Pyxis Command Reference

This document may be inaccurate and is given with no warranty. These commands and responses may change without warning. The use of this document to reverse engineer firmware or for the development of non-Optec hardware is forbidden.

This is intended for internal use only, if shared externally it is to be considered privileged information and may not be shared without written permission from Optec, Inc.

Contents

Introduction

The command syntax is universal regardless of which connection and communication method is being used to send the command and receive responses. A basic text string command structure is described herein.

Optec offer a freely available serial terminal application that can be used to connect with and control the Pyxis controller hub using the command syntax described below. The command processor included with the Pyxis controller is designed to use a Start-Command character and an End-Command character. This allows for commands of varying lengths and the ability to start a new command or terminate a command at any time. The Start-Command and End-Command characters are ‘<’, and ‘>’ respectively.

Following the Start-Command character the next character designates whether the command is intended for the rotator ‘R’ or the hub ‘H’ (controller) followed by a Device ID placeholder of ‘1’. The next two characters define a Transaction ID used as a response check for multi-threading processes. The following six characters define the Command ID as defined within the pages below. Any remaining characters before the end character ‘>’ will provide Boolean or Payload values of varying length.

Syntax Conventions

The individual commands are listed below and should follow the format below:

<DdiiCCCCCCyxxx…> where,

‘**D**’ is the **Target Device** – valid values are “R” for rotator or “H” for hub,
‘**d**’ is the **Device ID** – always “1” for the Pyxis rotator *(see note below)*,
‘**ii**’ is the **Transaction ID** – a 2-digit numeric character string from 00 to 99,
‘**CCCCCC**’ is the **Command ID**, always 6-characters, as detailed below,
‘**zzz…**’ is the **Payload**, a variable length character string used in certain commands requiring additional parameters. The length and contents of this Payload string are dependent upon the preceding Command ID. Additional details are given below.

In the following syntax reference, all upper-case letters are constant for the given command; all lower-case letters are variable parameters for the command. For example, valid **Target Device** identifiers ‘D’ are exclusively “R” or “H”.

The ‘d’ parameter is the **Device ID** placeholder and will always have a value of “1” for the Pyxis Rotator.

The ‘ii’ parameter is the **Transaction ID** and can be any numeric text value from 00 to 99. The Transaction ID created and defined by the client PC and is always echoed by the Pyxis controller preceded by the bang ‘!’ character. Using a Transaction ID allows multi-threaded client level applications to maintain proper registration of commands and responses.

The ‘CCCCCC’ **Command ID**’s are individually details in this document. Command ID’s fall into three classes, GET commands return configuration and device details. SET commands push configuration details to the device. DO action commands perform various device movements or other actions.

The ‘zzz…’ **Payload** parameter is a variable length string and will follow certain Command ID’s requiring additional details. For example, the Payload parameter may represents a target position in steps, or a network IP address, or a Device NickName.

Pyxis Responses

The Pyxis controller will respond to every command with a bang ‘!’ character, followed by the Transaction ID ‘ii’, and a newline (ASCII 0x10) character to indicate that a command was successfully received. Often a multiline response will finish with an “END” string followed by the newline character. If a syntax error was detected or an unrecognized command was received, the controller will respond with a string of text indicating an error code and error message in place of the expected response.

See Appendix B for additional Error Code details.

GET Configuration Commands

Command: Rotator - GET Device NickName

Syntax: `<RdiiGETDNN>`

Parameters: d =Device ID [1]; ii = Transaction ID.

Description: This command requests the rotator identification string or “Nickname”. Valid string lengths for “Nickname” are 1 to 16 ASCII characters. The controller will respond with the rotator’s user-defined nickname.

Note: GETDNN is generally used to ping the device to check communication status.

Example Command: `<R102GETDNN>`

Example Response:

```
!02
Nickname = Pollux
END
END
```

Command: Rotator - GET STAtus

Syntax: `<RdiiGETSTA>`

Parameters: d =Device ID [1]; ii = Transaction ID.

