mks

# SMC100CC & SMC100PP

## Single-Axix Motion Controller/Driver for DC or Stepper Motor

STATE DIAGRAM
Newport • Motion Controller / Driver • SMC100 Series
KEYPAD
RS232C
RS485 IN
RS485 OUT
CONFIG
Newport • Keypad • SMC-RC
Exec.

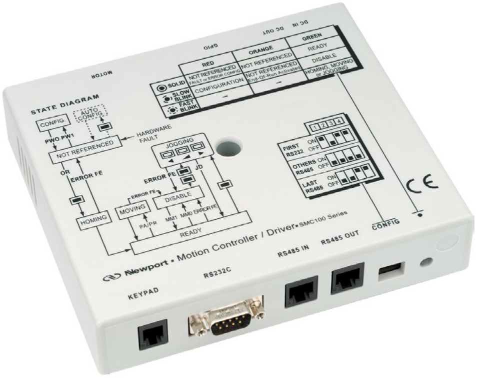
ESP COMPATIBLE

Newport®

# Command Interface Manual

V1.0.x

# Table of Contents



# Single-Axis Motion Controller/Driver SMC100CC & SMC100PP

## 1.0 Introduction

### 1.1 Purpose

The purpose of this document is to provide the method syntax of each command to communicate with the SMC100 device exposed in assembly Newport.SMC100.CommandInterface.dll. This .Net assembly is designed and developed by Newport. This DLL is used by SMC100 applet to communicate with SMC100 instrument.

### 1.2 Overview

Typically this DLL exposes all the commands the instrument supports. These commands are exposed as function calls of the Command Interface DLL.

Newport.SMC100.CommandInterface.dll is the assembly used for communicating with SMC100 instrument. This assembly gets installed when SMC100 applet is installed.

**NOTE**

**Each function name is defined with the command code “AA”.**

**For each command function, refer to the SMC100 programmer’s manual.**

## 1.3 Location

Newport.SMC100.CommandInterface.dll is located at
Program Installation Directory\Newport\MotionControl\SMC100\Bin

## 1.4 Possible uses of Newport.SMC100.CommandInterface.dll

Newport.SMC100.CommandInterface.dll is used by SMC100 applet for communicating with SMC100 instrument. The same DLL can be used as a reusable software component for creating Python script or for creating LabVIEW VIs.

# 2.0 Command Interface

## 2.1 Constructor

SMC100()

The constructor is used to create an instance of the SMC100 device.

## 2.2 Functions

### 2.2.1 General

### 2.2.1.1 OpenInstrument

**Syntax**

int OpenInstrument(string strDeviceKey)

string strDeviceKey: the device key is a serial COM port

return: 0 = successful or -1 = failure

**Decription**

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

### 2.2.1.2 CloseInstrument

**Syntax**

int CloseInstrument()

return: 0 = successful or -1 = failure

**Decription**

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

### 2.2.1.3 GetDevices

**Syntax**

string[] GetDevices()

return: list of strings that contains the accessible COM ports.

**Decription**

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

### 2.2.1.4 WriteToInstrument

**Syntax**

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

command: Instrument command

response: Response of the command

stage: Instrument Stage

return:

**Decription**

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**

int AC_Get(int controllerAddress, out double Acceleration, out string errstring)

controllerAddress: controllerAddress

Acceleration: Acceleration

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. Refer to the Controller's manual to get the command description.

#### 2.2.2.2 AC_Set

**Syntax**

int AC_Set(int controllerAddress, double Acceleration, out string errstring)

controllerAddress: controllerAddress

Acceleration: Acceleration

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. Refer to the Controller's manual to get the command description.

#### 2.2.2.3 BA_Get

**Syntax**

int BA_Get(int controllerAddress, out double Backlash, out string errstring)

controllerAddress: controllerAddress

Backlash: Backlash

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. Refer to the Controller's manual to get the command description.

### 2.2.2.4 BA_Set

**Syntax**

int BA_Set(int controllerAddress, double Backlash, out string errstring)

controllerAddress: controllerAddress

Backlash: Backlash

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. Refer to the Controller's manual to get the command description.

