CONEX-CC

Single-Axis DC Motion with Controller/Driver

Newport® Command Interface Manual
V2.0.x
©2018 by Newport Corporation, Irvine, CA. All rights reserved.

Original instructions.

No part of this document may be reproduced or copied without the prior written approval of Newport Corporation. This document is provided for information only, and product specifications are subject to change without notice. Any change will be reflected in future publishings.
# 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

   2.2.1.1 OpenInstrument ...................................................................................... 2

   2.2.1.2 CloseInstrument ..................................................................................... 2

   2.2.1.3 GetDevices ............................................................................................. 2

   2.2.1.4 WriteToInstrument ................................................................................. 3

2.2.2 Commands ..................................................................................................... 3

   2.2.2.1 AC_Get .................................................................................................. 3

   2.2.2.2 AC_Set ................................................................................................... 3

   2.2.2.3 BA_Get .................................................................................................. 4

   2.2.2.4 BA_Set ................................................................................................... 4

   2.2.2.5 BH_Get .................................................................................................. 4

   2.2.2.6 BH_Set ................................................................................................... 5

   2.2.2.7 DV_Get .................................................................................................. 5

   2.2.2.8 DV_Set ................................................................................................... 5

   2.2.2.9 FD_Get ................................................................................................... 6

   2.2.2.10 FD_Set ................................................................................................. 6

   2.2.2.11 FE_Get ................................................................................................ 6

   2.2.2.12 FE_Set .................................................................................................. 7

   2.2.2.13 FF_Get ................................................................................................. 7

   2.2.2.14 FF_Set ................................................................................................ 7

   2.2.2.15 HT_Get ................................................................................................ 8

   2.2.2.16 HT_Set ................................................................................................ 8

   2.2.2.17 ID_Get ................................................................................................ 8

   2.2.2.18 ID_Set ................................................................................................ 9

   2.2.2.19 JR_Get ................................................................................................ 9

   2.2.2.20 JR_Set ................................................................................................ 9

   2.2.2.21 KD_Get ................................................................................................. 10

   2.2.2.22 KD_Set ............................................................................................... 10

   2.2.2.23 KI_Get ................................................................................................ 10

   2.2.2.24 KI_Set ................................................................................................ 11

   2.2.2.25 KP_Get ............................................................................................... 11
<table>
<thead>
<tr>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2.2.26</td>
<td>KP_Set</td>
</tr>
<tr>
<td>2.2.2.27</td>
<td>KV_Get</td>
</tr>
<tr>
<td>2.2.2.28</td>
<td>KV_Set</td>
</tr>
<tr>
<td>2.2.2.29</td>
<td>MM_Get</td>
</tr>
<tr>
<td>2.2.2.30</td>
<td>MM_Set</td>
</tr>
<tr>
<td>2.2.2.31</td>
<td>OH_Get</td>
</tr>
<tr>
<td>2.2.2.32</td>
<td>OH_Set</td>
</tr>
<tr>
<td>2.2.2.33</td>
<td>OR</td>
</tr>
<tr>
<td>2.2.2.34</td>
<td>OT_Get</td>
</tr>
<tr>
<td>2.2.2.35</td>
<td>OT_Set</td>
</tr>
