If the electronics box temperature sensor is available, its readout can be used to monitor the electronics temperature to avoid overheating. The electronics box temperature should never go above +65°C. If the temperature goes above +65°C, immediate action should be taken in order to preserve the electronics – either cooling it or shutting it off. Higher temperatures would damage the electronics and the power supplies will automatically shut down at +70°C.

7, 8, 9 are available only for special-purpose mounts with temperature sensors.

11, 12, 13 are available only if a physical keypad version 2 is connected to the mount.

Available from version 2.3.0.

:GTMLT#

Gets the status of the detection of low temperature conditions where the maximum slewing performance of the mount can be limited.

Returns:

- 0 no low temperature condition has been detected

1 a low temperature condition has been detected, the mount performance can be limited.
Available from version 2.14.8.

:GTMPOHn#

Gets the temperature overheat threshold for the motors, where n is an ASCII digit (1...9) with the following interpretation:

- 7 Right Ascension/Azimuth motor
- 8 Declination/Altitude motor

Returns: a string containing the overheat temperature threshold T_H in degrees Celsius ($^{\circ}\text{C}$). When the temperature of the motor is above T_H , the motion is stopped and the heaters are turn off. The string is formatted as follows:

sTTT.T#

If the n parameter is invalid, the string "1" is returned.

Available only for special-purpose mounts with temperature sensors.

Available from version 2.7.8.

:GTMPThn#

Gets the temperature thresholds for the motors, where n is an ASCII digit (1...9) with the following interpretation:

- 7 Right Ascension/Azimuth motor
- 8 Declination/Altitude motor

Returns: a string containing the three temperature thresholds T_0 , T_1 and T_2 , in this order and in degrees Celsius ($^{\circ}\text{C}$). When the temperature of the motor is below T_0 , the motion is stopped and the heaters are powered. When the temperature drops below T_1 , the heaters are turned on, and when the temperature rises above T_2 , the heaters are turned off. The string is formatted as follows:

sTTT.T,sTTT.T,sTTT.T#

If the n parameter is invalid, the string "1" is returned.

Available only for special-purpose mounts with temperature sensors.

Available from version 2.3.0.

:GTRK# Get the current tracking status of the mount

Returns:

- 0 the mount is not tracking
- 1 the mount is tracking

Available from version 2.3.0.

:GTTRK# Get the tracking status of the target object

Returns:

- 0 the target object is located in a position where tracking is not allowed (i.e. below the horizon, or above $+89^{\circ}$ if using an altazimuth mount)
- 1 the target object is located in a position where tracking is allowed

Available from version 2.3.0.

:GTsid# Get the destination side of the target object

Returns:

- 0 no target defined, or the target object is located in a position where it is not possible to go
 - 2 the target is located in a position where the mount would slew the telescope to the west side so that after the slew the ":pS#" command would return "West#".
 - 3 the target is located in a position where the mount would slew the telescope to the east side so that after the slew the ":pS#" command would return "East#".
- Available from version 2.9.9.

Notes

See the :MSfs command.

Up to version 2.15.31 included, return values 2 and 3 were reversed in the southern hemisphere.

:GVD# Get firmware date.

Returns: mmm dd yyyy# (month as a three-letter string among "Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"; day; year)

:GVN# Get firmware number.

Returns: <string>#

A string containing the firmware revision.

The string is in the format "**maj**.**min**[.**fix**]#", where **maj** is the major version number (base 10), **min** is the minor version number (base 10), **fix** is the bugfix release number (base 10). The **.fix** part is optional; if omitted, you can assume that the bugfix release number is zero. For example the following strings are valid firmware version numbers.

"1.6.4#"

"1.7#"

"2.9#"

"2.15.35#"

"2.16#"

:GVP# Get product name.

Returns: <string>#

A string containing the product name:

"10micron GM1000HPS#" for a GM1000HPS mount

"10micron GM2000QCI#" for a GM2000QCI mount

"10micron GM2000HPS#" for a GM2000HPS mount

"10micron GM3000HPS#" for a GM3000HPS mount

"10micron GM4000QCI#" for a GM4000QCI mount

"10micron GM4000QCI 48V#" for a special GM4000QCI mount with 48V supply

"10micron GM4000HPS#" for a GM4000HPS mount

"10micron AZ2000#" for an altazimuth AZ2000 mount

"10micron AZ2000HPS#" for an altazimuth AZ2000HPS mount

"10micron AZ4000HPS#" for an altazimuth AZ4000HPS mount

:GVT# Get firmware time.

returns: HH:MM:SS# (hours, minutes, seconds)

:GVZ# Get control box hardware version.

Returns: <string#>

A string containing the control box version. Currently the string can be one of the following:

"Q-TYPE2016#"	identifies a Q-TYPE 2016 control box
"Q-TYPE2012#"	identifies a Q-TYPE 2012 control box
"PRE2012#"	identifies a pre-2012 control box
"UNKNOWN#"	identifies an unknown control box version

Available from version 2.9.9.

:GWOL# Get the wake-on-LAN configuration. Wake-on-LAN is available only on Q-TYPE2012 or Q-TYPE2016 control boxes.

Returns:

N# wake-on-LAN not available

0# wake-on-LAN not active

1# wake-on-LAN active

Available from version 2.15.7.

:GWAV# Query if wireless is available.

Returns:

the string "0#" if no wireless is available;

the string "1#" if a wireless adapter is available.

Available from version 2.10.

Note

Use this command to query for the existence of a wireless connection possibility. This doesn't mean that the wireless connection is active.

:GWRSC# Start scanning for wireless access points.

Returns:

the string "0#" if no wireless is available;

the string "1#" if a wireless adapter is available.

Available from version 2.10.

Note

This command will interrupt the wireless connection when the mount is configured as an hotspot.

:GWRAP# Get the wireless access points available.

Returns:

the string "0#" if no wireless is available;

a string composed by the character "1" followed by a comma-separated list of access point names, terminated by the character "#".

The access points names are escaped as explained at page 76.

Available from version 2.9.8.

:GWID# Get the ESSID of the wireless network if available

Returns:

the string "#" if the wireless network is not connected

the string "<ESSID>#" where <ESSID> is the current access point name (escaped as explained at page 76).

Available from version 2.10.

:GWUP# Get the wireless network status.

Returns:

the string "E#" if the wireless network has not been configured;

the string "0#" if the wireless network has been configured correctly and is setup as a client;

the string "1#" if the wireless network has been configured correctly and is setup as an hotspot;

the string "2#" if the wireless network configuration is underway.

Available from version 2.12.25.

:GWRAP2#

Get the wireless access points available with encryption information.

Returns:

the string "0#" if no wireless is available;

a string, whose first character is:

"1" – the last wireless access point scan (started by :GWRSC#) is underway

"2" – the last wireless access point scan is ended

and then is followed by a comma-separated list of access point names, terminated by the character "#". Each access point name is preceded by a single letter with the following meaning:

'o' – Open access point (no security)

'w' – WEP security

'1' – WPA PSK security

'2' – WPA2 PSK security

'x' – unsupported security

So if the last access point scan is finished, there are two access points, one with ESSID "Office" and WEP security, and another with ESSID "Home" and WPA-PSK security, the return string will be:

"2wOffice,1Home#"

The access points names are escaped as explained at page 76.

Available from version 2.10.

:GZ# Get the telescope azimuth. Returns the current azimuth of the telescope formatted as follows:

Emulation and precision	Return value
Any emulation, low precision	DDD*MM# (degrees, arcminutes)
Any emulation, high precision	DDD*MM:SS# (degrees, arcminutes, arcseconds)
Any emulation, ultra precision	DDD:MM:SS.S# (degrees, arcminutes, arcseconds and tenths of arcsecond)

:Gz# Get the target azimuth. Returns the azimuth of the current target formatted as follows:

Emulation and precision	Return value
Any emulation, low precision	DDD*MM# (degrees, arcminutes)
Any emulation, high precision	DDD*MM:SS# (degrees, arcminutes, arcseconds)
Any emulation, ultra precision	DDD:MM:SS.S# (degrees, arcminutes, arcseconds and tenths of arcsecond)

If the target position has been set using the :Sr and :Sd commands, the return value is the azimuth of the target computed from its equatorial coordinates at the time the :Gz command is received. Otherwise, it is the value set using the :Sz command. If neither an equatorial target position nor a target azimuth has been set, the return value is undefined.

From version 2.12.14 onwards, when the mount is commanded to slew to a target from the keypad / virtual keypad interface, the target azimuth is updated with the azimuth of the target set by the keypad / virtual keypad interface.

:pS# Get the pointing state on which the telescope is currently positioned.
Returns: the string "East#" or the string "West#".
This was formerly called "side of the pier". Please see the detailed explanation on page 72.

Note

Up to version 2.15.31 included, the East and West return values were reversed in the southern hemisphere.

:V# Get emulated firmware revision. This is implemented as some software needs it in order to function properly.
Returns: the string "G#".

:Gge# Get the guider port enable status.
Returns:
0 The guider port is disabled. Autoguide commands through the guider port are ignored.
1 The guider port is enabled. Autoguide commands through the guider port are executed.
Available from version 3.1.

Home & Park Commands

:hS#	Seeks the home position and stores the alignment information and the encoder values from the aligned mount at the home position in the nonvolatile memory of the mount. This command has effect only if the mount has homing sensors (GM4000QCI, AZ2000QCI). Returns: nothing
:hF#	Seeks the home position of the mount and sets/aligns the scope based on the encoder values and the alignment information stored in non-volatile memory. This command has effect only if the mount has homing sensors (GM4000QCI, AZ2000QCI). Returns: nothing
:hP#	Slew to park position Returns: nothing
:KA#	Slew to park position Returns: nothing
:PO#	Unpark Returns: nothing
:h?#	Query home status Returns: 0 home search failed 1 home search found 2 home search in progress

Notes

To detect the end of the park movement, the :D# command (see the next section) can be used.

The park position should be set with the hand controller or the virtual keypad software.

The alignment information and encoder values are saved when the mount is powered off even if it is not parked nor at the home position; as long as the mount is not moved manually, the position will be recovered at the next startup. The park command should be used in order to put the mount in a “safe” position, while the home commands are used when performing the initial alignment (in order to put the mount in a known position) and to recover the position information when the mount has been moved while unpowered. HPS and DDS mounts have absolute encoders, so the firmware always knows their position, even if the mount is moved manually when unpowered.

Movement Commands

:MA#	Slew to target altitude and azimuth. Returns:
------	--

Returns:

0 no error

if the target is below the lower limit: the string

"1Object Below Horizon #"

if the target is above the high limit: the string

"2Object Below Higher #"

if the slew cannot be performed due to another cause: the string

"3Cannot Perform Slew #"

if the mount is parked: the string

"4Mount Parked #"

if the mount is restricted to one side of the meridian and the object is on the other side:
the string

"5Object on the other side #"

Notes

The :MA# command will slew to the altazimuth coordinates defined by the commands :Sa (Set target altitude) and :Sz (Set target azimuth). After slewing to the target position, the mount will not track the object.

:Me# Move east (for equatorial mounts) or left (for altazimuth mounts) at current rate.

Returns: nothing

If this command is given while moving at the guide rate, and the ultra precision mode is not active, turning off the mount while this command is in effect will make the mount to stay parked when turned on again.

:Mn# Move north (for equatorial mounts) or up (for altazimuth mounts) at current rate.

Returns: nothing

:Ms# Move south (for equatorial mounts) or down (for altazimuth mounts) at current rate.

Returns: nothing

:Mw# Move west (for equatorial mounts) or right (for altazimuth mounts) at current rate.

Returns: nothing

:MgeXXXX#

:MeXXX# Corrects the position of the mount to the east (for equatorial mounts) or left (for altazimuth mounts) by an amount equivalent to a motion of XXX milliseconds at the current autoguide speed. The maximum length of the correction is 1000 milliseconds up to firmware revision 2.9.20. From firmware revision 2.10, the maximum length of the correction is 9999 milliseconds.

:MgnXXXX#

Notes

The :MS# command will slew to the equatorial coordinates defined by the commands :Sr (Set target right ascension) and :Sd (Set target declination). It is assumed that the coordinates are apparent, topocentric and NOT corrected for refraction. After slewing to the target position, the mount will track the object.

:MSfn# Slew to target object on the specified side, where **n** is 2 for west and 3 for east as defined in the :GMF and :SMF commands.

