CONEX-PSD

Two-Axis Position & Power Sensing device

Newport

Command Interface Manual
V3.0.x
# Table of Contents

1.0 Introduction ............................................................................................................. 1
  1.1 Purpose .................................................................................................................... 1
  1.2 Overview .................................................................................................................. 1

2.0 Command Interface ................................................................................................. 2
  2.1 Constructor .............................................................................................................. 2
  2.2 Functions .................................................................................................................. 2
    2.2.1 General .............................................................................................................. 2
         ◆ CloseInstrument .............................................................................................. 2
         ◆ GetDevices ....................................................................................................... 2
         ◆ OpenInstrument ............................................................................................... 2
         ◆ WriteToInstrument .......................................................................................... 3
    2.2.2 Commands ....................................................................................................... 3
         ◆ GP ..................................................................................................................... 3
         ◆ ID_Get ................................................................................................................. 3
         ◆ ID_Set ................................................................................................................. 4
         ◆ IS_Get ................................................................................................................. 4
         ◆ IS_Set ................................................................................................................. 4
         ◆ IX_Get ............................................................................................................... 5
         ◆ IX_Set ............................................................................................................... 5
         ◆ IY_Get ............................................................................................................... 5
         ◆ IY_Set ............................................................................................................... 5
         ◆ LF_Get ............................................................................................................... 6
         ◆ LF_Set ............................................................................................................... 6
         ◆ OF_Get ............................................................................................................... 7
         ◆ OF_Set ............................................................................................................... 7
         ◆ PS_Get................................................................................................................. 8
         ◆ PS_Set ................................................................................................................. 8
         ◆ PX_Get................................................................................................................. 8
         ◆ PX_Set ................................................................................................................. 9
         ◆ PY_Get................................................................................................................. 9
         ◆ PY_Set ................................................................................................................. 9
         ◆ PW_Get.............................................................................................................. 10
         ◆ PW_Set .............................................................................................................. 10
         ◆ RA ..................................................................................................................... 10
         ◆ RC ..................................................................................................................... 11
         ◆ RS ..................................................................................................................... 11
         ◆ RS485 .............................................................................................................. 11
3.0 Python Example................................................................. 14

Service Form ......................................................................................... 17
1.0 Introduction

1.1 Purpose
The purpose of this document is to provide the method syntax of each command to communicate with the CONEX-PSD device.

1.2 Overview
The Command Interface is the wrapper class that maintains a list of CONEX-PSD instruments. It exposes methods to communicate with any CONEX-PSD device.

NOTE
Each function name is defined with the command code “AA”.
For each command function, refer to the CONEX-PSD programmer’s manual.
2.0 Command Interface

2.1 Constructor

ConexPSD()

The constructor is used to create an instance of the CONEX-PSD device.

2.2 Functions

2.2.1 General

♦ CloseInstrument

**Syntax**

```c
int CloseInstrument()
```

return: 0 = successful or -1 = failure

**Description**

This function allows closing communication with the selected device. If the closing failed, the returned code is -1.

♦ GetDevices

**Syntax**

```c
string[] GetDevices()
```

return: list of connected devices available to communicate

**Description**

This function returns the list of connected devices available to communicate.

♦ OpenInstrument

**Syntax**

```c
int OpenInstrument(string strDeviceKey)
```

string strDeviceKey: device key

return: 0 = successful or -1 = failure

**Description**

This function allows opening communication with the selected device. If the opening failed, the returned code is -1.
♦ **WriteToInstrument**

**Syntax**

```csharp
int WriteToInstrument(string command, ref string response, int stage)
```

**command:** Instrument command

**response:** Response of the command

**stage:** Instrument Stage

**return:**

**Description**

This overridden function queries or writes the command given by the user to the instrument.

### 2.2.2 Commands

♦ **GP**

**Syntax**

```csharp
int GP(int controllerAddress, out double PositionX, out double PositionY, out double LaserPower, out string errstring)
```

**controllerAddress:** controllerAddress

**PositionX:** PositionX

**PositionY:** PositionY

**LaserPower:** LaserPower

**errString:** The failure reason

**return:** 0 in success and -1 on failure

**Description**

This function is used to process synchronous GP Get command which is used to get X, Y positions and laser power level.

♦ **ID_Get**

**Syntax**

```csharp
int ID_Get(int controllerAddress, out string SensorIdentifier, out string errstring)
```

**controllerAddress:** controllerAddress

**SensorIdentifier:** SensorIdentifier

**errString:** The failure reason

**return:** 0 in success and -1 on failure

**Description**

This function is used to process synchronous ID Get command which is used to get sensor identifier.
♦ ID_Set

Syntax
int ID_Set(int controllerAddress, string SensorIdentifier, out string errstring)
controllerAddress: controllerAddress
SensorIdentifier: SensorIdentifier
errString: The failure reason
return: 0 in success and -1 on failure

Description
This function is used to process synchronous ID Set command which is used to Set sensor identifier.