<tr>
<td>2.2.2.36</td>
<td>PA_Get</td>
</tr>
<tr>
<td>2.2.2.37</td>
<td>PA_Set</td>
</tr>
<tr>
<td>2.2.2.38</td>
<td>PR_Get</td>
</tr>
<tr>
<td>2.2.2.39</td>
<td>PR_Set</td>
</tr>
<tr>
<td>2.2.2.40</td>
<td>PT_Get</td>
</tr>
<tr>
<td>2.2.2.41</td>
<td>PT_Set</td>
</tr>
<tr>
<td>2.2.2.42</td>
<td>PW_Get</td>
</tr>
<tr>
<td>2.2.2.43</td>
<td>PW_Set</td>
</tr>
<tr>
<td>2.2.2.44</td>
<td>QIL_Get</td>
</tr>
<tr>
<td>2.2.2.45</td>
<td>QIL_Set</td>
</tr>
<tr>
<td>2.2.2.46</td>
<td>QIR_Get</td>
</tr>
<tr>
<td>2.2.2.47</td>
<td>QIR_Set</td>
</tr>
<tr>
<td>2.2.2.48</td>
<td>QIT_Get</td>
</tr>
<tr>
<td>2.2.2.49</td>
<td>QIT_Set</td>
</tr>
<tr>
<td>2.2.2.50</td>
<td>RS</td>
</tr>
<tr>
<td>2.2.2.51</td>
<td>RS485</td>
</tr>
<tr>
<td>2.2.2.52</td>
<td>SA_Get</td>
</tr>
<tr>
<td>2.2.2.53</td>
<td>SA_Set</td>
</tr>
<tr>
<td>2.2.2.54</td>
<td>SC_Get</td>
</tr>
<tr>
<td>2.2.2.55</td>
<td>SC_Set</td>
</tr>
<tr>
<td>2.2.2.56</td>
<td>SE</td>
</tr>
<tr>
<td>2.2.2.57</td>
<td>SL_Get</td>
</tr>
<tr>
<td>2.2.2.58</td>
<td>SL_Set</td>
</tr>
<tr>
<td>2.2.2.59</td>
<td>SR_Get</td>
</tr>
<tr>
<td>2.2.2.60</td>
<td>SR_Set</td>
</tr>
<tr>
<td>2.2.2.61</td>
<td>ST</td>
</tr>
<tr>
<td>2.2.2.62</td>
<td>SU_Get</td>
</tr>
<tr>
<td>2.2.2.63</td>
<td>SU_Set</td>
</tr>
<tr>
<td>2.2.2.64</td>
<td>TB</td>
</tr>
<tr>
<td>2.2.2.65</td>
<td>TE</td>
</tr>
<tr>
<td>2.2.2.66</td>
<td>TH</td>
</tr>
<tr>
<td>2.2.2.67</td>
<td>TK_Get</td>
</tr>
<tr>
<td>2.2.2.68</td>
<td>TK_Set</td>
</tr>
<tr>
<td>2.2.2.69</td>
<td>TP</td>
</tr>
</tbody>
</table>
2.2.2.70 TS.................................................................26
2.2.2.71 VA_Get.............................................................26
2.2.2.72 VA_Set............................................................27
2.2.2.73 VE.................................................................27
2.2.2.74 ZT.................................................................27

3.0 Python Example.................................................................28

Service Form .................................................................................31
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-CC device.

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

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

2.1 Constructor

ConexCC()

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

2.2 Functions

2.2.1 General

2.2.1.1 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.

2.2.1.2 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.

2.2.1.3 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.
2.2.1.4 **WriteToInstrument**

**Syntax**

```c
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**

2.2.2.1 **AC_Get**

**Syntax**

```c
int AC_Get(int controllerAddress, out double outAcceleration, out string errString)
```

- `controllerAddress`: Address of Controller
- `outAcceleration`: outAcceleration
- `errString`: The failure reason
- `return`: 0 in success and -1 on failure

**Description**

This function is used to process synchronous AC Get command which is used to Get acceleration.

2.2.2.2 **AC_Set**

**Syntax**

```c
int AC_Set(int controllerAddress, double inAcceleration, out string errString)
```

- `controllerAddress`: Address of Controller
- `inAcceleration`: inAcceleration
- `errString`: The failure reason
- `return`: 0 in success and -1 on failure

**Description**

This function is used to process synchronous AC Set command which is used to Set acceleration.
2.2.2.3 BA_Get

**Syntax**
int BA_Get(int controllerAddress, out double outBacklash, out string errString)
ccontrollerAddress: Address of Controller
outBacklash: outBacklash
errString: The failure reason
return: 0 in success and -1 on failure

**Description**
This function is used to process synchronous BA Get command which is used to Get backlash compensation.

2.2.2.4 BA_Set

**Syntax**
int BA_Set(int controllerAddress, double inBacklash, out string errString)
ccontrollerAddress: Address of Controller
inBacklash: inBacklash
errString: The failure reason
return: 0 in success and -1 on failure

**Description**
This function is used to process synchronous BA Set command which is used to Set backlash compensation.

2.2.2.5 BH_Get

**Syntax**
int BH_Get(int controllerAddress, out double outHysteresis, out string errString)
ccontrollerAddress: Address of Controller
outHysteresis: outHysteresis
errString: The failure reason
return: 0 in success and -1 on failure

**Description**
This function is used to process synchronous BH Get command which is used to Get hysteresis compensation.
### 2.2.2.6 BH_Set

**Syntax**