Returns:

0 no error

if the target is below the lower limit: the string

"1Object Below Horizon #"

if the target is above the high limit: the string

"2Object Below Higher #"

if the slew cannot be performed due to another cause: the string

"3Cannot Perform Slew #"

if the mount is parked: the string

"4Mount Parked #"

if the mount is restricted to one side of the meridian and the object is on the other side:
the string

"5Object on the other side #"

Available from version 2.9.9.

Note

Use the :MSfn command in conjunction with the :GTsid command in order to check the destination side before slewing to an object. Using :GTsid followed by :MS would permit getting a side and then slewing successfully to the other side if between the two commands the position of the object in the sky changes too much. Note that the behaviour obtained by using :GTsid for DestinationSideOfPier and :MS for slewing to the target (i.e. the possibility that the slew goes to the other side) is currently expected to happen within the ASCOM specification.

:MSnf# Slew to target object, disregarding the fine movement limit for slewing to the same meridian side.

Returns:

0 no error

if the target is below the lower limit: the string

"1Object Below Horizon #"

if the target is above the high limit: the string

"2Object Below Higher #"

if the slew cannot be performed due to another cause: the string

"3Cannot Perform Slew #"

if the mount is parked: the string

"4Mount Parked #"

if the mount is restricted to one side of the meridian and the object is on the other side:
the string

"5Object on the other side #"

Available from version 2.11.

:MSap# Slew to target object, missing it by a computed amount so that when the target is centered using the polar axis manual movements, the polar axis becomes aligned with the celestial pole. The command is available only on equatorial mounts. The command requires that a valid model with at least two stars is active. After changing the position of the polar axis, the model will be invalid (i.e. not useful for further tracking/slewing) and should be cleared. This command will slew to any target which can be reached, but in order for the alignment to work, the position of the target should satisfy some requisites. See the paragraph *Assisted polar axis alignment and orthogonality correction* for details.

Returns:

0 no error

if the target is below the lower limit: the string

"1Object Below Horizon #"

if the target is above the high limit: the string

"2Object Below Higher #"

if the slew cannot be performed due to another cause: the string

"3Cannot Perform Slew #"

if the mount is parked: the string

"4Mount Parked #"

if the mount is restricted to one side of the meridian and the object is on the other side:
the string

"5Object on the other side #"

Available from version 2.15.16.

:MSao# Slew to target object, missing it by a computed amount so that when the target is centered by physically adjusting the orthogonality of the telescope with the declination/altitude axis, the optical axis becomes orthogonal to the declination/altitude axis. The command requires that a valid model with at least three stars is active. After changing the orthogonality between the telescope and the declination/altitude axis, the model will be invalid (i.e. not useful for further tracking/slewing) and should be cleared. See the paragraph *Assisted polar axis alignment and orthogonality correction* for details.

Returns:

0 no error

if the target is below the lower limit: the string

"1Object Below Horizon #"

if the target is above the high limit: the string

"2Object Below Higher #"

if the slew cannot be performed due to another cause: the string

"3Cannot Perform Slew #"

if the mount is parked: the string

"4Mount Parked #"

if the mount is restricted to one side of the meridian and the object is on the other side:
the string

"5Object on the other side #"

Available from version 2.15.16.

:EW#	Swaps east and west directions. Returns: nothing
:NS#	Swaps north and south directions. Returns: nothing
:Q#	Halt all current slewing. Returns: nothing
:Qe#	Halt eastward (for equatorial mounts) or leftward (for altazimuth mounts) movements. Returns: nothing
:Qn#	Halt northward (for equatorial mounts) or upward (for altazimuth mounts) movements. Returns: nothing
:Qs#	Halt southward (for equatorial mounts) or downward (for altazimuth mounts) movements. Returns: nothing
:Qw#	Halt westward (for equatorial mounts) or rightward (for altazimuth mounts) movements. Returns: nothing
:FLIP#	This command acts in different ways on the AZ2000 and german equatorial (GM1000 – GM4000) mounts. On an AZ2000 mount: When observing an object near the lowest culmination, requests to make a 360° turn of the azimuth axis and point the object again. On a german equatorial mount: When observing an object near the meridian, requests to make a 180° turn of the RA axis and move the declination axis in order to point the object with the telescope on the other side of the mount. Returns: 1 if successful 0 if the movement cannot be done
:D#	Requests a string indicating the progress of the current slew operation. Returns: the string “■#”, where the block character has ascii code 127 (0x7F), if a slew is in progress or a slew has ended from less than the settle time set in command :Sstm. the string “#” if a slew has been completed or no slew is underway.

Note

If a dome is connected, check the :GDW# command, since it may be more appropriate for you.

Any rotator connected to the mount which is slewing will also cause this command to return the string corresponding to the slew in progress state.

:P# This command does nothing. It is included for compatibility with the LX200 command protocol, where it enables the high-precision pointing.

:NUDGEsXXXX,sYYYY#

This command cause the mount to move to a point which coordinates are offset from the current coordinates by sXXXX arcseconds in the RA/azimuth axis and sYYYY arcseconds in the declination/altitude axis. This is the equivalent of computing new target coordinates and slewing to them. Do not use this command for autoguiding. The typical usage of this command is to center objects after a previous slew.

Returns:

0 no error

the string "1Object Below Horizon #" if the target is below the lower limit.

the string "2Object Below Higher #" if the target is above the high limit.

the string "3Cannot Perform Nudge #" if the nudge cannot be performed due to another cause.

Available from version 2.7.4.

Note

If the speed correction option is selected (with the hand controller or the virtual keypad software) the actual movement on the RA/azimuth axis is increased in order to achieve the same angular movement on the sky at the different angles of declination/altitude.

The nudge operation is equivalent to a slew; i.e. it is possible to check if the operation is completed using the :D# command. Trying to start a new nudge operation while the previous operation is incomplete (or if the telescope is still slewing to a target) will fail.

Rate Commands

:RC# Set slew rate to Centering rate (2nd slowest)

Returns: nothing

:RCn# Sets the centering rate according to the value of n:

0 16x (0.067°/s)

1 64x (0.27°/s)

2 600x (2.5°/s)

3 1200x (5°/s)

If the selected rate is greater than the current slew rate, the centering rate is made equal to the slew rate. For example, if the slew rate is set to 900x (3.75°/s), the command :RC3# will set the centering rate to 900x (3.75°/s).

Returns: nothing

:RG# Set slew rate to Guiding Rate (slowest)

Returns: nothing

:RGn# Sets the guiding rate according to the value of n:

0 0.25x (3.75"/s)

1 0.5x (7.5"/s)

2 1.0x (15"/s)

Returns: nothing

:RM# Set slew rate to Find Rate (2nd fastest)

Returns: nothing

:RS# Set slew rate to max (fastest)

Returns: nothing

:RSn# Sets the slew rate according to the value of n:

0 1200x (5°/s)

1 900x (3.75°/s)

2 600x (2.5°/s)

If the selected rate is greater than the maximum slew rate, the slew rate is made equal to the maximum slew rate. For example, if the maximum slew rate is set to 3°/s, both the commands :RS0# and :RS1# will set the slew rate to 720x (3°/s).

Returns: nothing

:RADD.D# Set RA/azimuth slew rate to DD.D degrees per second (up to seven decimal places allowed).

Returns: nothing

:REDD.D# Set declination/altitude slew rate to DD.D degrees per second (up to seven decimal places allowed).

Returns: nothing

:RgSS.S# Set guiding rate to +/- SS.S to arc seconds per second.

When the guiding rate is selected (using the :RG# command), this rate is added or subtracted from the current tracking rates when the :Me# – :Mw# – :Mn# – :Ms# commands are issued, when the direction buttons are pressed on the keypad or the virtual keypad and also when guiding using the autoguide port. Rate shall not exceed sidereal speed (approx 15.041"/sec).

Returns: nothing

Note: with firmware version 3.1, the autoguider port can be disabled by software.

:RcXXX# Set the centering rate to XXX times the sidereal speed, where XXX is comprised between 1 and 255.

Available from version 2.7.8.

Returns: nothing

:RsXXXX# Set the slew rate to XXXX times the sidereal speed, where XXXX is comprised between 1 and 1200.

Available from version 2.7.8.
Returns: nothing

:RMsXX# Set the slew rate (for automated moves) to XX degrees/s in the allowed range for the mount. Note: this does not select the slew rate for axis movements. In order to select the slew rate for axis movements, use the :RS command after this one.

Returns

0 valid
1 invalid

Available from version 2.9.9.

:GMs# Get the current slew rate in degrees/s.

Returns: XX#
the current slew rate in degrees/s.
Available from version 2.9.9.

:GMsa# Get the minimum slew rate that can be set in the mount in degrees/s.

Returns: XX#
the minimum slew rate in degrees/s.
Available from version 2.9.9.

:GMsb# Get the maximum slew rate that can be set in the mount in degrees/s.

Returns: XX#
the maximum slew rate in degrees/s.
Available from version 2.9.9.

:Ggui# Get current guide rate.

Returns: S.SS# (arcseconds/s)
The current guide rate.
Available from version 2.9.11.

Set Commands

:BdDD*MM:SS# or :BdHH:MM:SS# or :BdHH:MM:SS.S#

Sets declination/altitude backlash to DD*MM:SS (degrees, arcminutes, arcseconds) or HH:MM:SS(.S) (hours, minutes, seconds and optionally tenths of seconds).
Returns:
1 valid

:BrDD*MM:SS# or :BrHH:MM:SS# or :BrHH:MM:SS.S#

Sets RA/azimuth backlash to DD*MM:SS (degrees, arcminutes, arcseconds) or HH:MM:SS(.S) (hours, minutes, seconds and optionally tenths of seconds).
Returns:
1 valid

:SasDD*MM# or :SasDD*MM:SS# or :SasDD*MM:SS.S#

Set target object altitude to sDD*MM (sign, degrees, arcminutes), sDD*MM:SS (sign, degrees, arcminutes, arcseconds) or sDD*MM:SS.S (sign, degrees, arcminutes, arcseconds, tenths of arcsecond)

Returns:

- 0 object out of slew range
- 1 object within slew range

:SBn# Set baud rate n, where n is an ASCII digit (0...9) with the following interpretation:

- 0 115.2K
- 1 57.6K
- 2 38.4K
- 4 19.2K
- 6 9600
- 7 4800
- 8 2400
- 9 1200

Returns:

- 0 invalid rate
- 1 new baud rate accepted; this character is returned at the current baud rate and then changes to the new rate for further communication

Notes

This command has effect only on the RS-232 communication link.

:SCMM/DD/YY# or :SCMM/DD/YYYY# or :SCYYYY-MM-DD#

Set date to MM/DD/YY (month, day, year), MM/DD/YYYY (month, day, year) or YYYY-MM-DD (year, month, day). The date is expressed in local time.

Returns:

- 0 if the date is invalid

The string "1Updating Planetary Data. #" if the date is valid.

The string "1<32 spaces>#<32 spaces>#" in extended LX200 emulation mode.

The character "1" without additional strings in ultra-precision mode (regardless of emulation).

:SdsDD*MM# or :SdsDD*MM:SS# or :Sd sDD*MM:SS.S#

Set target object declination to sDD*MM (sign, degrees, arcminutes), sDD*MM:SS (sign, degrees, arcminutes, arcseconds) or sDD*MM:SS.S (sign, degrees, arcminutes, arcseconds and tenths of arcsecond)

Returns:

- 0 invalid
- 1 valid

:SevsXXXX.X#

Set current site's elevation to sXXXX.X (sign, metres) in the range -1000.0 to 9999.9.

Returns:

0 invalid
1 valid

Available from version 2.9.9.

:SgsDDD*MM# or :SgsDDD*MM:SS# or :SgsDDD*MM:SS.S#

Set current site's longitude to sDDD*MM (sign, degrees, arcminutes), sDDD*MM:SS (sign, degrees, arcminutes, arcseconds) or sDDD*MM:SS.S (sign, degrees, arcminutes, arcseconds and tenths of arcsecond). **Note: East Longitudes are expressed as negative.**

Returns:

0 invalid
1 valid

:SGsHH.H# or :SGsHH:MM.M# or :SGsHH:MM:SS#

Set the number of hours added to local time to yield UTC (sign, hours and tenths of hour; sign, hours, minutes and tenths of minute; sign, hours, minutes and seconds)

Returns:

0 invalid
1 valid

Notes

The mount internally keeps UTC time. The :SG command operates only on the conversion between UTC time and the local time. In order to work with UTC, use :SG+00.0# before all time get/set commands. The UTC offset set by this command (and got by the :GG command) accounts for the DST set in the keypad: if the DST changes from off to on on the keypad, the UTC offset will decrease by an hour, and the time got from the :GL command will increase by an hour. If you are not using the keypad, you can ignore the DST setting.