♦ IS_Get

Syntax
int IS_Get(int controllerAddress, out double Offset, out string errstring)
controllerAddress: controllerAddress
Offset: Offset
errString: The failure reason
return: 0 in success and -1 on failure

Description
This function is used to process synchronous IS Get command which is used to Get offset on ADC input SUM.

♦ IS_Set

Syntax
int IS_Set(int controllerAddress, double Offset, out string errstring)
controllerAddress: controllerAddress
Offset: Offset
errString: The failure reason
return: 0 in success and -1 on failure

Description
This function is used to process synchronous IS Set command which is used to Set offset on ADC input SUM.
♦ IX_Get

**Syntax**

```c
int IX_Get(int controllerAddress, out double OffsetADC1, out string errstring)
```

callerAddress: controllerAddress

OffsetADC1: OffsetADC1

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronisous IX Get command which is used to get offset on ADC input X.

♦ IX_Set

**Syntax**

```c
int IX_Set(int controllerAddress, double OffsetADC1, out string errstring)
```

callerAddress: controllerAddress

OffsetADC1: OffsetADC1

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronisous IX Set command which is used to set offset on ADC input X.

♦ IY_Get

**Syntax**

```c
int IY_Get(int controllerAddress, out double OffsetADC2, out string errstring)
```

callerAddress: controllerAddress

OffsetADC2: OffsetADC2

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronisous IY Get command which is used to get offset on ADC input Y.
♦ **IY_Set**

**Syntax**

```c
int IY_Set(int controllerAddress, double OffsetADC2, out string errstring)
```

- `controllerAddress`: controllerAddress
- `OffsetADC2`: OffsetADC2
- `errString`: The failure reason
- `return`: 0 in success and -1 on failure

**Description**

This function is used to process synchronous IY Set command which is used to set offset on ADC input Y.

♦ **LF_Get**

**Syntax**

```c
int LF_Get(int controllerAddress, out double Frequency, out string errstring)
```

- `controllerAddress`: controllerAddress
- `Frequency`: Frequency
- `errString`: The failure reason
- `return`: 0 in success and -1 on failure

**Description**

This function is used to process synchronous LF Get command which is used to get low pass filter frequency.

♦ **LF_Set**

**Syntax**

```c
int LF_Set(int controllerAddress, double Frequency, out string errstring)
```

- `controllerAddress`: controllerAddress
- `Frequency`: Frequency
- `errString`: The failure reason
- `return`: 0 in success and -1 on failure

**Description**

This function is used to process synchronous LF Set command which is used to Set low pass filter frequency.
♦ **OF_Get**

**NOTE**
This command is not used with the CONEX-PSD9.

**Syntax**

```
int OF_Get(int controllerAddress, out double Offset1, out double Offset2, out double Offset3, out double Offset4, out string errstring)
```

controllerAddress: controllerAddress
Offset1: Offset #1
Offset2: Offset #2
Offset3: Offset #3
Offset4: Offset #4
erString: The failure reason
return: 0 in success and -1 on failure

**Description**
This function is used to process synchronous OF Get command which is used to get offsets. Refer to the CONEX-PSD Controller’s manual to get the command description.

♦ **OF_Set**

**NOTE**
This command is not used with the CONEX-PSD9.

**Syntax**

```
int OF_Set(int controllerAddress, double Offset1, double Offset2, double Offset3, double Offset4, out string errstring)
```

controllerAddress: Controller’s address
Offset1: Offset #1
Offset2: Offset #2
Offset3: Offset #3
Offset4: Offset #4
erString: The failure reason
return: 0 in success and -1 on failure

**Description**
This function is used to process synchronous OF Set command which is used to set offsets. Refer to the CONEX-PSD Controller’s manual to get the command description.
♦ **PS_Get**

**Syntax**

```c
int PS_Get(int controllerAddress, out double Gain, out string errstring)
controllerAddress: controllerAddress
Gain: Gain
errString: The failure reason
return: 0 in success and -1 on failure
```

**Description**

This function is used to process synchrounous PS Get command which is used to Get gain on ADC input SUM.

♦ **PS_Set**

**Syntax**

```c
int PS_Set(int controllerAddress, double Gain, out string errstring)
controllerAddress: controllerAddress
Gain: Gain
errString: The failure reason
return: 0 in success and -1 on failure
```

**Description**

This function is used to process synchrounous PS Set command which is used to Set gain on ADC input SUM.