```c
int BH_Set(int controllerAddress, double inHysteresis, out string errString)
```

- `controllerAddress`: Address of Controller
- `inHysteresis`: Hysteresis
- `errString`: The failure reason

**Description**

This function is used to process synchronous BH Set command which is used to set hysteresis compensation.

### 2.2.2.7 DV_Get

**Syntax**

```c
int DV_Get(int controllerAddress, out double outDriverVoltage, out string errString)
```

- `controllerAddress`: Address of Controller
- `outDriverVoltage`: Driver Voltage
- `errString`: The failure reason

**Description**

This function is used to process synchronous DV Get command which is used to get driver voltage.

### 2.2.2.8 DV_Set

**Syntax**

```c
int DV_Set(int controllerAddress, double inDriverVoltage, out string errString)
```

- `controllerAddress`: Address of Controller
- `inDriverVoltage`: Driver Voltage
- `errString`: The failure reason

**Description**

This function is used to process synchronous DV Set command which is used to set driver voltage.
2.2.2.9  **FD_Get**

**Syntax**

```c
int FD_Get(int controllerAddress, out double outLowPassFilterKd, out string errString)
```

- `controllerAddress`: Address of Controller
- `outLowPassFilterKd`: `outLowPassFilterKd`
- `errString`: The failure reason

**Description**

This function is used to process synchronous FD Get command which is used to Get low pass filter for Kd.

2.2.2.10  **FD_Set**

**Syntax**

```c
int FD_Set(int controllerAddress, double inLowPassFilterKd, out string errString)
```

- `controllerAddress`: Address of Controller
- `inLowPassFilterKd`: `inLowPassFilterKd`
- `errString`: The failure reason

**Description**

This function is used to process synchronous FD Set command which is used to Set low pass filter for Kd.

2.2.2.11  **FE_Get**

**Syntax**

```c
int FE_Get(int controllerAddress, out double outFollowingError, out string errString)
```

- `controllerAddress`: Address of Controller
- `outFollowingError`: `outFollowingError`
- `errString`: The failure reason

**Description**

This function is used to process synchronous FE Get command which is used to Get following error limit.
2.2.2.12  **FE_Set**

**Syntax**

```
int FE_Set(int controllerAddress, double inFollowingError, out string errString)
```

- `controllerAddress`: Address of Controller
- `inFollowingError`: inFollowingError
- `errString`: The failure reason

**Description**

This function is used to process synchronous FE Set command which is used to Set following error limit.

2.2.2.13  **FF_Get**

**Syntax**

```
int FF_Get(int controllerAddress, out double outFrictionCompensation, out string errString)
```

- `controllerAddress`: Address of Controller
- `outFrictionCompensation`: outFrictionCompensation
- `errString`: The failure reason

**Description**

This function is used to process synchronous FF Get command which is used to Get friction compensation.

2.2.2.14  **FF_Set**

**Syntax**

```
int FF_Set(int controllerAddress, double inFrictionCompensation, out string errString)
```

- `controllerAddress`: Address of Controller
- `inFrictionCompensation`: inFrictionCompensation
- `errString`: The failure reason

**Description**

This function is used to process synchronous FF Set command which is used to Set friction compensation.
2.2.2.15 **HT_Get**

**Syntax**

```c
int HT_Get(int controllerAddress, out int outHomeType, out string errString)
controllerAddress: Address of Controller
outHomeType: outHomeType
errString: The failure reason
return: 0 in success and -1 on failure
```

**Description**

This function is used to process synchronous HT Get command which is used to Get HOME search type.

2.2.2.16 **HT_Set**

**Syntax**

```c
int HT_Set(int controllerAddress, int inHomeType, out string errString)
controllerAddress: Address of Controller
inHomeType: inHomeType.
errString: The failure reason
return: 0 in success and -1 on failure
```

**Description**

This function is used to process synchronous HT Set command which is used to Set HOME search type.

2.2.2.17 **ID_Get**

**Syntax**

```c
int ID_Get(int controllerAddress, out string outStageIdentifier, out string errString)
controllerAddress: Address of Controller
outStageIdentifier: outStageIdentifier
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 stage identifier.
2.2.2.18 **ID\_Set**

**Syntax**

```c
int ID_Set(int controllerAddress, string inStageIdentifier, out string errString)
```

controllerAddress: Address of Controller

inStageIdentifier: inStageIdentifier.

