**EQASCOM: Supported ASCOM Properties and Methods**.

| ASCOM Properties | Required/ Optional | EQASCOM | Read | Write |
|---|---|---|---|---|
| AlignmentMode | O | ✓ | ✓ | **N/A** |
| Altitude | O | ✓ | ✓ | **N/A** |
| ApertureArea | O | **X** | | |
| AperetureDiameter | O | **X** | | |
| AtHome | R | ✓ | ✓ | **N/A** |
| AtPark | R | ✓ | ✓ | **N/A** |
| Azimuth | O | ✓ | ✓ | **N/A** |
| CanFindHome | R | ✓ | ✓ | **N/A** |
| CanPark | R | ✓ | ✓ | **N/A** |
| CanPulseGuide | R | ✓ | ✓ | **N/A** |
| CanSetDeclinationRate | R | ✓ | ✓ | **N/A** |
| CanSetGuideRates | R | ✓ | ✓ | **N/A** |
| CanSetPark | R | ✓ | ✓ | **N/A** |
| CanSetPierSide | R | ✓ | ✓ | **N/A** |
| CanSetRightAsscensionRate | R | ✓ | ✓ | **N/A** |
| CanSetTracking | R | ✓ | ✓ | **N/A** |
| CanSlew | R | ✓ | ✓ | **N/A** |
| CanSlewAltAz | R | ✓ | ✓ | **N/A** |
| CaqnSlewAltAxAsync | R | ✓ | ✓ | **N/A** |
| N/A CanSlewAsync | R | ✓ | ✓ | **N/A** |
| CanSync | R | ✓ | ✓ | **N/A** |
| CanSyncAltAz | R | ✓ | ✓ | **N/A** |
| CanUnpark | R | ✓ | ✓ | **N/A** |
| Connected | R | ✓ | ✓ | ✓ |
| Declination | R | ✓ | ✓ | **N/A** |
| DeclinationRate | O | ✓ | ✓ | ✓ |
| Description | R | ✓ | ✓ | **N/A** |
| DoesRefraction | O | **X** | | |
| DriverInfo | R | ✓ | ✓ | **N/A** |
| DriverVersion | R | ✓ | ✓ | **N/A** |
| EquatorialSystem | R | | ✓ | **N/A** |
| FocalLength | O | **X** | | |
| GuideRateDeclination | O | ✓ | ✓ | ✓ |
| GuideRateRightAscension | O | ✓ | ✓ | ✓ |
| InterfaceVersion | R | ✓ | ✓ | **N/A** |
| IsPulseGuiding | O | ✓ | ✓ | **N/A** |
| Name | R | ✓ | ✓ | **N/A** |
| RightAscension | R | ✓ | ✓ | **N/A** |
| RightAscensionRate | O | ✓ | ✓ | ✓ |
| SideOfPier | O | ✓ | ✓ | **X** |
| SiderealTime | R | ✓ | ✓ | **N/A** |
| SiteElevation | O | ✓ | ✓ | **X** |
| SiteLongitude | O | ✓ | ✓ | **X** |
| Slewing | O | ✓ | ✓ | **N/A** |

| | | | | |
|---|---|---|---|---|
| SlewSettleTime | O | ✓ | ✓ | ✓ |
| TargetDeclination | O | ✓ | ✓ | ✓ |
| TargetRightAscension | O | ✓ | ✓ | ✓ |
| Tracking | R | ✓ | ✓ | ✓ |
| TrackingRate | O | ✓ | ✓ | **X** |
| TrackingRates | O | ✓ | ✓ | **X** |
| UTCDate | R | ✓ | ✓ | **X** |

| **ASCOM Methods** | **Required /Optional** | **EQASCOM** | | |
|---|---|---|---|---|
| AbortSlew | O | ✓ | | |
| AxisRates | R | ✓ | | |
| CanMoveAxis | R | ✓ | | |
| CommandBlind | O | **X** | | |
| CommandBool | O | **X** | | |
| CommandString | O | ✓ | | |
| DestinationSiderOfPier | O | ✓ | | |
| FindHome | O | ✓ | | |
| MoveAxis | O | ✓ | | |
| Park* | O | ✓ | | |
| PulseGuide | O | ✓ | | |
| SetPark | O | ✓ | | |
| SetupDialog | R | ✓ | | |
| SlewToAltAz | O | **X** | | |
| SlewToAltAzAsync | O | **X** | | |
| SlewToCoordinates | O | ✓ | | |
| SlewToCoordinatesAsync | O | ✓ | | |
| SlewToTarget | O | ✓ | | |
| SlewToTargetAsync | O | ✓ | | |
| SyncToAltAz | O | **X** | | |
| SyncToCoordinates | O | ✓ | | |
| SyncToTarget | O | ✓ | | |
| Unpark | O | ✓ | | |