### 2.2.2.5 BH_Get

**Syntax**

int BH_Get(int controllerAddress, out double Hysteresis, out string errstring)

controllerAddress: controllerAddress

Hysteresis: Hysteresis

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. Refer to the Controller's manual to get the command description.

### 2.2.2.6 BH_Set

**Syntax**

int BH_Set(int controllerAddress, double Hysteresis, out string errstring)

controllerAddress: controllerAddress

Hysteresis: Hysteresis

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous BH Set command which is used to Set hysteresis compensation. Refer to the Controller's manual to get the command description.

### 2.2.2.7 DV_Get

**Syntax**

int DV_Get(int controllerAddress, out double Voltage, out string errstring)

controllerAddress: controllerAddress

Voltage: Voltage

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous DV Get command which is used to Get driver voltage. Refer to the Controller's manual to get the command description.

### 2.2.2.8 DV_Set

**Syntax**

int DV_Set(int controllerAddress, double Voltage, out string errstring)

controllerAddress: controllerAddress

Voltage: Voltage

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous DV Set command which is used to Set driver voltage. Refer to the Controller's manual to get the command description.

### 2.2.2.9 FD_Get

**Syntax**

int FD_Get(int controllerAddress, out double LowPassFilter, out string errstring)

controllerAddress: controllerAddress

LowPassFilter: LowPassFilter

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous FD Get command which is used to Get low pass filter for Kd. Refer to the Controller's manual to get the command description.

### 2.2.2.10 FD_Set

**Syntax**

int FD_Set(int controllerAddress, double LowPassFilter, out string errstring)

controllerAddress: controllerAddress

LowPassFilter: LowPassFilter

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous FD Set command which is used to Set low pass filter for Kd. Refer to the Controller's manual to get the command description.

#### 2.2.2.11 FE_Get

**Syntax**

int FE_Get(int controllerAddress, out double FollowingError, out string errstring)

controllerAddress: controllerAddress

FollowingError: FollowingError

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous FE Get command which is used to Get following error limit. Refer to the Controller's manual to get the command description.

#### 2.2.2.12 FE_Set

**Syntax**

int FE_Set(int controllerAddress, double FollowingError, out string errstring)

controllerAddress: controllerAddress

FollowingError: FollowingError

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous FE Set command which is used to Set following error limit. Refer to the Controller's manual to get the command description.

#### 2.2.2.13 FF_Get

**Syntax**

int FF_Get(int controllerAddress, out double FrictionCompensation, out string errstring)

controllerAddress: controllerAddress

FrictionCompensation: FrictionCompensation

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous FF Get command which is used to Get friction compensation. Refer to the Controller's manual to get the command description.

#### 2.2.2.14 FF_Set

**Syntax**

int FF_Set(int controllerAddress, double FrictionCompensation, out string errstring)

controllerAddress: controllerAddress

FrictionCompensation: FrictionCompensation

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous FF Set command which is used to Set friction compensation. Refer to the Controller's manual to get the command description.

### 2.2.2.15 FRM_Get

**Syntax**

int FRM_Get(int controllerAddress, out int MicroStepPerFullStepFactor, out string errstring)

controllerAddress: controllerAddress

MicroStepPerFullStepFactor: MicroStepPerFullStepFactor

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous FRM Get command which is used to Get micro-step per full step factor for stepper motor. Refer to the Controller's manual to get the command description.

### 2.2.2.16 FRM_Set

**Syntax**

int FRM_Set(int controllerAddress, int MicroStepPerFullStepFactor, out string errstring)

controllerAddress: controllerAddress

MicroStepPerFullStepFactor: MicroStepPerFullStepFactor

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous FRM Set command which is used to Set micro-step per full step factor for stepper motor. Refer to the Controller's manual to get the command description.

### 2.2.2.17 FRS_Get

**Syntax**

int FRS_Get(int controllerAddress, out double DistancePerMotorFullStep, out string errstring)

controllerAddress: controllerAddress

DistancePerMotorFullStep: DistancePerMotorFullStep

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous FRS Get command which is used to Get the motion distance per motor's full step for stepper motor. Refer to the Controller's manual to get the command description.