Description: This command will request the controller to report the status of the specified rotator. The first line of the status report is “!ii” and the last line is “END” to indicate that the output is complete. The status output contains the following fields:

Example Command: `<R104GETSTA>`

Example Response:

```
!04
Current Step = 0
Target Step = 0
Current PA = 180000
Target PA = 180000
Is Moving = 0
Is Homing = 0
Is Homed = 1
Is Sleeping = 0
END
```

Command: Rotator - GET ConFiGuration settings

Syntax: `<RxiiGETCFG>`

Parameters: x =Device ID [1]; ii = Transaction ID.

Description: This command will request the controller to report the configuration settings of the specified rotator. The first line of the configuration report is “!ii” and the last line is “END” to indicate that the output is complete. The configuration output contains the following fields:

Example Command: `<R106GETCFG>`

Example Response:

```
!06
Nickname = Rotator
Max Steps = 29332
Device Type = P2
Is Backlash Compensating = 0
Backlash Steps = 40
Home On Start = 1
Is Reversed = 0
Max Speed = 900
Park Position = 0
PA Offset = 0
END
```

Command: Pyxis Hub- GET ConFiGuration settings

Syntax: `<HxiiGETCFG>`

Parameters: x =Device ID [1]; ii = Transaction ID.

Description: The Hub Get Config command requests the hub to report the current Pyxis Hub configuration information. The first line of the configuration report is “!ii” and the last line is “END” to indicate that the output is complete. The Hub Configuration report contains the following fields.

Example Command: `<H107GETCFG>`

Example Response:

```
!07
Firmware Version = 3.0.0
Command Version = 0.0.1
Release Date = 2017/06/12
LED Brightness = 75
Hand Control = 0
Wired IP = 169.254.1.1
END
```

DO Action Commands

Command: Rotator- HALT

Syntax: `<RdiiDOHALT>`

Parameters: d =Device ID [1]; ii = Transaction ID.

Description: This command immediately halts the current rotator movement. The controller will respond with the “!ii” and “END” strings to acknowledge the command was received.

Note: If the rotator HALT is performed while the magnet is directly over the hall sensor, the IsHomed property will be set to false and a HOME procedure will be required.

Example Command: `<R111DOHALT>`

Example Response:

```
!11
END
```

Command: Rotator- HOME

Syntax: `<RdiiDOHOME>`

Parameters: d =Device ID [1]; ii = Transaction ID.

Description: This command requests the specified rotator to begin its homing routine. The controller will respond with “!ii” followed by “END” to indicate that is has started homing.

Note: The rotator IsMoving property will be set to True (1) and the IsHomed property will be set to False (0) until completion of the Home operation. RxiiDOHALT will immediate stop the homing operation.

Example Command: `<R113DOHOME>`

Example Response:

```
!13
END
```

Command: Rotator- Begin Relative Move

Syntax: `<RdiiDOMOVEy>`

Parameters: d =Device ID [1]; ii = Transaction ID, y = direction of movement.

Description: This command requests the specified rotator to begin moving in the y direction, y=’0’ moves CCW, y=’1’ moves CW direction. The move will initially travel at low speed, then shift into high speed after a few seconds.

Note: This command is designed for internal use with the momentary push buttons of the hand control. The target position is set to the extents of travel (0 for y=’0’, and Max Posi for y=’1’) and IsMoving is set to true (‘1’). Issue the DOSTOP command to stop the current movement. The device will automatically stop at either extent of travel.

Example Command: `<R116DOMOVE0>`

Example Response:

```
!16
END
```

Command: Rotator- Stop Relative Move

Syntax: `<RdiiDOSTOP>`

Parameters: d =Device ID [1]; ii = Transaction ID.

Description: This command requests the specified rotator to stop the current movement. IsMoving will be set to False (‘1’) and target position will be set to the current position.

Note: This command is designed for internal use with a release of the momentary push buttons in the hand control.