* Please note that by default EQASOM implements the park method as an ASYNCHRONOUS function. Clients can simply issue a park command and then poll the AtPark property to determine when a park operation has completed. Although this works well in most case it would seem that ASCOM intended for Park to be an SYNCHRONOUS method (although the ASCOM specification is woefully unclear on this). Synchronous methods only return control to the clients once the method has completed and as parking can take quite some time this actually seems a poor design decision when applied to mount control software. However, we are aware of some client applications that rely synchronous parking and so we provide an option in the EQASCOM setup to allow this type of park to be performed.

# EQASCOM CommandString Interface

Whilst ASCOM exposes many driver and mount specific properties and functions it is by no means a comprehensive standard (or indeed a particularly consistent one) and there is a general reluctance by those guiding ASCOM to expand or update the standard.

The CommandString interface provides a method by which a driver can expose additional proprietary properties and functions. Developers of ASCOM client applications (or indeed those using the standard windows Component Object Model) are welcome to access these additional properties and functions but must be aware that these are specific only to EQASCOM and are unlikely to be found on other ASCOM drivers.

| Function | Command String | EQASCOM Responds |
|---|---|---|
| Disable PEC | :PECENA,0# | |
| Enable PEC | :PECENA,1# | |
| Get PEC State | :PECSTA# | 0# (PEC Disabled)<br>1# (PEC Enabled) |
| Get Worm Tooth Count | :PECWTC# | worm_tooth_count# |
| Get Worm position | :PECIDX# | worm_position# |
| Get PEC Info | :PECINFO# | row_count,max_position # |
| Set Table Row | :PECSET,row_index,worm_position,pe # | 1# (success)<br>0# (failure) |
| Get Table Row | :PECSET,row_index# | 1,worm_position,pe#<br>0# (failure) |
| Load PEC Table | :PECLOAD,full_file_name# | 1# (success)<br>0# (failure) |
| Save PEC Table | :PECSAVE,full_file_name# | 1# (success)<br>0# (failure) |
| Get PEC Gain | :PECGAIN | gain# |
| Set PEC Gain | :PECGAIN,gain# | 1# (success)<br>0# (failure) |
| Get PEC Phase | :PECPHASE | phase# |
| Set PEC Phase | :PECPHASE,phase# | 1# (success)<br>0# (failure) |
| Get Mount Version | :MOUNTVER# | MountVersionString |
| Get EQASCOM Version | :DRIVERVER# | EQASCOM Version |
| Get eqcontrl.dll version | :DLLVER# | Dll version |
| Park | :PARK,parkmode# | 1# (mount parked or parking)<br>0# mount unparked |