:ShsDD# Set the highest altitude to which the telescope will slew to sDD degrees.

Returns:

0 invalid
1 valid

:SJDJJJJJJ.JJJJ#

Set the Julian Date to the given value (up to eight decimal places).

Returns:

0 invalid
1 valid

Note

The Julian Date is computed from the UTC time. During leap seconds, there is no valid value for the Julian Date, so you cannot use this command to set time during leap seconds.

:SLHH:MM:SS#, :SLHH:MM:SS.S# or :SLHH:MM:SS.SS#

Set the local Time to HH:MM:SS (hours, minutes, seconds), HH:MM:SS.S (hours, minutes, seconds and tenths of second) or HH:MM:SS.SS (hours, minutes, seconds and hundredths of second).

Returns:

0 invalid
1 valid

Note

The UTC time (and the local time which derives from it) may contain leap seconds. This means that sometimes the seconds field may contain values from 60.00 to 60.99.

:SLDTYYYY-MM-DD,HH:MM:SS#

Set together the local date and time to **HH:MM:SS** (hours, minutes, seconds) and the local date to **YYYY-MM-DD** (year, month, day of month).

You may specify the date also with **MM/DD/YY** (month, day, year from 2000) or **MM/DD/YYYY** (month, day, year).

You may specify the time also with **HH:MM:SS.S** or **HH:MM:SS.SS** (adding decimal digits to the seconds).

Returns:

0 invalid
1 valid

Available from version 2.12.26.

Note

The UTC time (and the local time which derives from it) may contain leap seconds. This means that sometimes the seconds field may contain values from 60.00 to 60.99.

:SUDTYYYY-MM-DD,HH:MM:SS#

Set together the UTC date and time to **HH:MM:SS** (hours, minutes, seconds) and the local date to **YYYY-MM-DD** (year, month, day of month).

You may specify the date also with **MM/DD/YY** (month, day, year from 2000) or **MM/DD/YYYY** (month, day, year).

You may specify the time also with **HH:MM:SS.S** or **HH:MM:SS.SS** (adding decimal digits to the seconds).

Returns:

0 invalid
1 valid

Available from version 2.12.26.

Note

The UTC time (and the local time which derives from it) may contain leap seconds. This means that sometimes the seconds field may contain values from 60.00 to 60.99.

:SMF_n# Set meridian side behaviour, where *n* is an ASCII digit (1..3) with the following interpretation:

- 1 both sides of the meridian allowed;
- 2 only objects west of the meridian allowed (plus slew flip tolerance) – all slews are done so that after the slew the ":pS#" command would return "East#";
- 3 only objects east of the meridian allowed (plus slew flip tolerance) – all slews are done so that after the slew the ":pS#" command would return "West#".

Returns:

0 invalid

1 valid
Available from version 2.7.7.

Note

The meridian side behaviour cannot be changed with altazimuth mounts.

:SosDD# Set the minimum altitude above the horizon to which the telescope will slew to sDD degrees. Valid values are between -5 and +45 degrees.

Returns:

0 invalid
1 valid

:SrHH:MM:T# or :SrHH:MM:SS# or :SrHH:MM:SS.S# or :SrHH:MM:SS.SS#

Set target object RA to HH:MM:T (hours, minutes and tenths of minutes), HH:MM:SS (hours, minutes, seconds), HH:MM:SS.S (hours, minutes, seconds and tenths of second) or HH:MM:SS.SS (hours, minutes, seconds and hundredths of second).

Returns:

0 invalid
1 valid

:SRLYn,m#

Set the status of relay n, where n is an ASCII digit (1...9) with the following interpretation:

1 User relay 1
2 User relay 2
3 User relay 3
4 User relay 4
5 User relay 5
6 User relay 6

m is the new status of the relay, with 0 = open and 1 = closed.

Returns:

0 invalid
1 valid

Available only for special purpose mounts with external relay control.

Available from version 2.3.

:SREFn# Sets the current status of the refraction correction, where n has the following interpretation:

0 deactivate refraction correction
1 activate refraction correction

Returns:

0 invalid
1 valid

Available from version 2.10 (previous versions have refraction always active).

:SRPRSPPPP.P# Sets the atmospheric pressure used in the refraction model to PPPP.P hPa. Note that this is the pressure at the location of the telescope, and not the pressure at sea level.

Returns:

0 invalid
1 valid

Available from version 2.3.0.

:SRTMPsTTT.T# Sets the temperature used in the refraction model to sTTT.T degrees Celsius (°C).

Returns:

0 invalid
1 valid

Available from version 2.3.0.

:SSCn# Sets the current status of the speed correction flag (see the :GSC# command), where n has the following interpretation:

0 deactivate speed correction
1 activate speed correction

Returns:

0 invalid
1 valid

:SstmNNNNN.NNN #

Sets the slew settle time to NNNNN.NNN seconds (from 0 to 99999 s). After a slew of the mount has been completed, the :D# and :GDW# commands will return a slewing status for the time duration set in this command.

Note: the :Gstat# command is not affected by this setting.

Returns:

0 invalid
1 valid

Available from version 2.9.14.

:SDstmNNNNN.NNN #

Sets the dome settle time to NNNNN.NNN seconds (from 0 to 99999 s). After a dome slew has been completed, the :GDW# and :GDw# commands will return a slewing status for the time duration set in this command.

Returns:

0 invalid
1 valid

Available from version 2.9.14.

:SlmtNN# Sets the meridian limit for tracking in degrees. The minimum value is the same as the meridian limit for slews.

Returns:

0 invalid

1 valid

Available from version 2.11.

Note: on AZ2000HPS and AZ4000HPS mounts the meridian limit cannot be set. This command returns always 0.

:SlmsNN# Sets the meridian limit for slews in degrees. Setting a meridian limit for slews greater than the meridian limit for tracking will increase the meridian limit for tracking to the same value.

Returns:

0 invalid

1 valid

Available from version 2.11.

Note: on AZ2000HPS and AZ4000HPS mounts the meridian limit cannot be set. This command returns always 0.

:SuafN# Enables or disables the unattended flip. Use N=1 to enable, N=0 to disable. This is set always to 0 after power up.

Returns: nothing

Available from version 2.11.

Note

unattended flip didn't work properly in firmware versions up to 2.13.8 included.

:StsDD*MM# or :StsDD*MM:SS# or :StsDD*MM:SS.S#

Sets the current site latitude to sDD*MM (sign, degrees, arcminutes), sDD*MM:SS (sign, degrees, arcminutes, arcseconds), or sDD*MM:SS.S (sign, degrees, arcminutes, arcseconds and tenths of arcsecond)

Returns:

0 invalid

1 valid

:STMPThn,sTTT.T,sTTT.T,sTTT.T#

Sets the temperature thresholds for the motors, where n is an ASCII digit (1...9) with the following interpretation:

7 Right Ascension/Azimuth motor

8 Declination/Altitude motor

sTTT.T are the three temperature thresholds T_0 , T_1 and T_2 , in this order and in degrees Celsius (°C). They must obey the relation $T_0 < T_1 < T_2$ and must be in the range -100 / +40 °C. When the temperature of the motor is below T_0 , the motion is stopped and the heaters are powered. When the temperature drops below T_1 , the heaters are turned on, and when the temperature rises above T_2 , the heaters are turned off.

Returns:

0 invalid

1 valid

Available only for special-purpose mounts with temperature sensors.

Available from version 2.3.

:STMPOH_nsTTT.T#

Sets the temperature overheat threshold for the motors, where n is an ASCII digit (1...9) with the following interpretation:

7 Right Ascension/Azimuth motor

8 Declination/Altitude motor

sTTT.T is the temperature threshold T_H in degrees Celsius (°C) . It be in the range 0 / +80 °C. When the temperature of the motor is above T_H , the motion is stopped and the heaters are turned off.

Returns:

0 invalid

1 valid

Available only for special-purpose mounts with temperature sensors.

Available from version 2.7.8.

:STOP# Halt all current movements, included tracking. If the mount is parked, parking or unparking it will be left in “parked” status. Otherwise, any movement command will return the mount to normal operation. Tracking can be restarted with the :AP# command.

Returns: nothing

Available from version 2.3.

:SwN# Set maximum slew rate to N degrees per second.

Returns:

0 invalid

1 valid

:SWOLN# Set the wake-on-LAN configuration to active (N=1) or inactive (N=0).

Returns:

0 invalid (also in case wake-on-LAN is not available)

1 valid

Available from version 2.15.7 (wake-on-LAN is available only for Q-TYPE2012 or Q-TYPE2016 electronic boxes).

:SWRLstring#

Set the wireless interface configuration.

To configure an hotspot (other devices will connect to the network created by the mount), the string will be formatted as follows:

1,**ssid,encryption,key,ip address,network mask**

To configure a client (the mount will connect to an existing network), and get the IP address from a DHCP server on the network, the string will be formatted as follows:

0,**ssid,encryption,key,DHCP**

To configure a client (the mount will connect to an existing network), and set the IP address manually, the string will be formatted as follows:

0,**ssid,encryption,key,ip address,network mask,gateway**

ssid – the network SSID, escaped as explained at page 76.

encryption – the string WEP (for WEP encryption) or WPA (for WPA-PSK encryption); only WEP is supported in hotspot mode until version 2.10.4 included, WPA is supported in hotspot mode from version 2.10.5.

key – the WEP key or WPA passphrase, escaped as explained at page 76.

ip address – the IP address of the mount

network mask – the network mask for the subnet

gateway – the IP address of the gateway

Returns:

the string "1#" if the configuration has succeeded

the string "0#" if the configuration failed

Available from version 2.9.8, only for mounts with the wireless connection option installed.

Note

If the parameters of the current connection are changed, it may happen that the connection is lost afterwards.

:SWRLC#

Shut down the wireless interface.

Available from version 2.12.3, only for mounts with the wireless connection option installed.

Returns:

the string "1#" if the configuration has succeeded

:SIPstring#

Set the LAN interface configuration.

To configure the LAN interface to get the IP address from a DHCP server, the string will be "1"

To configure the LAN interface to a fixed IP address, the string will be

"0,ip address,network mask,gateway"

Returns:

the string "1#" if the configuration has succeeded

the string "0#" if the configuration failed

Available from version 2.10

Note

If the parameters of the current connection are changed, it may happen that the connection is lost afterwards.

:SzDDD*MM# or :SzDDD*MM:SS# or :SzDDD*MM:SS.S#

Sets the target azimuth to DDD*MM (degrees, arcminutes), DDD*MM:SS (degrees, arcminutes, arcseconds), or DDD*MM:SS.S (degrees, arcminutes, arcseconds and tenths of arcsecond).

Returns:

0 invalid

1 valid

:SgeN# Set the guider port enable status. N is 1 to enable the guider port, 0 to disable the guider port. When the guider port is disabled, autoguide commands through the guider port are ignored.

Returns: nothing

Available from version 3.1.

Tracking Commands

:\$Q# Toggles the periodic error correction on and off. Has no effect if PEC training is active. Has no effect with the HPS mounts which don't feature PEC.

Returns: nothing

:p# Stops the periodic error correction. Has no effect with the HPS mounts which don't feature PEC.

Returns: nothing

:pP# Activates the periodic error correction. Has no effect with the HPS mounts which don't feature PEC.

Returns: nothing

:pR# Starts the periodic error correction training. Has no effect with the HPS mounts which don't feature PEC.

Returns: nothing

:pRX# Starts the periodic error correction training, where X has the following meaning:

0 short training (~15 minutes at sidereal speed)

1 medium training (~30 minutes at sidereal speed)

2 long training (~60 minutes at sidereal speed)

On an equatorial mount, only the R.A. axis is trained.

On an altazimuth mount, both axes are trained. The duration is proportionally longer due to the slower rates. Has no effect with the HPS mounts which don't feature PEC.

Returns: nothing

:pRaX# Starts the periodic error correction training of the altitude axis of an altazimuth mount, where X has the following meaning:

0 short training (~15 minutes at sidereal speed)

1 medium training (~30 minutes at sidereal speed)

2 long training (~60 minutes at sidereal speed)

The actual duration depends on the average rate on the altitude axis.

Has no effect with the HPS mounts which don't feature PEC.