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 stage identifier.

2.2.2.19 **JR\_Get**

**Syntax**

```c
int JR_Get(int controllerAddress, out double outJerkTime, out string errString)
```

controllerAddress: Address of Controller

outJerkTime: outJerkTime

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous JR Get command which is used to Get jerk time.

2.2.2.20 **JR\_Set**

**Syntax**

```c
int JR_Set(int controllerAddress, double inJerkTime, out string errString)
```

controllerAddress: Address of Controller

inJerkTime: inJerkTime.

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous JR Set command which is used to Set jerk time.
2.2.2.21 KD_Get

**Syntax**

```c
int KD_Get(int controllerAddress, out double outDerivativeGain, out string errString)
```

controllerAddress: Address of Controller
outDerivativeGain: outDerivativeGain
errString: The failure reason
return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous KD Get command which is used to Get derivative gain.

2.2.2.22 KD_Set

**Syntax**

```c
int KD_Set(int controllerAddress, double inDerivativeGain, out string errString)
```

controllerAddress: Address of Controller
inDerivativeGain: inDerivativeGain.
errString: The failure reason
return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous KD Set command which is used to Set derivative gain.

2.2.2.23 KI_Get

**Syntax**

```c
int KI_Get(int controllerAddress, out double outIntegralGain, out string errString)
```

controllerAddress: Address of Controller
outIntegralGain: outIntegralGain
errString: The failure reason
return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous KI Get command which is used to Get integral gain.
2.2.2.24 **KI_Set**

**Syntax**

```c
int KI_Set(int controllerAddress, double inIntegralGain, out string errString)
```

controllerAddress: Address of Controller  
inIntegralGain: inIntegralGain.  
errString: The failure reason  
return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous KI Set command which is used to Set integral gain.

2.2.2.25 **KP_Get**

**Syntax**

```c
int KP_Get(int controllerAddress, out double outProportionalGain, out string errString)
```

controllerAddress: Address of Controller  
outProportionalGain: outProportionalGain  
errString: The failure reason  
return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous KP Get command which is used to Get proportional gain.

2.2.2.26 **KP_Set**

**Syntax**

```c
int KP_Set(int controllerAddress, double inProportionalGain, out string errString)
```

controllerAddress: Address of Controller  
inProportionalGain: inProportionalGain.  
errString: The failure reason  
return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous KP Set command which is used to Set proportional gain.
2.2.2.27  KV_Get

**Syntax**

```c
int KV_Get(int controllerAddress, out double outVelocityFeedForward, out string errString)
```

- `controllerAddress`: Address of Controller
- `outVelocityFeedForward`: outVelocityFeedForward
- `errString`: The failure reason

**Description**

This function is used to process synchronous KV Get command which is used to Get velocity feed forward.

2.2.2.28  KV_Set

**Syntax**

```c
int KV_Set(int controllerAddress, double inVelocityFeedForward, out string errString)
```

- `controllerAddress`: Address of Controller
- `inVelocityFeedForward`: inVelocityFeedForward
- `errString`: The failure reason

**Description**

This function is used to process synchronous KV Set command which is used to Set velocity feed forward.

2.2.2.29  MM_Get

**Syntax**

```c
int MM_Get(int controllerAddress, out string outState, out string errString)
```

- `controllerAddress`: Address of Controller
- `outState`: outState
- `errString`: The failure reason

**Description**

This function is used to process synchronous MM Get command which is used to Enter/Leave DISABLE state.
2.2.2.30  **MM_Set**

**Syntax**

```c
int MM_Set(int controllerAddress, int inState, out string errString)
```

- **controllerAddress**: Address of Controller
- **inState**: inState.
- **errString**: The failure reason
- **return**: 0 in success and -1 on failure

**Description**

This function is used to process synchronous MM Set command which is used to Enter/Leave DISABLE state.