### 2.2.2.18 FRS_Set

**Syntax**

int FRS_Set(int controllerAddress, double DistancePerMotorFullStep, out string errstring)

controllerAddress: controllerAddress

DistancePerMotorFullStep: DistancePerMotorFullStep

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous FRS Set command which is used to Set the motion distance per motor's full step for stepper motor. Refer to the Controller's manual to get the command description.

### 2.2.2.19 HT_Get

**Syntax**

int HT_Get(int controllerAddress, out int HomeType, out string errstring)

controllerAddress: controllerAddress

HomeType: HomeType

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. Refer to the Controller's manual to get the command description.

### 2.2.2.20 HT_Set

**Syntax**

int HT_Set(int controllerAddress, int HomeType, out string errstring)

controllerAddress: controllerAddress

HomeType: HomeType

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. Refer to the Controller's manual to get the command description.

### 2.2.2.21 ID_Get

**Syntax**

int ID_Get(int controllerAddress, out string StageIdentifier, out string errstring)

controllerAddress: controllerAddress

StageIdentifier: StageIdentifier

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. Refer to the Controller's manual to get the command description.

### 2.2.2.22 ID_Set

**Syntax**

int ID_Set(int controllerAddress, string StageIdentifier, out string errstring)

controllerAddress: controllerAddress

StageIdentifier: StageIdentifier

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. Refer to the Controller's manual to get the command description.

### 2.2.2.23 JD

**Syntax**

int JD(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 JD Set command which is used to Leave JOGGING state. Refer to the Controller's manual to get the command description.

### 2.2.2.24 JM_Get

**Syntax**

int JM_Get(int controllerAddress, out int KeypadState, out string errstring)

controllerAddress: controllerAddress

KeypadState: KeypadState

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous JM Get command which is used to Enable/Disable Keypad. Refer to the Controller's manual to get the command description.

### 2.2.2.25 JM_Set

**Syntax**

int JM_Set(int controllerAddress, int KeypadState, out string errstring)

controllerAddress: controllerAddress

KeypadState: KeypadState

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous JM Set command which is used to Enable/Disable Keypad. Refer to the Controller's manual to get the command description.

### 2.2.2.26 JR_Get

**Syntax**

int JR_Get(int controllerAddress, out double JerkTime, out string errstring)

controllerAddress: controllerAddress

JerkTime: JerkTime

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. Refer to the Controller's manual to get the command description.

### 2.2.2.27 JR_Set

**Syntax**

int JR_Set(int controllerAddress, double JerkTime, out string errstring)

controllerAddress: controllerAddress

JerkTime: JerkTime

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. Refer to the Controller's manual to get the command description.

### 2.2.2.28 KD_Get

**Syntax**

int KD_Get(int controllerAddress, out double DerivativeGain, out string errstring)

controllerAddress: controllerAddress

DerivativeGain: DerivativeGain

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. Refer to the Controller's manual to get the command description.

### 2.2.2.29 KD_Set

**Syntax**

int KD_Set(int controllerAddress, double DerivativeGain, out string errstring)

controllerAddress: controllerAddress

DerivativeGain: DerivativeGain

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. Refer to the Controller's manual to get the command description.

### 2.2.2.30 KI_Get

**Syntax**

int KI_Get(int controllerAddress, out double IntegralGain, out string errstring)

controllerAddress: controllerAddress

IntegralGain: IntegralGain

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. Refer to the Controller's manual to get the command description.

### 2.2.2.31 KI_Set

**Syntax**

int KI_Set(int controllerAddress, double IntegralGain, out string errstring)

controllerAddress: controllerAddress

IntegralGain: IntegralGain

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. Refer to the Controller's manual to get the command description.

### 2.2.2.32 KP_Get

**Syntax**

int KP_Get(int controllerAddress, out double ProportionalGain, out string errstring)

controllerAddress: controllerAddress

ProportionalGain: ProportionalGain

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. Refer to the Controller's manual to get the command description.