| | | |
|---|---|---|
| Unpark | :UNPARK,unparkmode# | 1# (mount unparked or unparking)<br>0# (mount parked) |
| Get RA encoder | :RA_ENC# | Encoder position |
| Get DEC encoder | :DEC_ENC# | Encoder position |
| Get ST4 RA Guide Rate | :ST4_RARATE# | ST4GuideRate# |
| Set ST4 RA Guide Rate | :ST4_RARATE,ST4GuideRate# | 1# (success)<br>0# (failure) |
| Get ST4 DEC Guide Rate | :ST4_DECRATE# | ST4GuideRate# |
| Set ST4 DEC Guide Rate | :ST4_DECRATE,ST4GuideRate# | 1# (success)<br>0# (failure) |
| Get PulseGuide RA Guide Rate | :PG_RARATE# | PGGuideRate# |
| Set PulseGuide RA Guide Rate | :PG_RARATE,PGGuideRate# | 1# (success)<br>0# (failure) |
| Get PulseGuide DEC Guide Rate | :PG_DECRATE# | PGGuideRate# |
| Set PulseGuide DEC Guide Rate | :PG_DECRATE,PGGuideRate# | 1# (success)<br>0# (failure) |
| Get Alignment mode | :ALIGN_MODE# | 1# (append)<br>0# (dialog) |
| Set Alignment mode | :ALIGN_MODE,0#<br>:ALIGN_MODE,1# | 1# (success) |
| Clear sync | :ALIGN_CLEAR_SYNC# | 1# (success)<br>0# (failure) |
| Clear points | :ALIGN_CLEAR_POINTS# | 1# (success)<br>0# (failure) |
| Get sync limit status | :ALIGN_SYNC_LIMIT# | 1# (active)<br>0# (inactive) |
| Set sync limit status | :ALIGN_SYNC_LIMIT,0#<br>:ALIGN_SYNC_LIMIT,1# | 1# (success)<br>0# (failure) |
| Get Flipped Goto status | :FLIP_GOTO# | 1# (active)<br>0# (inactive) |
| Set Flipped Goto status | :FLIP_GOTO,0#<br>:FLIP_GOTO,1# | 1# (success)<br>0# (failure) |
| Set SNAP Port 1 status | :SNAP1,0#<br>:SNAP1,1# | Returns 0# or 1# mirroring the requested state. |

| | | |
|---|---|---|
| Set SNAP Port 2 status | :SNAP2,0#<br>:SNAP2,1# | Returns 0# or 1# mirroring the requested state. |
| Send Low Level Comms | >XX..XXX<br><br>XX..XX are the low level comms characters you wish to send i.e >:j1 will send:j1 to the mount (with LFCR appended) | YYYYY<br><br>Response from the mount motor controller |

All commands from the client begin with ":"
All commands end with "#"

Parameters are comma separated and passed as ASCII encoded integers. Avoid using any locale grouping settings (i.e. send 50132 rather than 50,132 ).

| | |
|---|---|
| worm_tooth_count | Number of teeth on RA axis gear. |
| row_count | Number or rows in the PEC table. Equates to the wormperiod. |
| max_position | Maximum worm position = microsteps per worm revolution –1 |
| worm_position | Current worm position ( 0 to max_position) |
| pe | Periodic error * 1000 |
| row_index | PEC table index ( 0 to row_count –1) |
| full_file_name | Full file path. |
| MountVersionString | XX.YY.ZZ<br><br>XX = major version (hex encoded)<br>YY = Minor Version (hex encoded)<br>ZZ = subversion (hex encoded) |
| parkmode | Numeric value as follows:<br>0 = park using current EQASCOM park operation<br>1 = Park to Home position<br>2 = Park to current Position<br>3 = Park to user position 1<br>4 = Park to user position 2<br>5 = Park to user position 3<br>6 = Park to user position 4<br>7 = Park to user position 5<br><br>note that not all user positions may be defined – check the response (1# or 0#) to determine success. |
| unparkmode | Numeric value as follows:<br>0 = unpark using current EQASCOM unpark operation<br>1 = unpark to current<br>2 = unpark to user position 1<br>3 = unpark to user position 2<br>4 = unpark to user position 3<br>5 = unpark to user position 4<br>6 = unpark to user position 5 |

| | |
|---|---|
| | note that EQASCOM provides up to 5 user defined park positions not all user positions may be defined – check the response (1# or 0#) to determine success. |
| ST4GuideRate | Either<br>0.25<br>0.5<br>0.75<br>1.00 |
| PGGuideRate | Either<br>0.1<br>0.2<br>0.3<br>0.4<br>0.5<br>0.6<br>0.7<br>0.8<br>0.9 |
| | |

Writes to the EQASCOM PEC table using the PECSET command are buffered into a temporary table. When the last row is written the table is automatically saved to disk as %APPDATA%\EQMOD\pec.txt and then loaded as the current PEC definition. EQASCOM will then use this file location for future automatic loads until such time as the user manually specifies an alternate filepath.

Reads of the EQASCOM PEC table using the PECGET command read the specified row of the currently loaded PEC file.

Steps per worm revolution, Worm period and Worm tooth count are all determined from parameters read from the mount itself. If the mount (or simulator) is not currently connected then reading these parameters will result in a value of –1.

# ASCOM Compliancy

These are the words ASCOM have to say on driver compliance.