2.2.2.31  **OH_Get**

**Syntax**

```c
int OH_Get(int controllerAddress, out double outHomeVelocity, out string errString)
```

- **controllerAddress**: Address of Controller
- **outHomeVelocity**: outHomeVelocity
- **errString**: The failure reason
- **return**: 0 in success and -1 on failure

**Description**

This function is used to process synchronous OH Get command which is used to Get HOME search velocity.

2.2.2.32  **OH_Set**

**Syntax**

```c
int OH_Set(int controllerAddress, double inHomeVelocity, out string errString)
```

- **controllerAddress**: Address of Controller
- **inHomeVelocity**: inHomeVelocity.
- **errString**: The failure reason
- **return**: 0 in success and -1 on failure

**Description**

This function is used to process synchronous OH Set command which is used to Set HOME search velocity.


2.2.2.33 **OR**

**Syntax**

int OR(int controllerAddress, out string errString)

clientID: Instrument ID
ccontrollerAddress: controllerAddress identifying the Address of Controller
erString: The failure reason
return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous OR Set command which is used to Execute HOME search.

2.2.2.34 **OT_Get**

**Syntax**

int OT_Get(int controllerAddress, out double outHomeTimeOut, out string errString)

ccontrollerAddress: Address of Controller
dooutHomeTimeOut: outHomeTimeOut
erString: The failure reason
return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous OT Get command which is used to Get HOME search time-out.

2.2.2.35 **OT_Set**

**Syntax**

int OT_Set(int controllerAddress, double inHomeTimeOut, out string errString)

ccontrollerAddress: Address of Controller
dinHomeTimeOut: inHomeTimeOut.
erString: The failure reason
return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous OT Set command which is used to Set HOME search time-out.
2.2.2.36 **PA_Get**

**Syntax**

```c
int PA_Get(int controllerAddress, out double outTargetPosition, out string errString)
```

- **controllerAddress**: Address of Controller
- **outTargetPosition**: The failure reason
- **errString**: The failure reason

**Description**

This function is used to process synchronous PA Get command which is used to Move absolute.

2.2.2.37 **PA_Set**

**Syntax**

```c
int PA_Set(int controllerAddress, double inTargetPosition, out string errString)
```

- **controllerAddress**: Address of Controller
- **inTargetPosition**: The failure reason
- **errString**: The failure reason

**Description**

This function is used to process synchronous PA Set command which is used to Move absolute.

2.2.2.38 **PR_Get**

**Syntax**

```c
int PR_Get(int controllerAddress, out double outStep, out string errString)
```

- **controllerAddress**: Address of Controller
- **outStep**: The failure reason
- **errString**: The failure reason

**Description**

This function is used to process synchronous PR Get command which is used to Move relative.
2.2.2.39 **PR_Set**

**Syntax**

```c
int PR_Set(int controllerAddress, double inStep, out string errString)
```

- `controllerAddress`: Address of Controller
- `inStep`: The failure reason
- `errString`: The failure reason
- `return`: 0 in success and -1 on failure

**Description**

This function is used to process synchronous PR Set command which is used to Move relative.

2.2.2.40 **PT_Get**

**Syntax**

```c
int PT_Get(int controllerAddress, out double outMotionTime, out string errString)
```

- `controllerAddress`: Address of Controller
- `outMotionTime`: The failure reason
- `errString`: The failure reason
- `return`: 0 in success and -1 on failure

**Description**

This function is used to process synchronous PT Get command which is used to Get motion time for a relative move.

2.2.2.41 **PT_Set**

**Syntax**

```c
int PT_Set(int controllerAddress, double inMotionTime, out string errString)
```

- `controllerAddress`: Address of Controller
- `inMotionTime`: The failure reason
- `errString`: The failure reason
- `return`: 0 in success and -1 on failure

**Description**

This function is used to process synchronous PT Set command which is used to Get motion time for a relative move.
2.2.2.42 PW_Get

**Syntax**

```c
int PW_Get(int controllerAddress, out int outState, out string errString)
```

- `controllerAddress`: Address of Controller
- `outState`: outState
- `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.