### 2.2.2.33 KP_Set

**Syntax**

int KP_Set(int controllerAddress, double ProportionalGain, out string errstring)

controllerAddress: controllerAddress

ProportionalGain: ProportionalGain

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. Refer to the Controller's manual to get the command description.

### 2.2.2.34 KV_Get

**Syntax**

int KV_Get(int controllerAddress, out double VelocityFeedForward, out string errstring)

controllerAddress: controllerAddress

VelocityFeedForward: VelocityFeedForward

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous KV Get command which is used to Get velocity feed forward. Refer to the Controller's manual to get the command description.

### 2.2.2.35 KV_Set

**Syntax**

int KV_Set(int controllerAddress, double VelocityFeedForward, out string errstring)

controllerAddress: controllerAddress

VelocityFeedForward: VelocityFeedForward

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous KV Set command which is used to Set velocity feed forward. Refer to the Controller's manual to get the command description.

### 2.2.2.36 MM_Get

**Syntax**

int MM_Get(int controllerAddress, out string DisableState, out string errstring)

controllerAddress: controllerAddress

DisableState: DisableState

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous MM Get command which is used to Enter/Leave DISABLE state. Refer to the Controller's manual to get the command description.

### 2.2.2.37 MM_Set

**Syntax**

int MM_Set(int controllerAddress, int DisableState, out string errstring)

controllerAddress: controllerAddress

DisableState: DisableState

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. Refer to the Controller's manual to get the command description.

### 2.2.2.38 OH_Get

**Syntax**

int OH_Get(int controllerAddress, out double HomeVelocity, out string errstring)

controllerAddress: controllerAddress

HomeVelocity: HomeVelocity

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. Refer to the Controller's manual to get the command description.

### 2.2.2.39 OH_Set

**Syntax**

int OH_Set(int controllerAddress, double HomeVelocity, out string errstring)

controllerAddress: controllerAddress

HomeVelocity: HomeVelocity

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. Refer to the Controller's manual to get the command description.

### 2.2.2.40 OR

**Syntax**

int OR(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 OR Set command which is used to Execute HOME search. Refer to the Controller's manual to get the command description.

### 2.2.2.41 OT_Get

**Syntax**

int OT_Get(int controllerAddress, out double HomeTimeOut, out string errstring)

controllerAddress: controllerAddress

HomeTimeOut: HomeTimeOut

errString: 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. Refer to the Controller's manual to get the command description.

### 2.2.2.42 OT_Set

**Syntax**

int OT_Set(int controllerAddress, double HomeTimeOut, out string errstring)

controllerAddress: controllerAddress

HomeTimeOut: HomeTimeOut

errString: 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. Refer to the Controller's manual to get the command description.

### 2.2.2.43 PA_Get

**Syntax**

int PA_Get(int controllerAddress, out double TargetPosition, out string errstring)

controllerAddress: controllerAddress

TargetPosition: TargetPosition

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous PA Get command which is used to Move absolute. Refer to the Controller's manual to get the command description.

### 2.2.2.44 PA_Set

**Syntax**

int PA_Set(int controllerAddress, double TargetPosition, out string errstring)

controllerAddress: controllerAddress

TargetPosition: TargetPosition

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous PA Set command which is used to Move absolute. Refer to the Controller's manual to get the command description.

### 2.2.2.45 PR_Get

**Syntax**

int PR_Get(int controllerAddress, out double Step, out string errstring)

controllerAddress: controllerAddress

Step: Step

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous PR Get command which is used to Move relative. Refer to the Controller's manual to get the command description.

### 2.2.2.46 PR_Set

**Syntax**

int PR_Set(int controllerAddress, double Step, out string errstring)

controllerAddress: controllerAddress

Step: Step

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. Refer to the Controller's manual to get the command description.

### 2.2.2.47 PT_Get

**Syntax**

int PT_Get(int controllerAddress, out double MotionTime, out string errstring)

controllerAddress: controllerAddress

MotionTime: MotionTime

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. Refer to the Controller's manual to get the command description.