Returns: nothing

:pRzX# Starts the periodic error correction training of the azimuth axis of an altazimuth mount, where X has the following meaning:

0 short training (~15 minutes at sidereal speed)

1 medium training (~30 minutes at sidereal speed)
2 long training (~60 minutes at sidereal speed)
The actual duration depends on the average rate on the azimuth axis.
Has no effect with the HPS mounts which don't feature PEC.
Returns: nothing

:T+# Increment custom tracking rate by 0.025 arcseconds per second of time.
Returns: nothing

:T-# Decrement custom tracking rate by 0.025 arcseconds per second of time.
Returns: nothing

:TL# Set lunar tracking rate
Returns: nothing

:TSOLAR# Set solar tracking rate
Returns: nothing

:TM# Set custom tracking rate
Returns: nothing

:TQ# Set default (sidereal) tracking rate
Returns: nothing

:TDDD.DDD#
Set custom tracking rate, where DDD.DDD is a decimal number which is four times the tracking rate expressed in arcseconds per second of time (see description of the :GT# command, Get tracking rate). This command should be issued before :TM#.

:STDDD.DDD#
Set the tracking rate to DDD.DDD, where DDD.DDD is a decimal number which is four times the tracking rate expressed in arcseconds per second of time.
Returns:
0 invalid
1 valid

:RT0# Set lunar tracking rate
Returns: nothing

:RT1# Set solar tracking rate
Returns: nothing

:RT2# Set default (sidereal) tracking rate
Returns: nothing

:RT9# Stop tracking
Returns: nothing

:RRsXXX.XXXX#
Set custom tracking rate in right ascension, where sXXX.XXXX is expressed in multiples of the sidereal speed. The rate is added to the standard sidereal tracking.
Returns:
1 valid
Available from version 2.7.8.

:RDsXXX.XXXX#
Set custom tracking rate in declination, where sXXX.XXXX is expressed in multiples of the sidereal speed.
Returns:
1 valid
Available from version 2.7.8.

:SdatN#
Configure the dual axis tracking setting. Use N=0 to disable the dual axis tracking, N=1 to enable the dual axis tracking. N=0 can be used only for equatorial mounts. This command does not start or stop tracking.
Returns:
1 valid
0 invalid
Available from version 2.13.9.
Note: up to version 3.1.5, using this command would actually cause tracking to start when the mount was in a stopped status. This has been corrected in version 3.1.6.

:Gdat#
Get the status of the dual axis tracking setting.
Returns:
0 dual axis tracking disabled (only for equatorial mounts)
1 dual axis tracking enabled
Available from version 2.13.9.

Precision Toggle

:U# Toggle between low and high precision modes. This controls the format of some values that are returned by the mount. In extended LX200 emulation mode, switches always to high precision (does not toggle).

Low precision: RA returned as HH:MM.T (hours, minutes and tenths of minutes), Dec/Az/Alt returned as sDD*MM (sign, degrees, arcminutes). Check the documentation of each command for details.

High precision: RA returned as HH:MM:SS (hours, minutes, seconds), Dec/Az/Alt returned as sDD*MM:SS (sign, degrees, arcminutes, arcseconds). Check the documentation of each command for details.

Returns: nothing

Note

From version 2.10, the 10micron mounts support also an ultra-precision mode that can set with :U2#. Using the :U# command in ultra-precision mode will switch back to high-precision mode.

The precision setting is specific to each single communication session. Each new communication session starts always as low-precision. See paragraph *Extended LX200 emulation and precision*.

:U0# Set low precision mode.

Returns: nothing

:U1# Set high precision mode.

Returns: nothing

:U2# Set ultra precision mode. In ultra precision mode, extra decimal digits are returned for some commands, and there is no more difference between different emulation modes.

Returns: nothing

Available from version 2.10.

Note

A Get Information command such as :GA# (Get altitude) can be used to detect the current precision setting, analysing the format of the response. See also the paragraph *Extended LX200 emulation and precision*.

Drive configuration

:SFA^tc^t.^{tt}# Sets the final approach time constant to the given value ^t.^{tt} (in seconds). The permitted range is 0.25 ... 5.00 seconds.

Returns:

E# function not supported

0# command failed (value out of range)

1# command successful

This function is supported only on the GM3000HPS and GM4000HPS mounts.

Available from version 2.14.21. From version 2.15 onwards, the value set with :SFA^tc is used only if the final approach mode is 1, and it is required to set the final approach mode to 0 with the command :SFA^md0# to use the default value.

:GFA^tc# Returns the final approach time constant used if the final approach mode is 1.

Returns:

E# function not supported

^t.^{tt}# the final approach time constant in seconds

This function is supported only on the GM3000HPS and GM4000HPS mounts.

Available from version 2.14.21. From version 2.15 onwards, this returns always the user-defined final approach time constant, and never 0.

:SFAImI.II#

Sets the distance limit for the final approach to the given value I.II in arcminutes, used if the final approach mode is 1. The permitted range is 0 ... 9.99 arcminutes. Use 0 to make always a final approach. The distance limit set with ::SFAIm is used only if the final approach mode is 1.

Returns:

E# function not supported
0# command failed (value out of range)
1# command successful

This function is supported only on the GM3000HPS and GM4000HPS mounts.
Available from version 2.15.

:GFAIm# Returns the final approach distance limit.

Returns:

I.II# the user-defined final approach distance limit in arcminutes. This value is used by the mount only if the final approach mode is 1. This function is supported only on the GM3000HPS and GM4000HPS mounts.

Available from version 2.15.

:SFAmdN# Sets the final approach mode according to the following values of N:

N = 0 use the standard configuration for time constant and distance limit;
N = 1 use the user-defined parameters for time constant and distance limit.

Returns:

E# function not supported
0# command failed (value out of range)
1# command successful

This function is supported only on the GM3000HPS and GM4000HPS mounts.

Note: enabling the user-defined parameters may result in the mount not reaching a completely settled state after slews. This depends on the characteristics of the load on the mount.

Available from version 2.14.21. In version 2.14.21, the behaviour was different: with N = 2 the mount made always a final approach, with N = 1 the mount made a final approach only if the distance was less than 2 arcminutes. Furthermore, the user-defined time constant was used also if mode = 0. From version 2.15 onwards, the behaviour is the one described here.

:GFAMd# Returns the final approach mode (see command :SFAmd for the meaning of the different values).

Returns:

E# function not supported
0# The final approach mode is standard
1# The final approach mode is custom

This function is supported only on the GM3000HPS and GM4000HPS mounts.

Available from version 2.14.21. In version 2.14.21 the mode had a different meaning (see :SFAMD) than in later versions (from 2.15 onwards).

Dome Control

:GDA# Gets the dome azimuth, if a dome is connected.

Returns: XXXX#

The current azimuth of the dome in tenths of degree from 0 to 3599. In case of error, returns 9999#.

Available from version 1.6.4.

Note: due to an error, in versions < 2.9.11 the azimuth is given with a 180 degrees offset.

:GDF# Gets the flap status of the dome, if a dome is connected.

Returns:

0# no dome connected

1# flap closed

2# flap open

3# flap moving

4# flap not detected

Available from version 2.14.17.

:GDH# Gets the homing operation status on the dome.

Returns:

0# no homing operation

1# homing operation in progress

2# homing operation completed

Available from version 2.7.4.

:GDS# Gets the shutter status of the dome, if a dome is connected.

Returns:

0# no dome connected

1# shutter closed

2# shutter open

3# shutter moving

4# shutter not detected

Available from version 1.6.4. Answer "4#" added since version 2.14.17.

:GDW# Gets the status of the slew operation for the dome. Use this command in place of :D# if you want to check if both the telescope and the dome have arrived at target. The result is valid only if the dome is under the control of the internal mount logic.

Returns:

0# no slew in progress, dome at internally computed target

1# slew in progress or dome not at internally computed target

Available from version 2.7.4.

Note

Any rotator connected to the mount which is slewing will also cause this command to return the string corresponding to the slew in progress state.

:GDw# Gets the status of the slew operation for the dome. The result is valid only if the dome is under external control via :SDA commands.

Returns:

0# no slew in progress, dome at manually set target

1# slew in progress, dome not at manually set target

Available from version 2.9.11.

:SDFn# Commands the dome flap, where n is an ASCII digit (1...2) with the following interpretation:

1 close flap

2 open flap

Returns:

0 failure

1 success (note: this means only that the command has been received, check the status of the flap with :GDF# for ensuring completion of the command)

Available from version 2.14.17.

:SDH# Starts homing on the dome. Note that this command succeeds even if no dome is connected. Please use :GDA# or :GDS# to check if a dome is connected.

Returns:

1 success – this does not mean that a dome is connected.

Available from version 2.7.4.

:SDMn# Sets the dome control, where n is an ASCII digit (0...2) with the following interpretation:

0 disconnect dome

1 dome on RS-232 port

2 dome on GPS port

Returns:

0 failure (if something else is connected to the selected port)

1 success – this does not mean that a dome is connected.

Available from version 1.6.4.

:SDSn# Commands the dome shutter, where n is an ASCII digit (1...2) with the following interpretation:

1 close shutter

2 open shutter

Returns:

0 failure

1 success (note: this means only that the command has been received, check the status of the flap with :GDF# for ensuring completion of the command)

Available from version 1.6.4.

:SDRXXXX#

Sets the dome radius to XXXX mm.
Returns: nothing
Available from version 1.6.4.

:SDTn# Sets the mount type for dome control, where n is an ASCII digit (1...2) with the following interpretation:

- 1 GM4000QCI/HPS with shoulders sticking out on the front side
- 2 GM4000QCI/HPS with shoulders sticking out on the back side

Returns: nothing
Available from version 1.6.4.

Note

This command has no effect on mounts other than the GM4000.

:SDUSS#

Sets the dome update interval to SS seconds (i.e. the dome is commanded to an updated position every SS seconds).
Returns: nothing
Available from version 1.6.4.

:SDXMsXXXX#

:SDYMsXXXX#

:SDZMsXXXX#

Set the mount position with respect to the centre of the dome, where sXXXX is the offset in mm towards North, East and the Zenith, respectively, measured from the centre of the base of the mount.
Returns: nothing
Available from version 1.6.4.

:SDXsXXXX#

:SDYsXXXX#

Set the position of the optical axis of the telescope relative to the declination mounting flange, where sXXXX is the measure in mm. :SDX specifies the distance from the flange to the optical axis (usually it is the radius of the optical tube), :SDY specifies the lateral displacement, measured from the centre of the mounting flange, positive towards right when looking from the back of the optical tube. Usually you will specify :SDY+0000# if the tube is not laterally displaced.
Returns: nothing
Available from version 1.6.4.

:SDAXXXX#

Slews the dome to the given azimuth (from 0 to 3600). This overrides the internal logic of the mount in order to give direct control of the dome azimuth to the controlling program. Setting any dome parameter from the keypad, or any of the following commands will give control again to the internal logic of the mount: :SDR, :SDT, :SDU, :SDXM, :SDYM, :SDZM, :SDX, :SDY, :SDAr.

Returns:

0 invalid (angle out of permitted range)

1 valid

Available from version 2.9.11.

:SDAr#

Release dome control to the internal logic of the mount.

Returns: nothing

Available from version 2.9.11.

Axis angular position commands

:GaXa# Gets the angular position on r.a. axis (for equatorial mounts) or azimuth axis (for altazimuth mounts).

Returns: sXXX.XXXX#

The current angular position, with sign, in degrees and decimals.

Available from version 2.9.9.

:GaXb# Gets the angular position on dec. axis (for equatorial mounts) or altitude axis (for altazimuth mounts).

Returns: sXXX.XXXX#

The current angular position, with sign, in degrees and decimals.

Available from version 2.9.9.

:SaXasXXX.XXXX#

Sets the target angular position on r.a. axis (for equatorial mounts) or azimuth axis (for altazimuth mounts).

Returns

0 invalid (angle out of acceptable range)

1 valid

Available from version 2.9.9.

:SaXbsXXX.XXXX#

Sets the target angular position on dec. axis (for equatorial mounts) or altitude axis (for altazimuth mounts).

Returns

0 invalid (angle out of permitted range)

1 valid

Available from version 2.9.9.

:QaXa# Gets the target angular position on r.a. axis (for equatorial mounts) or azimuth axis (for altazimuth mounts).

Returns: sXXX.XXXX#

The current target angular position, with sign, in degrees and decimals, as set by command :SaXa; if no target angular position has been set, returns the string "E#"

Available from version 2.9.9.

:QaXb# Gets the target angular position on dec. axis (for equatorial mounts) or altitude axis (for altazimuth mounts).