2.2.2.43 PW_Set

**Syntax**

```c
int PW_Set(int controllerAddress, int inState, out string errString)
```

- `controllerAddress`: Address of Controller
- `inState`: inState
- `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.

---

**NOTE**

The PW command is limited to 100 writes. Unit failure due to excessive use of the PW command is not covered by warranty.

The PW command is used to change the configuration parameters that are stored in memory, and not parameters that are needed to be changed on the fly.

---

2.2.2.44 QIL_Get

**Syntax**

```c
int QIL_Get(int controllerAddress, out double outMotorPeakLimit, out string errString)
```

- `controllerAddress`: Address of Controller
- `outMotorPeakLimit`: outMotorPeakLimit
- `errString`: The failure reason
- `return`: 0 in success and -1 on failure

**Description**

This function is used to process synchronous QIL Get command which is used to Get motor’s peak current limits.
2.2.2.45 **QIL\_Set**

**Syntax**

```
int QIL_Set(int controllerAddress, double inMotorPeakLimit, out string errString)
controllerAddress: Address of Controller
inMotorPeakLimit: inMotorPeakLimit.
errString: The failure reason
return: 0 in success and -1 on failure
```

**Description**

This function is used to process synchronous QIL Set command which is used to Set motor’s peak current limits.

2.2.2.46 **QIR\_Get**

**Syntax**

```
int QIR_Get(int controllerAddress, out double outMotorMsLimit, out string errString)
controllerAddress: Address of Controller
outMotorMsLimit: outMotorMsLimit
errString: The failure reason
return: 0 in success and -1 on failure
```

**Description**

This function is used to process synchronous QIR Get command which is used to Get motor’s ms current limits.

2.2.2.47 **QIR\_Set**

**Syntax**

```
int QIR_Set(int controllerAddress, double inMotorMsLimit, out string errString)
controllerAddress: Address of Controller
inMotorMsLimit: inMotorMsLimit.
errString: The failure reason
return: 0 in success and -1 on failure
```

**Description**

This function is used to process synchronous QIR Set command which is used to Set motor’s ms current limits.
2.2.2.48 **QIT\_Get**

**Syntax**

```c
int QIT_Get(int controllerAddress, out double outMotorAveragingTime, out string errString)
```

- `controllerAddress`: Address of Controller
- `outMotorAveragingTime`: outMotorAveragingTime
- `errString`: The failure reason

**Description**

This function is used to process synchronous QIT Get command which is used to Get motor’s ms current averaging time.

2.2.2.49 **QIT\_Set**

**Syntax**

```c
int QIT_Set(int controllerAddress, double inMotorAveragingTime, out string errString)
```

- `controllerAddress`: Address of Controller
- `inMotorAveragingTime`: inMotorAveragingTime
- `errString`: The failure reason

**Description**

This function is used to process synchronous QIT Set command which is used to Set motor’s ms current averaging time.

2.2.2.50 **RS**

**Syntax**

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

- `controllerAddress`: controllerAddress identifying the Address of Controller
- `errString`: The failure reason

**Description**

This function is used to process synchronous RS Set command which is used to Reset controller.
### 2.2.2.51 RS485

**Syntax**

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

- `clientID`: Instrument ID
- `controllerAddress`: controllerAddress identifying the Address of Controller
- `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’s address to 1.

---

### 2.2.2.52 SA_Get

**Syntax**

```c
int SA_Get(int controllerAddress, out int outRS485Address, out string errString)
```

- `controllerAddress`: Address of Controller
- `outRS485Address`: outRS485Address
- `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.

---

### 2.2.2.53 SA_Set

**Syntax**

```c
int SA_Set(int controllerAddress, int inRS485Address, out string errString)
```

- `controllerAddress`: Address of Controller
- `inRS485Address`: inRS485Address
- `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.
2.2.2.54  SC\_Get

**Syntax**

```
int SC_Get(int controllerAddress, out int outControlLoopState, out string errString)
```

- **controllerAddress**: Address of Controller
- **outControlLoopState**: outControlLoopState
- **errString**: The failure reason
- **return**: 0 in success and -1 on failure

**Description**

This function is used to process synchronous SC Get command which is used to Get control loop state.