Example Command: `<R118DOSTOP>`

Example Response:

```
!18
END
```

Command: Rotator- Move Absolute Position Angle

Syntax: `<RdiiMOVEPAzzzzzz>`

Parameters: d =Device ID [1]; ii = Transaction ID; z =target position angle (x1000).

Description: This command requests the specified rotator to begin moving to the absolute position angle specified. The target position must be between 0 and the 359999. The TargetPA will be set to ‘z’ and IsMoving will be set to true (‘1’).

Note: All position angles are instrumental position angles and multiplied by 1000 with no decimal point. (i.e. instrumental PA=15 is represented by 15000).

Example Command: `<R120MOVEPA90000>` *(Request move to position angle 90000 or PA=90°)*

Example Response:

```
!20
END
```

Command: Rotator- Move Relative Angle

Syntax: `<RdiiMOVEREzzzzzz>`

Parameters: d =Device ID [1]; ii = Transaction ID; z =target position angle (x1000).

Description: This command requests the specified rotator to begin moving to the absolute position angle specified. The target position must be between -359999 and the 359999. The TargetPA will be set to ‘z’ and IsMoving will be set to true (‘1’).

Note: All position angles are instrumental position angles and multiplied by 1000 with no decimal point.

Note: Under most circumstances you should use the Move Absolute Position Angle Command and offset the current position in the request.

Example Command: `<R120MOVEPA-15000>` *(Request move of -15°)*

Example Response:

```
!20
END
```

Command: Rotator- Sleep (Reserved)

Command: Rotator- Wake (Reserved)

SET Configuration Commands

The following commands are used to set device configuration settings. Unlike the Get commands which tend to be all-inclusive in the properties they report, the Set commands are separated to apply to only a single device property. In the syntax definitions lower-case letters still indicate a parameter. See the Parameters section under each command for more information.

Command: Rotator- Set Device Nickname

Syntax: `<RdiiSETDNNzzz…>`

Parameters: d =Device ID [1]; ii = Transaction ID; z = Device NickName (max of 16 characters).

Description: This command is used to set a custom nickname for the specified rotator. The nickname is the string that is returned by the controller when the GETDNN command is executed and is also reported in the GETCFG response. The Pyxis controller will respond with “END” to indicate that the setting was successfully updated.

The following example is used to set the rotator nickname to “Pollux”

Example Command: `<R131SETDNNPollux>`

Example Response:

```
!31
END
```

Command: Rotator- Set Device Type (Reserved)

Syntax: `<FdiiSETDEVzz>`

Parameters: d =Device ID [1]; ii = Transaction ID; z = Device Type identifier.

Description: The Set Device Type command declares the type identification for the specified device. For the Pyxis Rotator the Device Type will always be “P2”.

Example Command: `<R132SETDEVB>`

Example Response:

```
!32
END
```

Command: Rotator- Set Home on Start

Syntax: `<RdiiSETHOSy>`

Parameters: d =Device ID [1]; ii = Transaction ID; y = Boolean value.

Description: The Set Home on Start command sets whether or not the device will home upon initial start-up or power on. A payload parameter of y = ‘0’ will turn off the home on start feature; y=’1’ sets the device to home on start.

Example Command: `<R134SETHOS0>`   (sets Home on Start to false)

Example Response:

```
!34
END
```

Command: Rotator- Set Backlash Compensation Enabled

Syntax: `<FdiiSETBCEy>`

Parameters: d =Device ID [1]; ii = Transaction ID; y = Boolean value.

Description: This command allows the client to enable or disable the backlash compensation feature built into the Pyxis rotator. The payload parameter of y='1' turns on this feature; y='0' turns backlash compensation off.

Example Command: `<R141SETBCE0>` (turns off Backlash Compensation)

Example Response:

```
!41
SET
```

Command: Rotator- Set Backlash Compensation Steps

Syntax: `<RdiiSETBCSzz>`

Parameters: d =Device ID [1]; ii = Transaction ID; z = number of steps (0 to 99).

