iOptron iEQ45 RS485 Command Language

November 10, 2010

This command set can be used to control an iEQ45 German equatorial mount via RS485 communication protocol. The commands will be sent to the mount via HBX port on the iEQ45 mount. A USB to HBX converting cable is needed. The interface is RS485, half duplex. Baud rate 28800, 8 bit, 1 start bit and 1 stop bit.

There is no assurance of accuracy or correctness of the document or verification of compliance by all versions of products. Only currently required commands are presented. The protocols are subject to and are expected to change and be revised due to continuing product improvements. Limited technical support will be provided.

This command set is based on C language. A pritnf format is adopted.

Mount Control Commands：

:T.GDriverInfo#   ***get mount parameters***.

sprintf(ReturnString,":DriverInfo %d,%d,%d,%d,%d,%d,%d,%d,%d,%d #"

RaWormTeeth,	R.A. worm wheel teeth number
RaEncoderLines,	R.A. per turn encoder counts
RaDecelerateRatio1,	R.A. reducing ratio #1
RaDecelerateRatio2,	R.A. reducing ratio #2
RaBackLash,	R.A. back lash in steps
DecWormTeeth,	DEC worm wheel teeth number
DecEncoderLines,	DEC per turn encoder counts
DecDecelerateRatio1,	DEC reducing ratio #1
DecDecelerateRatio2,	DEC reducing ratio #2
DecBackLash);	DEC back lash in steps

The reducing ratio for R.A. or DEC will be ratio #1:ratio #2.

:T.GMountType# ***get mount type***

Sprintf(ReturnString,“:MountType %d # ", MountType);

For iEQ45，MountType returns 0，indicates it is a German EQ Mount.

:T.GPierSide#   ***get Pier Side direction***

Sprintf(ReturnString, ":PierSide %d #",PierSide);

PierSide ==   PIER_WEST == 0
PierSide ==   PIER_EAST == 1

:T.GParkInfo#   ***get park position information***

sprintf(ReturnString,":ParkInfo %d,%d,%d #",ParkPosition,ParkRaSteps,ParkDecSteps);

ParkPosition

// 0 = original

```
        // 1 = Left and vertical
        // 2 = Left and horizon
        // 3 = Right and vertical
        // 4 = Right and horizon
        // 5 = User Position    defined by ParkRaSteps and ParkDecSteps
```

:T.GLocation#   ***get position parameters***

```
sprintf(ReturnString,":Location %10.6f,%11.6f,%6.1f,%5.1f,%6.1f #",
    Location.latitude,        Latitude information, North is +, South is -
    Location.longitude,       Longitude information, East is +, West is -
    Location.height,          Elevation
    Location.temperature,     Temperature
    Location.pressure);       Pressure
```

:T.GLocaltTime#   ***get local time***

```
sprintf(ReturnString,":LocaltTime %d,%d,%d,%d,%d,%d,%d,%d,%d #",
                LocalTime.Year,           Year
                LocalTime.Month,          Month
                LocalTime.Day,            Day
                LocalTime.Hour,           Hour
                LocalTime.Min,            Minute
                LocalTime.Sec,            Second
                DayLightSaving,           Daylight Saving Time 1=Y, 0=N
                TimeZone,                 Time Zone
                TimeZoneMin);             Time zone offset to GMT (in minutes)
```

:T.GMotorInfo#   ***get motor information***

```
sprintf(ReturnString,":MotorInfo %d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d #",
                RaCommErr,                R.A. communication error
                DecCommErr,               DEC communication error
                RaSteps,                  R.A. motor steps
                DecSteps,                 DEC motor steps
                RaSpeedNow,               R.A. current speed
                DecSpeedNow,              DEC current speed
                RaMotorStatus,            R.A. motor status
                DecMotorStatus,           DEC motor status
                RaDirection,              R.A. motor direction
                DecDirection,             DEC motor direction
                IfRaPECIndex,             If R.A. motor find PEC index point
                PEC_Start                 PEC index point
```

:T.GLocalGST#   ***get local sidereal time***

```
sprintf(ReturnString,":LocalGST %10.6f #",LocalGst);
```

:T.GAstroTele# ***get telescope parameter***

```
sprintf(ReturnString,":AstroTel %10.6f,%10.6f,%10.6f,%10.6f #",
        RaTele,         current R.A. of mount
        DecTele,        current DEC of mount
        AltTele,        current altitude of mount
        AziTele);       current azimuth of mount
```

:T.GAstroTarget# ***get target parameters***

```
    sprintf(ReturnString,":AstroTarget %10.6f,%10.6f,%10.6f,%10.6f #",
        RaTarget,       current R.A. of the target
        DecTarget,      current DEC of the target
        AltTarget,      current altitude of the target
        AziTarget);     current azimuth of the target
```

:T.GMoveInfo# ***get mount moving status***

```
    sprintf(ReturnString,":MoveInfo %d,%d,%d,%d,%d,%d,%d,%d #",
        IfInPECRecording,       if the mount is recording PEC
        IfInPECPlayBack,        if the mount is paling back the PEC
        IfInSlewing,            if the mount is slewing
        IfInTracking,           if the mount us tracking
        TrackingRates,          tracking rate SOLAR      0
                                          LUNAR          1
                                          SIDEREAL       2
                                          KING (user defined)     3
        IfSwapNS,               if the North and South button swapped
        IfSwapEW,               if the East and West button swapped
        IfZenithReverse);       If the mount will automatic reverse the side when passing through
                                the Zenith
```

:T.GGPSInfo# ***get GPS information***

```
    sprintf(ReturnString,":GGPSInfo %d,%d #",
        IfGPSInternal,          if an internal GPS equipped，iEQ45 =Y
        IfGPSFixed);            if GPS received data
```