### 2.2.2.48 PT_Set

**Syntax**

int PT_Set(int controllerAddress, double MotionTime, out string errstring)

controllerAddress: controllerAddress

MotionTime: MotionTime

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. Refer to the Controller's manual to get the command description.

### 2.2.2.49 PW_Get

**Syntax**

int PW_Get(int controllerAddress, out int ConfigurationMode, out string errstring)

controllerAddress: controllerAddress

ConfigurationMode: ConfigurationMode

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. Refer to the Controller's manual to get the command description.

### 2.2.2.50 PW_Set

**Syntax**

int PW_Set(int controllerAddress, int ConfigurationMode, out string errstring)

controllerAddress: controllerAddress

ConfigurationMode: ConfigurationMode

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. Refer to the Controller's manual to get the command description.

### 2.2.2.51 QIL_Get

**Syntax**

int QIL_Get(int controllerAddress, out double MotorPeakCurrentLimits, out string errstring)

controllerAddress: controllerAddress

MotorPeakCurrentLimits: MotorPeakCurrentLimits

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. Refer to the Controller's manual to get the command description.

### 2.2.2.52 QIL_Set

#### Syntax

int QIL_Set(int controllerAddress, double MotorPeakCurrentLimits, out string errstring)

controllerAddress: controllerAddress

MotorPeakCurrentLimits: MotorPeakCurrentLimits

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. Refer to the Controller's manual to get the command description.

### 2.2.2.53 QIR_Get

#### Syntax

int QIR_Get(int controllerAddress, out double MotorMsCurrentLimits, out string errstring)

controllerAddress: controllerAddress

MotorMsCurrentLimits: MotorMsCurrentLimits

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. Refer to the Controller's manual to get the command description.

### 2.2.2.54 QIR_Set

#### Syntax

int QIR_Set(int controllerAddress, double MotorMsCurrentLimits, out string errstring)

controllerAddress: controllerAddress

MotorMsCurrentLimits: MotorMsCurrentLimits

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. Refer to the Controller's manual to get the command description.

### 2.2.2.55 QIT_Get

**Syntax**

int QIT_Get(int controllerAddress, out double MotorMsCurrentAveragingTime, out string errstring)

controllerAddress: controllerAddress

MotorMsCurrentAveragingTime: MotorMsCurrentAveragingTime

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous QIT Get command which is used to Get motor′s ms current averaging time. Refer to the Controller's manual to get the command description.

### 2.2.2.56 QIT_Set

**Syntax**

int QIT_Set(int controllerAddress, double MotorMsCurrentAveragingTime, out string errstring)

controllerAddress: controllerAddress

MotorMsCurrentAveragingTime: MotorMsCurrentAveragingTime

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous QIT Set command which is used to Set motor′s ms current averaging time. Refer to the Controller's manual to get the command description.

### 2.2.2.57 RA

**Syntax**

int RA(nt controllerAddress, out double AnalogInputValue, out string errstring)

controllerAddress: controllerAddress

AnalogInputValue: AnalogInputValue

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 analog input value. Refer to the Controller's manual to get the command description.

### 2.2.2.58 RB

**Syntax**

int RB(int controllerAddress, out int TTLInputValue, out string errstring)

controllerAddress: controllerAddress

TTLInputValue: TTLInputValue

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous RB Get command which is used to Get TTL input value. Refer to the Controller's manual to get the command description.

### 2.2.2.59 RS

**Syntax**

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. Refer to the Controller's manual to get the command description.

### 2.2.2.60 SA_Get

**Syntax**

int SA_Get(int controllerAddress, out 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 Get command which is used to Get controller′s RS-485 address. Refer to the Controller's manual to get the command description.

### 2.2.2.61 SA_Set

**Syntax**

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. Refer to the Controller's manual to get the command description.

### 2.2.2.62 SB_Get

**Syntax**

int SB_Get(int controllerAddress, out int TTLOutputValue, out string errstring)

controllerAddress: controllerAddress

TTLOutputValue: TTLOutputValue

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous SB Get command which is used to Get TTL output value. Refer to the Controller's manual to get the command description.