♦ **PX_Get**

**Syntax**

```c
int PX_Get(int controllerAddress, out double GainADC1, out string errstring)
controllerAddress: controllerAddress
GainADC1: GainADC1
errString: The failure reason
return: 0 in success and -1 on failure
```

**Description**

This function is used to process synchrounous PX Get command which is used to Get gain on ADC input X.
♦ PX_Set

Syntax
int PX_Set(int controllerAddress, double GainADC1, out string errstring)
controllerAddress: controllerAddress
GainADC1: GainADC1
errString: The failure reason
return: 0 in success and -1 on failure

Description
This function is used to process synchronous PX Set command which is used to Set
gain on ADC input X.

♦ PY_Get

Syntax
int PY_Get(int controllerAddress, out double GainADC2, out string errstring)
controllerAddress: controllerAddress
GainADC2: GainADC2
errString: The failure reason
return: 0 in success and -1 on failure

Description
This function is used to process synchronous PY Get command which is used to Get
gain on ADC input Y.

♦ PY_Set

Syntax
int PY_Set(int controllerAddress, double GainADC2, out string errstring)
controllerAddress: controllerAddress
GainADC2: GainADC2
errString: The failure reason
return: 0 in success and -1 on failure

Description
This function is used to process synchronous PY Set command which is used to Set
gain on ADC input Y.
♦ PW_Get

**Syntax**

```c
int PW_Get(int controllerAddress, out int ConfigurationState, out string errString)
controllerAddress: controllerAddress
ConfigurationState: ConfigurationState
errString: The failure reason
return: 0 in success and -1 on failure
```

**Description**

This function is used to process synchronous PW Get command which is used to Enter/Leave CONFIGURATION state.

♦ PW_Set

**Syntax**

```c
int PW_Set(int controllerAddress, int ConfigurationState, out string errString)
controllerAddress: controllerAddress
ConfigurationState: ConfigurationState
errString: The failure reason
return: 0 in success and -1 on failure
```

**Description**

This function is used to process synchronous PW Set command which is used to Enter/Leave CONFIGURATION state.

♦ RA

**Syntax**

```c
int RA(int controllerAddress, out double RawAnalogInput1, out double RawAnalogInput2, out string errString)
controllerAddress: controllerAddress
RawAnalogInput1: RawAnalogInput1
RawAnalogInput2: RawAnalogInput2
errString: The failure reason
return: 0 in success and -1 on failure
```

**Description**

This function is used to process synchronous RA Get command which is used to get raw analog input values.
♦ **RC**

**Syntax**

```c
int RC(int controllerAddress, out double CorrectedAnalogInput1, out double CorrectedAnalogInput2, out string errstring)
```

- `controllerAddress`: controllerAddress
- `CorrectedAnalogInput1`: CorrectedAnalogInput1
- `CorrectedAnalogInput2`: CorrectedAnalogInput2
- `errString`: The failure reason

{return: 0 in success and -1 on failure}

**Description**

This function is used to process synchronous RC Get command which is used to Get corrected analog input values.

♦ **RS**

**Syntax**

```c
int RS(int controllerAddress, out string errstring)
```

- `controllerAddress`: controllerAddress
- `errString`: The failure reason

{return: 0 in success and -1 on failure}

**Description**

This function is used to process synchronous RS Set command which is used to Reset controller.

♦ **RS485**

**Syntax**

```c
int RS485(int controllerAddress, out string errstring)
```

- `controllerAddress`: controllerAddress
- `errString`: The failure reason

{return: 0 in success and -1 on failure}

**Description**

This function is used to process synchronous RS485 Set command which is used to Reset controller’s address to 1.
♦ **SA_Get**

**Syntax**

```c
int SA_Get(int controllerAddress, out int Adress, out string errstring)
controllerAddress: controllerAddress
Adress: Adress
errString: The failure reason
return: 0 in success and -1 on failure
```

**Description**

This function is used to process synchronous SA Get command which is used to get controller’s RS-485 address.

♦ **SA_Set**

**Syntax**

```c
int SA_Set(int controllerAddress, int Address, out string errstring)
controllerAddress: controllerAddress
Address : Address
errString: The failure reason
return: 0 in success and -1 on failure
```

**Description**

This function is used to process synchronous SA Set command which is used to Set controller’s RS-485 address.