:T.GGuiderInfo# ***get Autoguiding Port information***

```
  sprintf(ReturnString,":GuiderInfo %d,%d,%d,%d,%5.2f,%5.2f #",
    IfHaveOwnGuider,            if an Autoguiding Port equipped
    IfGuiderOnLine,             if a guider connected
    IfGuiderCanReverseDir,      if pin arrangement of a guider can be reversed
    GuiderType,                 the type of the guider (pin arrangement)
    RaAutoGuideSpeed,           R.A. autoguiding speed
    DecAutoGuideSpeed);          DEC autoguiding speed
```

:T.GNSInfo# ***get North/South hemisphere information***

```
sprintf(ReturnString,":Hemisphere %d #", EarthHemisphere);
    EarthHemisphere ＝ 1        NORTH_HEMISPHERE
    EarthHemisphere ＝ 0        SOUTH_HEMISPHERE
```

:T.GReticle# ***get dark field reticle illuminator value***

```
sprintf(ReturnString,":Reticle %d #", ReticleValue);
```

The bigger the ReticleValue, the brighter the reticle LED.

:T.SDriverInfo# ***set driver parameters.*** For an iEQ45, these numbers are preloaded and set as default. Any changing of the parameters will cause the mount perform incorrectly. These parameters will be recovered after a reset.

```
sscanf((char*)Recive,":T.SDriverInfo %hd,%hd,%hd,%hd,%hd,%hd,%hd,%hd,%hd,%hd #",
                &RaGearTeeth,
                &RaEncoderLines,
                &RaDecelerateRatio1,
                &RaDecelerateRatio2,
                &RaBackLash,
                &DecGearTeeth,
                &DecEncoderLines,
                &DecDecelerateRatio1,
                &DecDecelerateRatio2,
                &DecBackLash);
```

:T.SMountType# ***set mount type.*** For an iEQ45, this parameter is preset and set as default. Changing of the parameter will cause the mount perform incorrectly. The parameter will be recovered after a reset.

```
sscanf((char*)Recive,":T.SMountType %hd #",&MountType);
```

:T.SParkInfo# ***set park parameters***

```
sscanf((char*)Recive,":T.SParkInfo %hd,%d,%d #",
                &ParkPosition,&ParkRaSteps,&ParkDecSteps);
```

:T.SLocation# ***set current position parameters.*** It will be overridden after GPS receives data from satellites.

```
sscanf((char*)Recive,":T.SLocation %la,%la,%la,%la,%la #",
                &Location.latitude,        // ±90.xxxxxx
                &Location.longitude,       // ±180.xxxxxx
                &Location.height,
                &Location.temperature,
                &Location.pressure);
```

:T.SLocaltTime# ***set local time.*** It will be overridden after GPS receives time from satellites.

```
sscanf((char*)Recive,":T.SLocaltTime %hd,%hd,%hd,%hd,%hd,%hd,%hd,%hd,%hd #",
                        &LocalTime.Year,
                        &LocalTime.Month,
                        &LocalTime.Day,
                        &LocalTime.Hour,
                        &LocalTime.Min,
                        &LocalTime.Sec,
                        &DayLightSaving,
                        &TimeZone,
                        &TimeZoneMin);
```

:T.SAstroTarget# ***set target parameters***

```
      sscanf((char*)Recive,":T.SAstroTarget %la,%la,%la,%la #",
      &RaTarget,&DecTarget,&AltTarget,&AziTarget);
```

Set R.A., DEC, altitude and azimuth of a target.

:T.SResetRaDEC# ***reset R.A. and DEC motors to zero (0) position.***

R.A. and DEC is set to zero position

:T.SMoveInfo# ***set mount moving parameters***

```
      sscanf((char*)Recive,":T.SMoveInfo %hd,%hd,%hd,%hd #",
      &TrackingRates,&IfSwapNS,&IfSwapEW,&IfZenithReverse);
```

:T.SGuiderInfo# ***set Autoguing parameters***

```
      sscanf((char*)Recive,":T.SGuiderInfo %a,%a,%hd #",
      RaAutoGuideSpeed,&DecAutoGuideSpeed,&GuiderType);
```

:T.SHemisphere 1 # north
:T.SHemisphere 0 # south
set Noth/South hemisphere

:T.DUpGradeFirmWare# ***set iEQ45 main controller to firmware upgrade mode***.

:T.DStoreToFlash# ***store variables into iEQ45 flash memory (for power drop protection)***

:T.DGotoTargetEQ xxx# ***goto R.A. and DEC of a target (with MAX speed)***

```
sscanf((char*)Recive,":T.DGotoTargetEQ %hd #", &GotoMaxSpeed);
```

:T.DGotoTargetAA# ***goto altitude and azimuth of a target (not supported by iEQ45)***

:T.DStopGoto# ***stop Goto***

:T.DSyncToTarget# ***synchronize to target's current R.A. and DEC***

:T.DParkTele#   ***park the telescope to park position***

:T.DTeleGotoZero#   ***telescope go to zero position***

:T.DRaMove %g #   ***move the mount R.A. axle at a speed of %g***

:T.DDecMove %g# ***move the mount DEC axle at a speed of %g***

":T.DStartTracking TrackingRates,KingTrackingValue #   ***start tracking a target at a user defined speed***

:T.DStopTracking#   ***stop tracking***

:T.DStartPECRec 1#   ***start PEC recording.***   For iEQ45, only one PEC period is supported

:T.DCancelPECRec#   ***cancel PEC recording.***

:T.DStorePCERec#   ***store recorded PEC data to flash.***

:T.DStartPECPlay#   ***start PEC playback***

:T.DStopPECPlay#   ***stop PEC playback***

:T.DReticle xxx #   adjust dark field reticle intensity to xxx，0 ≤ xxx ≤ 255