2.2.2.55  SC\_Set

**Syntax**

```
int SC_Set(int controllerAddress, int inControlLoopState, out string errString)
```

- **controllerAddress**: Address of Controller
- **inControlLoopState**: inControlLoopState
- **errString**: The failure reason
- **return**: 0 in success and -1 on failure

**Description**

This function is used to process synchronous SC Set command which is used to Set control loop state.

2.2.2.56  SE

**Syntax**

```
int SE(int controllerAddress, double inTargetPosition, out string errString)
```

- **controllerAddress**: Address of Controller
- **inTargetPosition**: inTargetPosition
- **errString**: The failure reason
- **return**: 0 in success and -1 on failure

**Description**

This function is used to process synchronous SE Set command which is used to Configure/Execute simultaneous started move.
2.2.2.57  **SL_Get**

**Syntax**

```c
int SL_Get(int controllerAddress, out double outNegativeLimit, out string errString)
```

- `controllerAddress`: Address of Controller
- `outNegativeLimit`: outNegativeLimit
- `errString`: The failure reason
- `return`: 0 in success and -1 on failure

**Description**

This function is used to process synchronous SL Get command which is used to Get negative software limit.

2.2.2.58  **SL_Set**

**Syntax**

```c
int SL_Set(int controllerAddress, double inNegativeLimit, out string errString)
```

- `controllerAddress`: Address of Controller
- `inNegativeLimit`: inNegativeLimit
- `errString`: The failure reason
- `return`: 0 in success and -1 on failure

**Description**

This function is used to process synchronous SL Set command which is used to Set negative software limit.

2.2.2.59  **SR_Get**

**Syntax**

```c
int SR_Get(int controllerAddress, out double outPositiveLimit, out string errString)
```

- `controllerAddress`: Address of Controller
- `outPositiveLimit`: outPositiveLimit
- `errString`: The failure reason
- `return`: 0 in success and -1 on failure

**Description**

This function is used to process synchronous SR Get command which is used to Get positive software limit.
2.2.2.60 SR_Set

**Syntax**

```c
int SR_Set(int controllerAddress, double inPositiveLimit, out string errString)
```
controllerAddress: Address of Controller
errString: The failure reason
return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous SR Set command which is used to Set positive software limit.

2.2.2.61 ST

**Syntax**

```c
int ST(int controllerAddress, out string errString)
```
controllerAddress: controllerAddress identifying the Address of Controller
errString: The failure reason
return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous ST Set command which is used to Stop motion.

2.2.2.62 SU_Get

**Syntax**

```c
int SU_Get(int controllerAddress, out double outEncoderIncrement, out string errString)
```
controllerAddress: Address of Controller
outEncoderIncrement: outEncoderIncrement
errString: The failure reason
return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous SU Get command which is used to Get encoder increment value.
2.2.2.63 SU_Set

**Syntax**

```c
int SU_Set(int controllerAddress, double inEncoderIncrement, out string errString)
controllerAddress: Address of Controller
inEncoderIncrement: inEncoderIncrement.
errString: The failure reason
return: 0 in success and -1 on failure
```

**Description**

This function is used to process synchronous SU Set command which is used to Set encoder increment value.

2.2.2.64 TB

**Syntax**

```c
int TB(int controllerAddress, string inError, out string outError, out string errString)
controllerAddress: Address of Controller
inError: inError.
outError: outError
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.

2.2.2.65 TE

**Syntax**

```c
int TE(int controllerAddress, out string outError, out string errString)
controllerAddress: Address of Controller
outError: outError
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.
2.2.2.66 **TH**

**Syntax**

```c
int TH(int controllerAddress, out double outSetPointPosition, out string errString)
```

controllerAddress: Address of Controller
outSetPointPosition: outSetPointPosition
errString: The failure reason

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

**Description**

This function is used to process synchronous TH Get command which is used to Get set-point position.

2.2.2.67 **TK_Get**

**Syntax**

```c
int TK_Get(int controllerAddress, out string outState, out string errString)
```

controllerAddress: Address of Controller
outState: outState
errString: The failure reason

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

**Description**

This function is used to process synchronous TK Get command which is used to Enter/Leave ReadyT state.