♦ **TB_Get**

**Syntax**

```c
int TB_Get(int controllerAddress, string inErrorCode, string outErrorCode, out string errstring)
controllerAddress: controllerAddress
inErrorCode: Input error code (optional)
outErrorCode: Output error description
errString: The failure reason
return: 0 in success and -1 on failure
```

**Description**

This function is used to process synchronous TB Get command which is used to Get command error string.
♦ TE

**Syntax**

```c
int TE(int controllerAddress, out string LastCommandError, out string errstring)
controllerAddress: controllerAddress
LastCommandError: LastCommandError
errString: The failure reason
return: 0 in success and -1 on failure
```

**Description**

This function is used to process synchronous TE Get command which is used to Get last command error.

♦ TS

**Syntax**

```c
int TS(int controllerAddress, out string ErrorCode, out string StatusCode, out string errstring)
controllerAddress: controllerAddress
ErrorCode: ErrorCode
StatusCode: StatusCode
errString: The failure reason
return: 0 in success and -1 on failure
```

**Description**

This function is used to process synchronous TS Get command which is used to Get positioner error and controller state.

♦ VE

**Syntax**

```c
int VE(int controllerAddress, out string ControllerVersion, out string errstring)
controllerAddress: controllerAddress
ControllerVersion: ControllerVersion
errString: The failure reason
return: 0 in success and -1 on failure
```

**Description**

This function is used to process synchronous VE Get command which is used to Get controller revision information.
3.0 Python Example

```python
# Initialization Start
# The script within Initialization Start and Initialization End is needed for properly
# initializing Command Interface for Conex-PSD instrument.
# The user should copy this code as is and specify correct paths here.
import sys

# Command Interface DLL can be found here.
print "Adding location of Newport.CONEXPSD.CommandInterface.dll to sys.path"
sys.path.append(r'C:\Program Files (x86)\Newport\MotionControl\CONEX-PSD\Bin')

# The CLR module provide functions for interacting with the underlying
# .NET runtime
import clr

# Add reference to assembly and import names from namespace
clr.AddReferenceToFile("Newport.CONEXPSD.CommandInterface.dll")
from CommandInterface import *

import System

# Instrument Initialization
# The key should have double slashes since
# (one of them is escape character)
instrument="CONEX-PSD (A6T7NSPR)"
print 'Instrument Key=>', instrument

# create a device instance
PSD = ConexPSD()

componentID = PSD. OpenInstrument(instrument);
print 'componentID=>', componentID

# Get analog output #1 value
result, X, Y, LaserPower, errString = PSD.GP_Get(1)
if result == 0 :
    print 'X =>', X
    print 'Y =>', Y
    print 'Laser power =>', LaserPower
else:
    print 'Error=>', errString
```
# Get controller revision information
result, response, errString = PSD.VE(1)
if result == 0 :
    print 'controller revision=>', response
else:
    print 'Error=>',errString

# Get last command error
result, response, errString = PSD.TE(1)
if result == 0 :
    print 'Last command error =>', response
else:
    print 'Error=>',errString

# Unregister device
PSD. CloseInstrument();
Service Form

Name: _________________________________________________  Return authorization #: ____________________
Company: ____________________________________________
Address: ________________________________________________  (Please obtain prior to return of item)
Country: ________________________________________________  Date: ____________________
P.O. Number: ____________________________________________  Phone Number: ____________________
Item(s) Being Returned: ____________________________________  Fax Number: ____________________
Model#: ________________________________________________  Serial #: ____________________

Description: __________________________________________________________________________________________
Reasons of return of goods (please list any specific problems): ______________________________________________________
__________________________________________________________________________________________________________
__________________________________________________________________________________________________________
__________________________________________________________________________________________________________
__________________________________________________________________________________________________________
__________________________________________________________________________________________________________
__________________________________________________________________________________________________________
__________________________________________________________________________________________________________
__________________________________________________________________________________________________________
__________________________________________________________________________________________________________
__________________________________________________________________________________________________________
__________________________________________________________________________________________________________
__________________________________________________________________________________________________________
__________________________________________________________________________________________________________
__________________________________________________________________________________________________________

Your Local Representative
Tel.: ____________________
Fax: ____________________
North America & Asia
Newport Corporation
1791 Deere Ave.
Irvine, CA 92606, USA
Sales
Tel.: (800) 222-6440
e-mail: sales@newport.com
Technical Support
Tel.: (800) 222-6440
e-mail: tech@newport.com
Service, RMAs & Returns
Tel.: (800) 222-6440
e-mail: service@newport.com

Europe
MICRO-CONTROLE Spectra-Physics S.A.S
9, rue du Bois Sauvage
91055 Évry CEDEX
France
Sales
Tel.: +33 (0)1.60.91.68.68
e-mail: france@newport.com
Technical Support
e-mail: tech_europe@newport.com
Service & Returns
Tel.: +33 (0)2.38.40.51.55