Returns: sXXX.XXXX#

The current target angular position, with sign, in degrees and decimals, as set by command :SaXb; if no target angular position has been set, returns the string "E#"

Available from version 2.9.9.

:MaX# Slews to the target angular positions sets with commands :SaXa and :SaXb and stops.

Returns:

0 no error

if the target is below the lower limit: the string

"1Object Below Horizon #"

if the target is above the high limit: the string

"2Object Below Higher #"

if the slew cannot be performed due to another cause: the string

"3Cannot Perform Slew #"

if the mount is parked: the string

"4Mount Parked #"

Note: even if commands :SaXa and :SaXb have been successful, it is not guaranteed that :MaX succeeds, since the target angles can be both within range when taken one at time but the resulting target position can be still outside limits (such as limits defined by the altitude above the horizon).

Available from version 2.9.9.

:PaX# Slews to the target angular positions sets with commands :SaXa and :SaXb and parks.

Returns:

"0#" no error

"1#" the target is below the lower limit

"2#" the target is above the high limit

"3#" the slew cannot be performed due to another cause

"4#" the mount is already parked

Note: even if commands :SaXa and :SaXb have been successful, it is not guaranteed that :PaX succeeds, since the target angles can be both within range when taken one at time but the resulting target position can be still outside limits (such as limits defined by the altitude above the horizon).

Available from version 2.9.9.

:PiP# Parks the mount in the current position.

Returns:

"0#" error

"1#" mount parked

Available from version 2.9.9.

:PsX# Slews to the saved park angular position and parks.

Returns:

"0#" no error

"1#" the target is below the lower limit

"2#" the target is above the high limit

"3#" the slew cannot be performed due to another cause

"4#" the mount is already parked

Available from version 2.9.9.

:PyX# Saves the current angular position as parking position for the :PsX# command.

Returns:

0 no error

0 error

Available from version 2.9.9.

Dithering Commands

:SditMRR,DD#

Sets the dithering amount to **RR** arcseconds in right ascension and to **DD** arcseconds in declination. Allowed range is from 0 to 30 arcseconds.

Returns:

"0#" if the command failed

"1#" if the command succeeded.

Available from version 2.14.

:SditTD,E,I#

Sets the dithering timer. **D** is the delay in seconds (from 0 to 356400), **E** is the exposure in seconds (from 0 to 356400), **I** is the interval in seconds (from 5 to 356400).

Returns:

"0#" if the command failed

"1#" if the command succeeded.

Available from version 2.14.

:SditS# Starts dithering.

Returns:

"0#" if the command failed (the mount is in a state that doesn't allow dithering).

"1#" if the command succeeded.

Available from version 2.14.

:SditQ# Stops dithering.

Returns:

"0#" if the command failed

"1#" if the command succeeded.

Available from version 2.14.

:SditN# Execute dithering immediately. Dithering must be active.

Returns:

"0#" if the command failed (dithering isn't active).

"1#" if the command succeeded.

Available from version 2.14.

Note: a bug in this command that could cause a systematic motion when repeating the dithering has been corrected in version 2.15.38.

:GditP# Gets the dithering parameters.

Returns:

R,D,L,E,I# a string where R is the dithering amount in right ascension in arcseconds, D is the dithering amount in declination in arcseconds, L is the delay in seconds, E is the exposure in seconds, I is the interval in seconds.

Available from version 2.14.

:GditS# Gets status of dithering.

Returns:

"0#" if dithering isn't active

"1#" if dithering is active.

Available from version 2.14.

Weather Station Commands

These commands return data provided by a weather station (or combined weather station / GPS module) connected directly to the GPS port of the control box, with the GPS port configured for GPS input. Data is received via a proprietary NMEA sentence.

:WSP# Get pressure datum in hPa from the weather station.

Returns:

XXXX.X,TTTT# if datum is available, where XXXX.X is the pressure in hPa, and TTTT is the time elapsed from the reading (in seconds);

E# if datum is not available (or it is older than 300s).

Available only for version>=2.15.11.

:WST# Get temperature datum in °C from the weather station.

Returns:

XXX.X,TTTT# if datum is available, where XXX.X is the temperature in °C (with sign) and TTTT is the time elapsed from the reading (in seconds);

E# if datum is not available (or it is older than 300s).

Available only for version>=2.15.11.

:WSH# Get humidity datum in % from the weather station.

Returns:

XXX.X,TTTT# if datum is available, where XXX.X is the humidity in % and TTTT is the time elapsed from the reading (in seconds);

E# if datum is not available (or it is older than 300s).

Available only for version \geq 2.15.11.

:WSD# Get dew point datum in °C from the weather station.

Returns:

XXX.X,TTTT# if datum is available, where XXX.X is the dew point in °C (with sign) and TTTT is the time elapsed from the reading (in seconds);

E# if datum is not available (or it is older than 300s).

Available only for version \geq 2.15.11.

:WSSN# Set mode of the automatic update of the refraction model data from the weather station. The automatic update feeds the data from the weather station into the refraction temperature and pressure settings of the mount. N can be one of the following digits:

0 do not update the refraction model data;

1 update only while the mount is not tracking;

2 update continuously, with a 15s smoothing filter.

Returns:

0 failure

1 success

Available only for version \geq 2.15.11.

:WSG# Get status of the automatic update of the refraction model from the weather station. The automatic update feeds the data from the weather station into the refraction temperature and pressure settings of the mount.

Returns:

N#, where N is the mode of the automatic update as follows:

0 do not update the refraction model data;

1 update only while the mount is not tracking;

2 update continuously, with a 15s smoothing filter.

Available only for version \geq 2.15.11.

Satellite Orbital Elements Commands

:TLEDLn#

Loads orbital elements for a satellite from the TLE database in the mount. n is the index of the orbital elements in the database, starting from 1 to the number returned by the TLEDN command.

Returns:

E# the mount database doesn't contain a TLE with the given index.

<two line elements># an escaped string containing the TLE data from the mount database which has been loaded. Lines are terminated by ASCII newline (ASCII code 10).

Available from version 2.13.20.

:TLEDN# Gets the number of TLEs in the mount database.

Returns:

n# the number of TLEs in the mount database.

Available from version 2.13.20.

:TLEG# Returns the currently-loaded two line elements.

Returns:

<two line elements># a string containing the two line elements. Lines are terminated by the ASCII newline character (ASCII code 10), and the entire string is escaped with the mechanism described in the "escaped strings" section below.

If no TLE is currently loaded, returns the string E#.

Available from version 2.13.20.

:TLELO<two line element>#

Loads satellite orbital elements in two-line format directly from the command protocol.

<two line element> is a string containing the two line elements. Each lines can be terminated by escaped newline (ASCII code 10), carriage return (ASCII code 13) or a combination of both. The first line may contain the satellite name. The entire string is escaped with the mechanism described in the "escaped strings" section below.

The TLE format is described here:

<https://www.celestrak.com/NORAD/documentation/tle-fmt.asp>

For example, loading the NOAA 14 element set of that page can be accomplished with:

```
:TLELONOA14.....$0a
1·23455U·94089A···97320.90946019···00000140··00000-0··10191-3·0··2621$0a
2·23455··99.0090·272.6745·0008546·223.1686·136.8816·14.11711747148495#
```

Returns:

E# invalid format

V# valid format

Available from version 2.13.20.

:TLEGAZJD#

Gets the apparent altazimuth coordinates of the satellite with the currently loaded orbital elements, computed for Julian Date JD (UTC).

Returns:

E# no TLE loaded

+AA.AAAA,ZZZ.ZZZZ# the apparent altazimuth coordinates, where +AA.AAAA is the altitude in degrees with decimals accounting for refraction, and ZZZ.ZZZZ is the azimuth in degrees with decimals.

Available from version 2.14.5.

:TLEGEQJD#

Gets the apparent equatorial coordinates of the satellite with the currently loaded orbital elements, computed for Julian Date **JD** (UTC).

Returns:

E# no TLE loaded

RR.RRRRR,+DD.DDDD# the apparent equatorial coordinates, where RR.RRRRR is the right ascension in hours with decimals, and +DD.DDDD is the declination in degrees with decimals.

Available from version 2.14.5.

:TLEP**JD,min**#

Precalculates the first transit of the satellite with the currently loaded orbital elements, starting from Julian Date **JD** (UTC) and for a period of **min** minutes, where **min** is from 1 to 1440.

Two-line elements have to be loaded with the :TLEL command.

Returns:

E# no TLE loaded or invalid command

N# no passes in the given amount of time

JDstart,JDend,flags# data for the first pass in the given interval. JDstart and JDend mark the beginning and the end of the given transit (UTC Julian Date). Flags is a string which can be empty or contain the letter F – meaning that mount will flip during the transit.

Available from version 2.13.20.

:TLES# Slews to the start of the satellite transit that has been precalculated with the :TLEP command.

Returns:

E# no transit has been precalculated

F# slew failed due to mount parked or other status blocking slews

V# slewing to the start of the transit, the mount will automatically start tracking the satellite.

S# the transit has already started, slewing to catch the satellite

Q# the transit has already ended, no slew occurs

Available from version 2.13.20.

:TLESCK# Gets the status of the slew to a precalculated satellite transit.

Returns:

V# slewing to the start of the transit

P# stopped at the start of the transit, waiting for the satellite

S# slewing to catch the satellite

T# tracking the satellite

Q# the transit has ended, not tracking

E# no slew to a satellite has been requested.

Available from version 2.14.22.

Additional Trajectory Commands

An arbitrary trajectory can be uploaded into the mount, defined as a sequence of altazimuth positions. The arbitrary trajectory may be used in place of a satellite tracking trajectory computed by the mount.

While following a satellite tracking trajectory or an arbitrary trajectory, it is possible to specify angular or time offsets to correct the position of the mount.

:TRNEW*JJJJJJ.JJJJJJ*#

Starts a new arbitrary trajectory definition. The trajectory will start on Julian Date *JJJJJJ.JJJJJJ* (UTC).

Returns:

V# ready to receive points

Available from version 3.0.

Note: the return value was incorrectly specified as *N#* in past versions of this document.

:TRADD*zzz.zzzzz,aa.aaaaa*#

Adds a new altazimuth position to the arbitrary trajectory definition, with azimuth *zzz.zzzzz*, altitude *aa.aaaaa*, in degrees and decimals. You can add up to a total of 900 positions. The time interval between adjacent positions is one second.

Returns:

E# invalid position, or number of positions exceeded.

nn# the total number of positions in the trajectory definition.

Available from version 3.0.

:TRP# Precalculates the movement to follow the loaded arbitrary trajectory. After this command you may start following the trajectory with the **:TLES#** command.

Returns:

N# no arbitrary trajectory defined, or the trajectory cannot be computed due to limits.

JDstart,JDend,flags# data for the trajectory. *JDstart* and *JDend* mark the beginning and the end of the trajectory (UTC Julian Date). *Flags* is a string which can be empty or contain the letter F – meaning that mount will flip during the trajectory.

Available from version 3.0.

Note: the return value in case of failure was incorrectly specified as *E#* in past versions of this document.

:TRREPLAY#

Precalculates the movement to follow the loaded arbitrary trajectory, but putting the start of the trajectory at the current time. Apart from this, it is the same as command **:TRP#**.

Returns:

E# no arbitrary trajectory defined

JDstart,JDend,flags# data for the trajectory. *JDstart* and *JDend* mark the beginning and the end of the trajectory (UTC Julian Date). *Flags* is a string which can be empty or contain the letter F – meaning that mount will flip during the trajectory.

Available from version 3.0.

:TROFFADD id ,+vvvv.v#

Adds an offset while following the current trajectory (satellite or arbitrary trajectory). The amount of the offset is +vvvv.v. The type of offset is identified by the numeric code id as follows.

id	meaning
1	Offset along the first mount axis (right ascension if equatorial mount, azimuth if altazimuth mount), expressed in arcseconds in the range $-1800.0 \dots +1800.0$.
2	Offset along the second mount axis (declination if equatorial mount, altitude if altazimuth mount), expressed in arcseconds in the range $-1800.0 \dots +1800.0$.
3	Offset along the first mount axis (right ascension if equatorial mount, azimuth if altazimuth mount), multiplied by $\cos(\text{dec})^{-1}$ or $\cos(\text{altitude})^{-1}$ to keep the angular offset constant in the sky, expressed in arcseconds in the range $-1800.0 \dots +1800.0$.
4	Time offset, expressed in milliseconds in the range $-1000.0 \dots 1000.0$.