> In order to be called "ASCOM compliant", a driver, component, or application scripting interface must meet *all* of the *applicable* guidelines in this document. Only then may a driver, interface, or a component's packaging and user interface, carry the ASCOM logo.
>
> 1. The driver *must* install and run on on Microsoft Windows 10, 8.1, 7, Vista, and XP with the latest service packs at the time of driver release, and with User Account Control enabled at its default/normal setting. It *should* work on both 32- and 64-bit systems. Support for Windows 2000 is no longer provided.
>
> 2. The driver author must implement the published standard for the device type as a late-bound COM interface. The published standards are specific about the data types that are used for properties and method parameters. These data types (and COM itself) are what make drivers cross language compatible. Note that by using a .NET language and the .NET driver templates we provide with the Platform Developer Components, this is all taken care-of for you. Also see item 6 below.
>
> 3.The driver *must never* "extend" the standard interface (add private members - properties and/or methods). If private members are desired, they *must* be exposed through a separate non-standard interface. Starting with Platform 6, driver authors can also extend their drivers through the new Action and SupportedActions members that are now common to all device interface standards.
>
> 4. The driver *must never* display a modal window which requires user interaction to dismiss. All errors must be raised/thrown back to the client.
>
> 5. The driver *must* use the Profile's Register() method for ASCOM registration. It is recommended that drivers also use the Profile object for storage of their persistent configuration, state data, etc., as well as the Serial object serial port I/O. for The components are part of the ASCOM Platform and serve to isolate drivers from changes in Platform architecture. They also make develpment easier by providing high level functionality commonly needed by drivers.
>
> 6. Prior to release, the driver *must* pass the Conform tests using the current/latest version of the Conformance Checker test tool.
>
> 7. The driver *must* be delivered as a self-contained installer. It is unacceptable to ask users to copy files, edit the registry, run BAT files, etc.

The items coloured in green EQMOD has no issues with. The Items in yellow are subject to interpretation or are poorly specified.

**Notes:**

Item 3: It may be that this requirement is just poorly phrased. For EQMOD the standard interface that ASCOM sits upon is COM and EQMOD does provide a few additional COM interfaces for applications to use if they wish. This is entirely normal part of windows application design and is not something ASCOM has any right to attempt to restrict.

Item 4: EQMOD sometimes puts up some modal dialogs that require user interaction to dismiss but then that is fine because EQMOD is not just a driver but it is a mount control application in its own right. –

Item 5. EQMOD prefers to retain control its own parameter management and serial comms.

Item 6: The interpretation of standards and compulsory guidelines is under continual review as is the conformance testing tool. This means that a driver that met all the necessary criteria when it was released (and is therefore compliant) may conceivably fail later releases of conformance tool. The conformance checker itself has had issues in the past and has many internal options that can freely be enabled or

disabled which seems rather odd given its role in granting compliance. The EQMOD ASCOM driver does try to implement the ASCOM interface the way ASCOM intend but there are situations where ASCOM itself becomes a sticking point in getting drivers and applications to work. EQMOD therefore also provides several optional “workarounds” that put it in a non-compliant mode. EQMOD is not alone in this – indeed ASCOM's own simulator products provide options that will cause conform failures such as “disconnect on park”

The EQMOD ASCOM Driver V1.29a introduces a new set-up option whereby it can be placed in a “strict ASCOM compliance” mode. When in this mode:

1. Exceptions are always raised and cannot be disabled
2. Side of Pier algorithm is always set to “Pointing” and there is no option to change this
3. MoveAxis methods are not supported.

In strict compliance mode the driver will pass Conform V6.2.58.0 with no errors or Issues raised.

If strict compliance isn't applied but side of pier algorithm is set to pointing and exceptions are set to be raised then the driver will fail he conform test with two issues relating to the way moveaxis is implemented – all other aspects of the driver pass their respective tests. The problem with moveaxis has never to our knowledge caused a real life practical issue. The MoveAxis command is essential should you wish to do satellite tracking via EQMODLX.

Unless your setup requires the moveaxis method, or requires exception's to be suppressed, the advice would be to place the EQMOD ASCOM driver into strict compliance mode.

So is EQMOD ASCOM Compliant? – yes pretty much but on any given day you can probably argue the toss either way.  Does it play well with other ASCOM applications – absolutely.