2.2.2.68 **TK_Set**

**Syntax**

```c
int TK_Set(int controllerAddress, int inState, out string errString)
```

controllerAddress: Address of Controller
inState: inState
errString: The failure reason

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

**Description**

This function is used to process synchronous TK Set command which is used to Enter/Leave ReadyT state.
2.2.2.69 TP

Syntax
int TP(int controllerAddress, out double outCurrentPosition, out string errString)
controllerAddress: Address of Controller
outCurrentPosition: outCurrentPosition
errString: The failure reason
return: 0 in success and -1 on failure

Description
This function is used to process synchronous TP Get command which is used to Get current position.

2.2.2.70 TS

Syntax
int TS(int controllerAddress, out string errorCode, out string controllerState, out string errString)
controllerAddress: Address of Controller
errorCode: errorCode
controllerState: controllerState
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.

2.2.2.71 VA_Get

Syntax
int VA_Get(int controllerAddress, out double outVelocity, out string errString)
controllerAddress: Address of Controller
outVelocity: outVelocity
errString: The failure reason
return: 0 in success and -1 on failure

Description
This function is used to process synchronous VA Get command which is used to Get velocity.
2.2.2.72 **VA_Set**

**Syntax**

```
int VA_Set(int controllerAddress, double inVelocity, out string errString)
```

- `controllerAddress`: Address of Controller
- `inVelocity`: inVelocity.
- `errString`: The failure reason

**Description**

This function is used to process synchronous VA Set command which is used to Set velocity.

2.2.2.73 **VE**

**Syntax**

```
int VE(int controllerAddress, out string outControllerVersion, out string errString)
```

- `controllerAddress`: Address of Controller
- `outControllerVersion`: outControllerVersion
- `errString`: The failure reason

**Description**

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

2.2.2.74 **ZT**

**Syntax**

```
int ZT(int controllerAddress, out List<string> AxisParameters, out string errString)
```

- `controllerAddress`: Address of Controller
- `AxisParameters`: AxisParameters
- `errString`: The failure reason

**Description**

This function is used to process synchronous ZT Get command which is used to Get all axis parameters.
3.0 Python Example

```python
# Initialization Start
# The script within Initialization Start and Initialization End is needed for properly
# initializing Command Interface for CONEX-CC 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.CONEXCC.CommandInterface.dll to sys.path"
sys.path.append(r'C:\Program Files\Newport\MotionControl\CONEX-CC\Bin')
sys.path.append(r'C:\Program Files (x86)\Newport\MotionControl\CONEX-CC\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.CONEXCC.CommandInterface.dll")
from CommandInterface import *

import System

# Instrument Initialization
# The key should have double slashes since
# (one of them is escape character)
instrument="COM25"
print 'Instrument Key=>', instrument

# create a device instance and open communication with the instrument
CC = ConexCC()
ret = CC.OpenInstrument(instrumentKey)
print 'OpenInstrument => ', ret

# Get positive software limit
result, response, errString = CC.SR_Get(1)
if result == 0 :
    print 'positive software limit=>', response
else:
    print 'Error=>', errString
```

EDH0278En1032 — 12/18
Page 28
# Get negative software limit
result, response, errString = CC.SL_Get(1)
if result == 0:
    print 'negative software limit=>', response
else:
    print 'Error=>', errString

# Get controller revision information
result, response, errString = CC.VE(1)
if result == 0:
    print 'controller revision=>', response
else:
    print 'Error=>', errString

# Get current position
result, response, errString = CC.TP(1)
if result == 0:
    print 'position=>', response
else:
    print 'Error=>', errString

# Unregister device
CC.CloseInstrument();
Service Form

Your Local Representative
Tel.: __________________
Fax: __________________

Name: _________________________________________________  Return authorization #: ____________________________________

Company:_______________________________________________
(Please obtain prior to return of item)

Address: ________________________________________________  Date: __________________________________________________

Country: ________________________________________________  Phone Number: _______________________________________

P.O. Number: ____________________________________________  Fax Number: ________________________________________

Item(s) Being Returned: ____________________________________

Model#: ________________________________________________  Serial #: __________________________________________

Description: ____________________________________________________________________________________________

Reasons of return of goods (please list any specific problems): ____________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
________________________________________________________________________________________________________
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