Returns:

E# invalid offset code, or invalid range, or the mount is not following a trajectory.

V# offset applied.

Available from version 3.0.

:TROFFGET id #

Gets the offset used while following the current trajectory. See the description of command :TROFFADD for the meaning of offset ids.

Returns:

E# invalid offset code

+vvvv.v# the offset for the given id .

Available from version 3.1.4.

This command was described as available since version 3.0, but this was not correct.

:TROFFSET id ,+vvvv.v#

Sets an offset while following the current trajectory. The current value of the offset is replaced with the new value. See the description of command :TROFFAD for the meaning of offset ids.

Returns:

E# invalid offset code, or invalid range, or the mount is not following a trajectory.

V# offset applied.

Available from version 3.0.

:TROFFCLR#

Clears all offsets while following the current trajectory.

Returns:

E# the mount is not following a trajectory.

V# offsets cleared.

Available from version 3.0.

Network Discovery and Web Interface Commands

:NTGdisc# Returns information about the discovery service.

Returns:

0# Discovery service not available.

0,name# Discovery service available and not active, with given name.

1,name# Discovery service available and active, with given name.

Available from version 2.16.5

:NTSdiscN#

:NTSdiscN,name#

Setup the discovery service with given name (optional). N must be set to 0 to deactivate the discovery service, 1 to activate the discovery service.

Returns:

E# Invalid values.

1# Configuration done.

Available from version 2.16.5

:NTGweb# Return information about the web interface.

Returns:

0# Web interface not available or not active.

1# Web interface active.

Available from version 2.16.5

:NTSwebN#

Setup the web interface. N must be set to 0 to deactivate the web interface, 1 to activate the web interface.

Returns:

E# Invalid value or web interface not available.

1# Configuration done.

Available from version 2.16.5

Rotator and Focuser Commands

:FocQmax# Get the maximum index where a rotator/focuser can be found.

Returns:

N# a string containing the maximum index that can be used in focuser/rotator commands.

:FocIN# Get information about focuser N.

Returns:

a string formatted as follows

name,type,serial#

or # in case of error.

:FocQN# Queries focuser N.

Returns:

0# focuser not available

1# focuser available

:FocGuFN# Reads the position of focuser N (in μm).

Returns:

+XXXXXX# a string containing the position in μm , with sign.

:FocSuFN,+PPPPPP#

Sets the destination position of focuser N to +PPPPPP (in μm).

Returns:

1 valid

0 invalid

:FocSSN#

Starts motion of focuser N towards the destination position.

Returns:

0 invalid

1 valid

:FocGuFN# Reads the destination position of focuser N (in μm), not temperature compensated

Returns:

+XXXXXX# a string containing the destination position in μm , with sign.

:FocDN# Queries the moving status of focuser N.

Returns:

0 the focuser is at the destination.

1 the focuser is moving towards the destination.

:FocdN# Queries the status of focuser N.

Returns: a number depending on the current status of the focuser, terminated by a '#', with the following meaning:

0# The focuser is stopped.

1# The focuser is tracking.

2# The focuser is in a manually controlled motion.

3# The focuser is slewing to a target position.

4# The focuser is slewing to the home position.

5# The focuser is going to stop.

6# The focuser has lost the position and needs homing.

97# The focuser has been stopped due to overload.

98# Unknown status.

99# Error or invalid focuser index.

Available from version 3.0.

:FocTN# Reads the temperature of focuser N.

Returns:

+TT.T# a string containing the temperature in °C, with sign.

:FocSmsN,DDDDD#

Sets the focuser N maximum speed to DDDDD µm/s. Allowed values must be in the range returned by :FocGmsr. Returns:

1 valid

0 not valid (outside allowed range or invalid focuser index)

:FocGmsN#

Gets the focuser N maximum speed in µm/s. Returns:

DDDDD# the focuser maximum speed in µm/s.

E# error

:FocGmsrN#

Gets the focuser N maximum speed range in µm/s. Returns:

dddd,DDDDD# the minimum and maximum configurable speed

E# error

:FocGvN#

Gets the validity of the position of focuser N.

Returns:

1 valid (an homing procedure has been completed, or the position was recovered from the backup battery)

0 not valid (the position was lost)

:FocSZN,+MMMMM,+NNNNN#

Sets the range for focuser N. +MMMMM, +NNNNN are the minimum and maximum value of the allowed positioning range for the rotator, in µm, with respect to the homing position of the focuser. +NNNNN must be at least 1000 µm greater than +MMMMM, and both must fit into the full mechanical range of the rotator.

Returns:

0 invalid

1 valid

:FocGZN# Gets the range for focuser N.

Returns:

+MMMMM,+NNNNN# a string containing the minimum (+MMMMM) and maximum (+NNNNN) of the allowed positioning range for the focuser, in µm, with respect to the homing position of the focuser.

E# error (rotator not available).

:FocHSN# Starts homing for focuser N. Note: if the range has been restricted away from the zero position, the homing procedure will always fail.

Returns:

0 invalid (can't start homing in this status)
1 valid (homing procedure started)

:FocHGN# Gets the homing operation status for focuser N.

Returns:

0# no homing operation or homing operation failure
1# homing operation in progress
2# homing operation completed successfully

:FocSsN,+DDDDDD#

Start moving the focuser N at +DDDDDD μ m per second (limited to the maximum speed).

Returns:

0 invalid
1 valid

:FocSqN#

Stops moving the focuser N.

Returns:

nothing

:F+# Start motion of focuser 1 inward (toward optical system).

Returns: nothing

:F-# Starts motion of focuser 1 outward (away from optical system).

Returns: nothing

:FF# Sets speed of focuser 1 to fastest setting.

Returns: nothing

:FS# Sets speed of focuser 1 to 0.01 mm/s.

Returns: nothing

:FQ# Halts motion of focuser 1

:Fn# Sets speed of focuser 1 accordingly to the following table for n.

n	Focuser 1 speed
1	0.01 mm/s
2	0.1 mm/s
3	1 mm/s
4	the fastest speed attainable with the focuser

Returns: nothing

:RotQmax# Get the maximum index where a rotator/focuser can be found.

Returns:

N# a string containing the maximum index that can be used in focuser/rotator commands.

:RotIN# Get info about rotator N.

Returns:

a string formatted as follows

name,type,serial#

or # in case of error.#

:RotQN# Queries rotator N.

Returns:

0 rotator not available

1 rotator available

:RotGMN# Reads the angle of rotator N in the mechanical system (0° = fixed reference with respect to the rotator mechanics).

Returns:

+PPP.PPPP# the position of the rotator in degrees.

E# error (rotator not available).

:RotGRN# Reads the angle of rotator N in the equatorial system (0° = north).

Returns:

+PPP.PPPP# the position of the rotator in degrees.

E# error (rotator not available).

:RotGON# Reads the angle of rotator N in the optical system (0° = fixed reference with respect to the optical system; it coincides with the mechanical system if the rotator is fixed directly to the telescope).

Returns:

+PPP.PPPP# the position of the rotator in degrees.

E# error (rotator not available).

:RotSofN,+ZZZ.ZZZZ#

Sets the offset used in the display of the position angle of rotator N on the keypad to +ZZZ.ZZZZ degrees.

Returns:

1 valid

0 invalid (outside allowed range or invalid rotator index)

Available from version 3.0.

Note: this offset is used only for values displayed or entered on the keypad.

:RotGofN# Gets the offset used in the display of the position angle of rotator N on the keypad.

Returns:

+ZZZ.ZZZZ# the offset in degrees

E# error

Available from version 3.0.

Note: this offset is used only for values displayed or entered on the keypad.

:RotSmZN#

Sets the offset used in the display of the position angle of rotator **N** on the keypad to the mechanical readout angle, so that the mechanical position angle on the keypad reads zero. Also sets the destination angle of rotator **N** in the mechanical system to the current mechanical readout angle.

Returns:

V# valid

E# error (invalid rotator index)

Available from version 3.0.

Note: this offset is used only for values displayed or entered on the keypad.

:RotSeZN#

Sets the offset used in the display of the position angle of rotator **N** on the keypad to the equatorial position angle, so that the equatorial position angle on the keypad reads zero. Also sets the destination angle of rotator **N** in the equatorial system to the current equatorial position angle.

Returns:

V# valid

E# error (invalid rotator index)

Available from version 3.0.

Note: this offset is used only for values displayed or entered on the keypad.

:RotSmsN,DDD#

Sets the rotator **N** maximum speed to **DDD** deg/s. Allowed values must be in the range returned by :RotGmsr. Returns:

1 valid

0 not valid (outside allowed range or invalid rotator index)

:RotGmsN#

Gets the rotator **N** maximum speed in deg/s. Returns:

DDD# the rotator maximum speed in deg/s.

E# error

:RotGmsrN#

Gets the rotator **N** maximum speed range in deg/s. Returns:

ddd,DDD# the minimum and maximum configurable speed

E# error

:RotGvN#

Gets the validity of the position of focuser **N**.

Returns:

- 1 valid (an homing procedure has been completed, or the position was recovered from the backup battery)
- 0 not valid (the position was lost)

:RotSmN,+DDD.DDDD#

Sets the destination angle of rotator N to +DDD.DDDD degrees in the mechanical system (0° = fixed reference with respect to the rotator mechanics).

Returns:

- 0 invalid
- 1 valid

:RotSrN,+DDD.DDDD#

Sets the destination angle of rotator N to +DDD.DDDD degrees in the equatorial system (0° = north).

Returns:

- 0 invalid
- 1 valid

:RotSoN,+DDD.DDDD#

Sets the destination angle of rotator N to +DDD.DDDD degrees in the optical system (0° = fixed reference with respect to the optical system; it coincides with the mechanical system if the rotator is fixed directly to the telescope).

Returns:

- 0 invalid
- 1 valid

:RotGmN#

Gets the destination angle of rotator N in the mechanical system.

Returns:

- +PPP.PPPP# the destination angle of rotator N in degrees.
- E# error (rotator not available).

:RotGrN#

Gets the destination angle of rotator N in the equatorial system.

Returns:

- +PPP.PPPP# the destination angle of rotator N in degrees.
- E# error (rotator not available).

:RotGoN#

Gets the destination angle of rotator N in the optical system.

Returns:

- +PPP.PPPP# the destination angle of rotator N in degrees.
- E# error (rotator not available).

:RotSSmN#

Starts rotation of rotator **N** towards the destination in the mechanical system.

Returns:

0	invalid
1	valid

:RotSSo**N**#

Starts rotation of rotator **N** towards the destination in the optical system.

Returns:

0	invalid
1	valid

:RotSSr**N**#

Starts rotation of rotator **N** towards the destination in the equatorial system.

Returns:

0	invalid
1	valid

:RotD**N**# Queries the slewing status of rotator **N**.

Returns:

0	the rotator is at the destination (stopped or tracking) or in a manual motion.
1	the rotator is moving towards the destination (slewing, homing, etc.).

:Rotd**N**# Queries the status of rotator **N**.

Returns: a number depending on the current status of the rotator, terminated by a '#', with the following meaning:

0#	The rotator is stopped.
1#	The rotator is tracking.
2#	The rotator is in a manually controlled motion.
3#	The rotator is slewing to a target position.
4#	The rotator is slewing to the home position.
5#	The rotator is going to stop.
6#	The rotator has lost the position and needs homing.
97#	The rotator has been stopped due to overload.
98#	Unknown status.
99#	Error or invalid rotator index.

Available from version 3.0.

Note

Numbers **differ** from the numbers returned by :Gstat# for the mount operation.

:RotAN,**T**# Sets the automatic mode for slaving the pointing of rotator **N** to the mount pointing. Use **T** accordingly to the following table.

T	Slave mode
0	No slaving mode – the rotator stays at the same mechanical angle when slewing the mount.
1	The rotator is rotated to the destination angle* relative to the equatorial system.

2 The rotator is rotated to the destination angle* relative to the optical system.

3 The rotator is rotated to the destination angle* relative to the mechanical system.

* The angles are set with commands :RotSm, :RotSr, :RotSo. If they haven't been set, they are supposed to be zero. If the rotator is set for a reduced range (such as 0°/180° or 0°/90°), these angles can be taken mod 180° or mod 90°.

Returns:

0 invalid
1 valid

:RotTN,T# Sets the tracking mode for rotator N. Use T accordingly to the following table.

T Tracking mode

0 The rotator is stopped (fixed with respect to the mechanical system).

1 The rotator tracks in the equatorial system.

2 The rotator tracks in the optical system.

Returns:

0 invalid
1 valid

:RotGAN# Gets the automatic mode for rotator N.