Description: This command allows the user to specify the number of steps that the rotator will move beyond the target position before returning to the target position. The following example sets the number of backlash compensation steps for the Pyxis rotator to 99 steps.

Example Command: `<R142SETBCS99>`

Example Response:

```
!42
SET
```

Command: Rotator- Set Reverse Property

Syntax: `<RdiiSETREVy>`

Parameters: d =Device ID [1]; ii = Transaction ID; y = Boolean value.

Description: The Set Reverse Property command allows the client to set the rotator reverse property to true or false. A payload parameter y='1' sets Reverse to true; y='0' sets Reverse to false meaning the direction of movement will be in the default direction. The default direction for increasing Position Angle values is always clockwise. Setting the Reverse property to true will not affect the underlying step position of the rotator. However, the Pyxis hub responses for position angles will be reversed when this property is true. Instrumental PA=0 and PA=180 will be unaffected.

Note: The Pyxis rotator will always home in the anti-clockwise direction until it locates the home magnet, then move in a clockwise direction to instrumental PA = 0. From that position

Example Command: `<R143SETREV1>` (set Reverse property to True)

Example Response:

```
!43
SET
```

Command: Pyxis Hub- Set LED Brightness

Syntax: `<HdiiSETLEDzz>`

Parameters: d =Device ID [1]; ii = Transaction ID; z = brightness value (0 to 99).

Description: This command can be used to set the brightness of the power-indication LED on the Pyxis controller enclosure. Changing the values adjusts the total current through the LED thus increasing or decreasing its brightness. A value of '0' will turn the LED off.

Example Command: `<H197SETLED75>` (set LED brightness to 75%)

Example Response:

```
!97
SET
```

Command: Pyxis - Reset To Factory Defaults

Syntax: <RdiiRESETR>,

Parameters: d =Device ID [1]; ii = Transaction ID.

Description: This command will reset all configuration and status information to factory defaults.

Example Command: <H198RESETH>

Example Response:

!98

SET

Command: Pyxis Hub- Reset To Factory Defaults

Syntax: <HdiiRESETH>,

Parameters: d =Device ID [1]; ii = Transaction ID.

Description: This command will reset all configuration and status information to factory defaults.

Example Command: <H198RESETH>

Example Response:

!98

SET

Command: Pyxis Hub- Reboot

Syntax: <HdiiREBOOT>

Parameters: d =Device ID [1]; ii = Transaction ID.

Description: This command will perform a soft reboot of the Pyxis hub for remote operations.

Example Command: <H199REBOOT>

Example Response:

!99

SET

Appendix A – Device Types

Reserved

Appendix B – Error Codes

The Pyxis firmware performs various error checks on input commands. When an incorrectly formatted command is received the hub will respond with an error code. A brief explanation of the various error codes is provided below.

Error 0 – Invalid Command Received

Example Command: `<xian;f>`

Example Response:

```
ERROR ID = 0
ERROR TEXT = The received command is formatted incorrectly
END
```

Error 1 – Invalid

Example Command: `<>`

Example Response:

```
ERROR ID = 1
ERROR TEXT = The received command was too long
END
```

Error 2 – Invalid Parameter Received

Example Command: `<R192MOVABS216000>`

Example Response:

ERROR ID = 2
ERROR TEXT = The received command contained invalid parameters
END

Error 3 – Identifier Not Recognized

Example Command: `<>`

Example Response:

```
ERROR ID = 3
ERROR TEXT = The received identifier was not recognized
END
```

Error 4 – Invalid Target Device

Example Command: `<G123GETCFG>`

Example Response:

```
ERROR ID = 4
ERROR TEXT = The command received was for an invalid target device
END
```

Error 5 – Device is Homing

Example Command: `<F123DOMOVE1>`

Example Response:

```
ERROR ID = 5
ERROR TEXT = The command is invalid because the device is homing
END
```

Error 11 – Device is Not Homed

Example Command: `<F123DOMOVE1>`

Example Response:

```
ERROR ID = 11
ERROR TEXT = The command failed because the rotator is not homed
END
```