Returns:

E# invalid rotator

A# The number A, where A is the automatic mode for rotator N (see command :RotA).

:RotGTN# Gets the tracking mode for rotator N.

Returns:

E# invalid rotator

T# The number T, where T is the tracking mode for rotator N (see command :RotT).

:RotSFTN,J#

Sets the focus type for rotator N. Use J accordingly to the following table.

J Focus type

0 Cassegrain / refractor / newtonian

1 Nasmyth - rotator attached to the mount body

2 Cassegrain / refractor /newtonian with reversed direction

3 Nasmyth - rotator attached to the mount body with reversed direction

Returns:

0 invalid
1 valid

:RotGFTN#

Gets the focus type for rotator N.

Returns:

J# a string containing the focus type (see command :RotSFT for the meaning of T).

E# error (rotator not available).

:RotSZN,+MMM,+NNN#

Sets the range for rotator N. +MMM, +NNN are the minimum and maximum value of the allowed rotation range for the rotator, in degrees, with respect to the mechanical system of the rotator. +NNN must be at least 90° greater than +MMM, and both must fit into the full mechanical range of the rotator.

Returns:

0 invalid
1 valid

:RotGZN# Gets the range for rotator N.

Returns:

+MMM,+NNN# a string containing the minimum (+MMM) and maximum (+NNN) of the allowed rotation range for the rotator, in degrees, with respect to the mechanical system of the rotator.

E# error (rotator not available).

:RotHSN#Starts homing for rotator N.

Returns:

0 invalid (can't start homing in this status, or rotator not available)
1 valid (homing procedure started)

:RotHGN# Gets the homing operation status for rotator N.

Returns:

0# no homing operation or homing operation failure
1# homing operation in progress
2# homing operation completed successfully
E# error (rotator not available).

:RotSsN,+DD.DDDD#

Start moving the rotator N at +DD.DDDD degrees per second (limited to the maximum attainable physical speed).

Returns:

0 invalid
1 valid

:RotSqN#

Stops moving the rotator N.

Returns:

nothing

Other Commands

:EMUAP# Sets Astro-Physics compatible emulation mode.

Returns: nothing

Note: in ultra-precision mode, there is no difference between emulation modes.

:EMULX# Sets LX200 emulation mode.

Returns: nothing

Note: in ultra-precision mode, there is no difference between emulation modes.

:startlog# Starts a log of the commands received by the mount.

Returns: nothing

:stoplog# Ends the communication log.

Returns: nothing

:shutdown# Shuts down the mount. Available only with electronics model 2012 or above.
Do NOT remove the power after receiving a successful answer.

Returns:

0 failure

1 success

Available from version 2.9.2.

:getlog# Gets the communication log.

Returns: a text containing the communication log (up to 256Kbytes).

:evlog# Gets the event log.

Returns: a text containing the communication log (up to 3Kbytes).

Available from version 2.7.8.

:USEROK#

Allow the mount to move, after an inconsistency has been signaled. See also the :Gstat# command.

Returns: nothing

Available from version 2.8.13.

:USERWAIT#

Stops the mount and waits the user to send the :USEROK# command or authorize movements again using the keypad. You can use this command to block the mount in case your system detects some inconsistency. See also the :Gstat# command.

Returns: nothing

Available from version 2.8.13.

:GETID#

Gets an unique hardware identifier for the mount. The identifier does not change (unless the mount is serviced, which could lead to a different identifier).

This command can be use to detect if different connections (i.e. a serial port and a LAN connection) are actually a connection to the same mount.

Returns: a 20-digit (64-bit) number terminated by # unique for each mount.

XXXXXXXXXXXXXXXXXXXXX#

Available from version 2.9.11.

:NUTimsXXX#

Adjust the time of the mount of the given amount in milliseconds, from +999 to -999.

Returns:

"0#" if the command failed

"1#" if the command succeeded.

Available from version 2.10.

:STOPPH# Stop the pre-heating procedure if it is underway.

Returns: nothing

Available from version 3.1.

Extended LX200 emulation and precision

The mount can be configured to use an "Extended LX200 emulation" which modifies the return value of some commands, allowing operation with some software designed for Astro-Physics mounts, or to standard LX200 emulation in order to allow operation with some software designed for Meade LX200 mounts. These modes can be set using the :EMUAP# or :EMULX# commands. While the commands highlighted in green are always available, when the extended LX200 emulation mode is active the behaviour of some commands is modified. Furthermore, some commands will have different behaviour depending on the precision (low precision, high precision or ultra precision).

The precision setting pertains to the specific communication session. This means that each of the two serial ports and each single LAN connection begin with the default low-precision and can be independently configured with different precision settings. In this way, different programs with different precision requirements can be used at the same time.

When ultra-precision is set, all commands will behave in the same way regardless of the emulation chosen. The ultra precision has been added to firmware 2.10, so if you are writing software that will not be used with previous versions, our advice is to use the ultra-precision mode by sending the :U2# command (after checking the firmware version).

All differences in the command usage are described in each command.

Notes on pointing objects

The mount has a series of commands used to point objects. Usually you want to point a specific position in the sky with its equatorial coordinates (right ascension and declination). In this case, use the :Sr and :Sd commands to set the target coordinates. After doing that, you can use the :Ga and :Gz commands to get the current altitude and azimuth of the target (changing with time). You can also use the :GTTRK command to know if the target is located at a position where the mount will be able to track or not (the altazimuth mount cannot track objects above 89 degrees altitude).

The :MS command will slew the mount to the target coordinates.

If you use the :MA command, the scope will slew at the current altazimuth coordinates of the target and will remain there.

The situation is reversed if you use the :Sa and :Sz commands to set the target altazimuth coordinates. After doing that, the :Gr and :Gd commands return the current right ascension and declination of the target (changing with time). The :MA command will slew the mount to the target coordinates. The :MS command will calculate the current equatorial coordinates of the target and slew to them, and then tracking will be active.

The mount will not cope with different input coordinates from multiple sources, i.e. the keypad, virtual keypad, the serial connections and the network connection. Sending a command such as :Sr, :Sd, :Sa, :Sz to one of these interfaces will overwrite the coordinates set with the others.

Notes on pointing state

The "pointing state" of the mount refers to what was formerly called "side of the pier" (sometimes called also "meridian side")¹. For a german equatorial mount there are two pointing states. In Fig. 1 we have a representation of the sky, where the two pointing states are highlighted in blue and orange colours. When the mount has to move to a point of the sky highlighted in a different colour, it has to pass through the pole: this is called the "meridian flip". This is to ensure that the telescope stays always above the counterweight bar, instead of hitting the pier or the tripod. 10micron german equatorial mounts allow also to slew and track in the alternative pointing state, with the telescope below the counterweight bar, for a configurable angle. The zones of the sky where this is possible are highlighted with stripes of both colors.

For historical reasons, the blue pointing state is called "East" (because the telescope is on the east side of the pier, looking west), and the orange pointing state is called "West" (because the telescope is on the west side of the pier, looking east). "East#" and "West#" are also the strings returned by command :pS#.

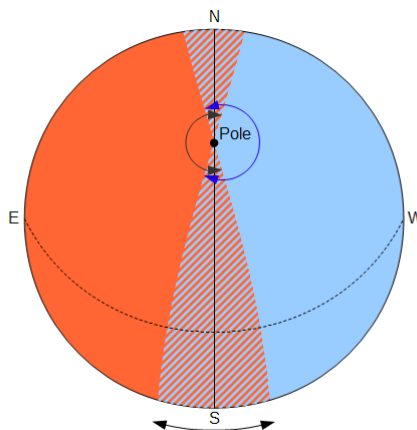


Fig. 1: Pointing states for a german equatorial mount.

¹ A detailed explanation with examples for a german mount is also available in the *ASCOSideOfPier Behaviour* document, available from the ASCOM standard website in the developer components.

Pointing state	String returned by :pS#
Blue	East#
Orange	West#

For the AZ1000HPS ... AZ4000HPS altazimuth mounts there are also two pointing states, but the representation of the sky is different, and it depends on the way the mount is assembled on the pier. Fig. 2 shows the pointing states in the case the position of the azimuth adjustment knobs is east or west, while Fig. 3 shows the pointing states in the case it is north or south. When the mount has to move to a point of the sky highlighted in a different colour, it has to pass through the zenith. This is called a "pointing flip". Since there is no concept of the telescope being above or below the counterweight bar, the zones where it is possible to point in both pointing states are larger. This makes the pointing flip a much rarer occurrence than in an equatorial mount. The string returned by the :pS# command depends on the pointing state and also on the position of the azimuth adjustment knobs.

Azimuth side	Pointing state	String returned by :pS#
North	Blue	East#
	Orange	West#
East	Blue	West#
	Orange	East#
South	Blue	West#
	Orange	East#
West	Blue	East#
	Orange	West#

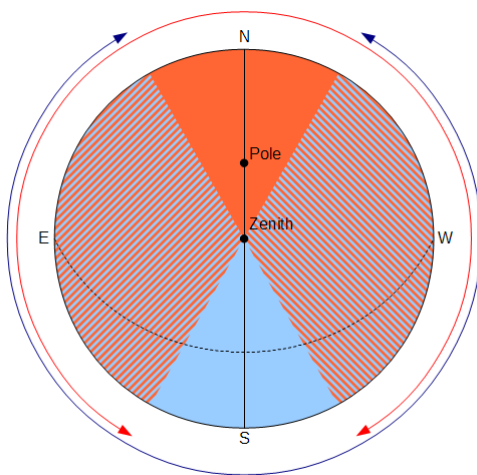


Fig. 2: Pointing states for AZ1000HPS ... AZ4000HPS (azimuth adjustment knobs east or west).

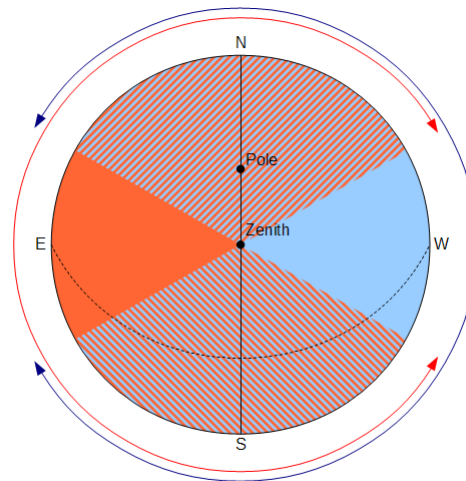


Fig. 3: Pointing states for AZ1000HPS ... AZ4000HPS (azimuth adjustment knobs north or south).

Notes on building an alignment model

Starting with firmware version 2.8.15, the mounts have a series of commands used to make and manipulate the alignment model. The :CMS command works by matching the target set by the :Sr and :Sd commands to the current position of the mount as an alignment star.

Furthermore, the :CM/:CMR commands can be set to have the same function of the :CMS command, by sending the command :CMCFG1# or activating the "Sync Refines" function with the keypad.

If you are starting with no alignment model (see the :getalst and :delalig commands), the first three position you match in this way act as a 3-Stars alignment. The following stars act as successive Refining commands given by the keypad.

These commands allow synchronizing the mount with a plate-solved position. You can then automate the alignment procedure by slewing to a sequence of target positions. In each position, take a picture and find the position of the centre of the field of view by plate solving. Send the position to the mount with the :Sr and :Sd commands, then give the :CMS command to add the point (also the :CM or :CMR command, if the "Sync Refines" function on the keypad is active).

You can also check the alignment model using the :getali/:getalp command, which gives you the position and error of each alignment star. You may also delete a star from the alignment model with the :delalst command.

It is important to check the possible error given by all these commands. In particular, the :delalst command may return an error due to the impossibility of calculating a new model after the star is removed. In this case, in order to delete the star from the model you will have to either delete the entire alignment model with the :delalig command, or add another star to the model before continuing.

All these commands are described in the "Sync Control" section.

Entering an alignment model

Starting from firmware 2.10.6, it is possible to enter into the mount an alignment model specification without the need of using the synchronization commands. In this case, instead of synchronizing the mount on each alignment point, you will send all the relevant data and compute the alignment model in one pass.

Each alignment point will be defined by a position in the sky (usually a well-known star, or the centre of an exposure that has been astrometrically plate-solved using a dedicated tool), and by the mount-reported coordinates.

The alignment specification is to be created as follows. First, you have to begin a new alignment specification by sending the command :newalig#. This will NOT clear the alignment model the mount is using, but *only* the alignment points for building the next model. Then you will add in sequence all the alignment points with the :newalpt command. For each point, you need the mount-reported coordinates (right ascension, declination and pier side), the plate-solved coordinates (i.e. the "real" coordinates the telescope is pointing at), and the sidereal time of the match. When you have added all points, use the :endalig# command to compute a new alignment model.

Note that it is not necessary to clear the previously loaded alignment model before entering the new alignment points. Thus it is possible to create a new alignment model while still using an alignment model that was previously computed. On the other hand, it is extremely important that the alignment model is not *changed* between starting the measurement of the alignment points and computing the new model with :endalign#.

Assisted polar axis alignment and orthogonality correction

Since firmware 2.15.16, two slew commands, :MSap# and :MSao#, can be used to align the polar axis of the mount and to clear the orthogonality error of the telescope. They allow using the functionality already available with the "Polar Align" and "Ortho Align" commands of the keypad / virtual keypad with the command protocol.

When you slew to a target using these two commands, they will miss it by a computed amount. In case of :MSap#, the amount is computed so that, when the target is centered using the polar axis manual movements, the polar axis is aligned with the celestial pole. In case of :MSao#, the amount is computed so that, when the target is centered by correcting the orthogonality between the telescope and the mount, the optical axis of the telescope is orthogonal to the declination/altitude axis. :MSap# is available only for equatorial mounts.

Both commands require that a valid model is active, with at least two stars for :MSap# and at least three stars for :MSao#. After changing the physical position of the polar axis, or the orthogonality of the telescope, the model becomes invalid and should be cleared with the :delalign# command, then a new model should be built.

Polar axis alignment

This can be done only with an equatorial mount. The sequence of operations for performing the polar axis alignment should be the following.

1. Build a model with at least two stars (a few more is much better).
2. Select a target for doing the polar axis alignment. You should avoid places in the sky where the manual movements have no or little effect on the direction the telescope is pointing at. So you need to stay away from the zenith, where the azimuth movement has no effect, and from the E-W directions, where the altitude movement has no effect. At least 30 degree of distance from these directions is advised.
3. Compute the JNOW coordinates of the target – remember that besides correcting for precession, light aberration etc. you should also consider the proper motion for stars. Some stars have significant proper motions (for example Arcturus has a proper motion of ~2.2 arcseconds per year).
4. Send the coordinates to the mount with the :Sr and :Sd commands (use at least arcsecond precision).
5. Disable the dual axis tracking with :Sdat0#. Keeping the dual axis tracking may lead to errors depending on the tracking correction imposed by the model.
6. Issue the :MSap# command. The mount will slew to the target (it will miss it by some amount).

7. Use the manual movements of the polar axis to adjust its azimuth and altitude until the target is centered.
8. Clear the alignment model with :delalig#, then build a new model.

Orthogonality axis alignment

The sequence of operations for performing the polar axis alignment should be the following.

1. Build a model with at least three stars (a few more is much better).
2. Select a target for doing the orthogonality correction. There are no particular requirements on the location of the target.
3. Compute the JNOW coordinates of the target – remember that besides correcting for precession, light aberration etc. you should also consider the proper motion for stars. Some stars have significant proper motions (for example Arcturus has a proper motion of ~2.2 arcseconds per year).
4. Send the coordinates to the mount with the :Sr and :Sd commands (use at least arcsecond precision).
5. Issue the :MSao# command. The mount will slew to the target (it will miss it by some amount).
6. Adjust the position of the telescope on the mounting flange by operating on the retention rings, shims or equivalent until the target is centered.
7. Clear the alignment model with :delalig#, then build a new model.

Rotators and Focusers

Some 10micron mounts allow to connect one or more rotators/focusers that can be controlled directly with the command protocol. Each device is given an index, starting from 1, depending on the position it has been connected to. The maximum index can be queried by command :RotQmax# or :FocQmax# (they return the same result). Some devices are a combination of rotator and focuser, so the rotator and focuser part will appear at the same index. Some devices are just a rotator or a focuser. It is possible to detect available devices by the commands :RotQN# and :FocQN#. If a rotator/focuser is connected at index 1, :RotQ1# and :FocQ1# will both return 1. If a rotator without focuser is connected at index 1, :RotQ1# will return 1 and :FocQ1# will return 0. Do not assume that if you do not detect a rotator/focuser at index 1, then there is no rotator/focuser connected. They are not necessarily in sequence.

You can query a device by using command :FocI# and :RotI# (they return the same result). The serial number can be used as an identifier should the device be connected in a different position (assuming thus a different index).

The position of a rotator can be queried and set in different reference systems, which depend on the configuration of the focus type and the position the telescope is pointing at. The *mechanical* reference system is the mechanical angle of the rotator with respect to the mounting flange. This angle changes by driving the motor. The *optical* reference system is the mechanical angle of the rotator with respect to the image plane of the telescope. It will match the mechanical reference system if the rotator is mounted directly on the optical tube, but sometimes it may happen that in a Nasmyth focus configuration the rotator is not attached to the optical tube, but it is attached to the

fork, so it does not rotate with the telescope when it changes altitude. This can be set by the “Nasmyth” focus type configuration. The *equatorial* reference system is the angle between the zero position of the rotator and the north direction in the image plane of the telescope. It will change when the telescope moves, based on the mount model. The rotator can be set to track automatically this angle (i.e. to move so that the equatorial angle remains constant) when tracking or slewing, which is the configuration needed to compensate for field rotation.

It is important to check that the position of a rotator or focuser is valid before attempting motion. Rotators and focusers have an home position which can be searched for with commands :RotHS (start homing) and :RotHG (query homing state) for rotators, and :FocHS and :FocHG for focusers. It is not necessary to perform an homing procedure each time the rotator or focuser is switched on, because the motor position is kept in memory thanks to a backup battery. If the position is lost due, for example, to an exhausted backup battery, the rotator/focuser will not move and you will have to perform an homing procedure. You can check whether the position is valid by using commands :FocGv and :RotGv.

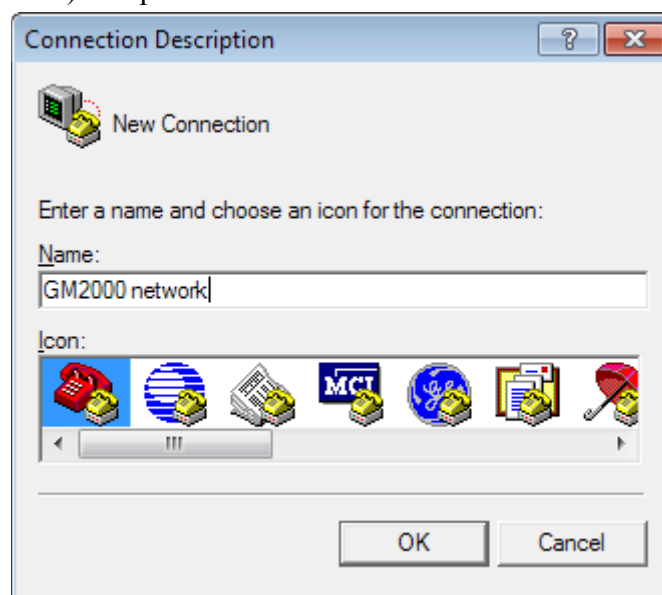
Escaped strings

Some strings are escaped in order to communicate with the mount. The escaping scheme is the following:

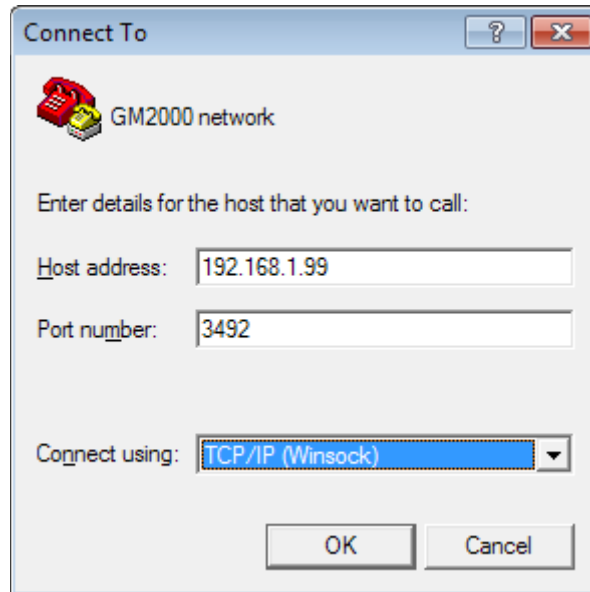
- the character '\$' is escaped as the sequence "\$\$";
- the character '#' is escaped as the sequence "\$23";
- the character ',' is escaped as the sequence "\$2C";
- any character with ASCII code less than hexadecimal 20 or greater than hexadecimal 7E is escaped as '\$XX' where XX is the hexadecimal ASCII code;
- any other character is passed unchanged.

Example of a HyperTerminal Session

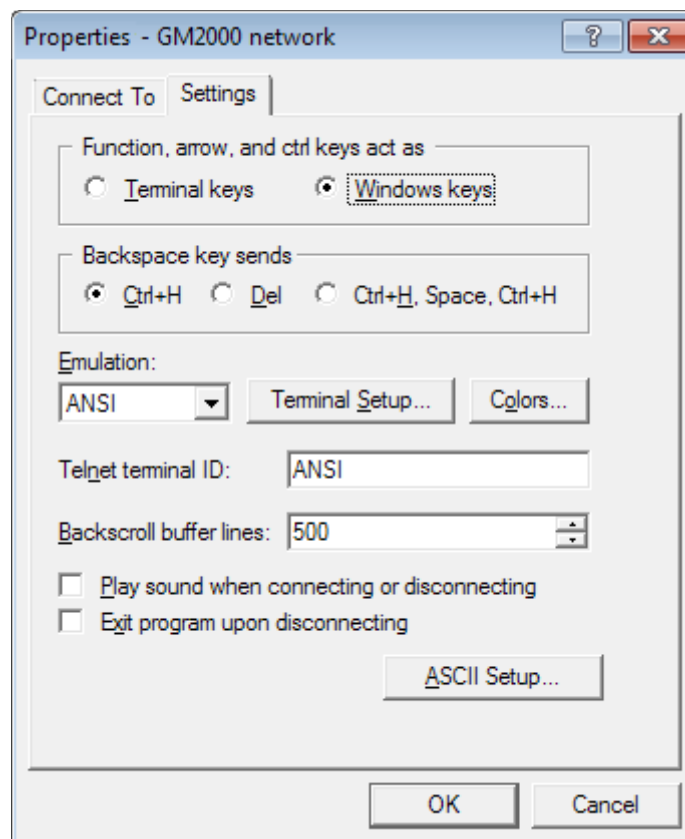
The following example shows how to connect to the mount through the Ethernet using HyperTerminal Private Edition version 6.3. Create a new connection, enter the connection name (such as "GM2000 network") and press OK.



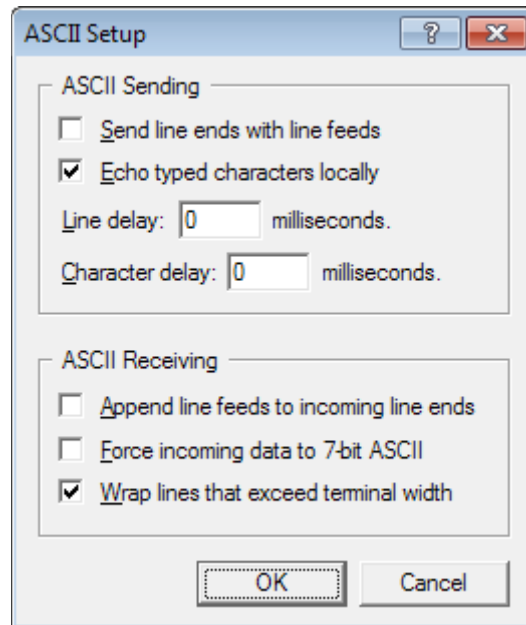
In the next window choose "TCP/IP (Winsock)", then enter the Host address (such as 192.168.1.99, depending on your mount configuration) and the port number (3490 or 3492).



The terminal will connect (look for "Connected" in the status bar). Now you should choose from the menu File → Properties, then the "Settings" tab. Choose "Function, arrow, and ctrl keys act as" "Windows keys" in order to enable copy and paste from the terminal window as shown in the figure.



Then click on "ASCII setup". Check "Echo typed characters locally" as shown in the next figure.



The terminal should now work as shown in the next figure.