### 2.2.2.63 SB_Set

**Syntax**

int SB_Set(int controllerAddress, int TTLOutputValue, out string errstring)

controllerAddress: controllerAddress

TTLOutputValue: TTLOutputValue

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous SB Set command which is used to Set TTL output value. Refer to the Controller's manual to get the command description.

### 2.2.2.64 SC_Get

**Syntax**

int SC_Get(int controllerAddress, out int ControlLoopState, out string errstring)

controllerAddress: controllerAddress

ControlLoopState: ControlLoopState

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. Refer to the Controller's manual to get the command description.

### 2.2.2.65 SC_Set

**Syntax**

int SC_Set(int controllerAddress, int ControlLoopState, out string errstring)

controllerAddress: controllerAddress

ControlLoopState: ControlLoopState

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. Refer to the Controller's manual to get the command description.

### 2.2.2.66 SE

**Syntax**

int SE(int controllerAddress, double TargetPosition, out string errstring)

controllerAddress: controllerAddress

TargetPosition: TargetPosition

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. Refer to the Controller's manual to get the command description.

### 2.2.2.67 SL_Get

**Syntax**

int SL_Get(int controllerAddress, out double LeftLimit, out string errstring)

controllerAddress: controllerAddress

LeftLimit: LeftLimit

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. Refer to the Controller's manual to get the command description.

### 2.2.2.68 SL_Set

**Syntax**

int SL_Set(int controllerAddress, double LeftLimit, out string errstring)

controllerAddress: controllerAddress

LeftLimit: LeftLimit

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. Refer to the Controller's manual to get the command description.

#### 2.2.2.69 SR_Get

**Syntax**

int SR_Get(int controllerAddress, out double RightLimit, out string errstring)

controllerAddress: controllerAddress

RightLimit: RightLimit

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. Refer to the Controller's manual to get the command description.

#### 2.2.2.70 SR_Set

**Syntax**

int SR_Set(int controllerAddress, double RightLimit, out string errstring)

controllerAddress: controllerAddress

RightLimit: RightLimit

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. Refer to the Controller's manual to get the command description.

#### 2.2.2.71 ST

**Syntax**

int ST(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 ST Set command which is used to Stop motion. Refer to the Controller's manual to get the command description.

### 2.2.2.72 SU_Get

**Syntax**

int SU_Get(int controllerAddress, out double EncoderIncrement, out string errstring)

controllerAddress: controllerAddress

EncoderIncrement: EncoderIncrement

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. Refer to the Controller's manual to get the command description.

### 2.2.2.73 SU_Set

**Syntax**

int SU_Set(int controllerAddress, double EncoderIncrement, out string errstring)

controllerAddress: controllerAddress

EncoderIncrement: EncoderIncrement

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. Refer to the Controller's manual to get the command description.

### 2.2.2.74 TB

**Syntax**

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. Refer to the Controller's manual to get the command description.

### 2.2.2.75 TE

#### Syntax

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. Refer to the Controller's manual to get the command description.

### 2.2.2.76 TH

#### Syntax

int TH(int controllerAddress, out double SetPointPosition, out string errstring)

controllerAddress: controllerAddress

SetPointPosition: SetPointPosition

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. Refer to the Controller's manual to get the command description.

### 2.2.2.77 TP

#### Syntax

int TP(int controllerAddress, out double CurrentPosition, out string errstring)

controllerAddress: controllerAddress

CurrentPosition: CurrentPosition

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. Refer to the Controller's manual to get the command description.

### 2.2.2.78 TS

**Syntax**

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. Refer to the Controller's manual to get the command description.

### 2.2.2.79 VA_Get

**Syntax**

int VA_Get(int controllerAddress, out double Velocity, out string errstring)

controllerAddress: controllerAddress

Velocity: Velocity

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. Refer to the Controller's manual to get the command description.

### 2.2.2.80 VA_Set

**Syntax**

int VA_Set(int controllerAddress, double Velocity, out string errstring)

controllerAddress: controllerAddress

Velocity: Velocity

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous VA Set command which is used to Set velocity. Refer to the Controller's manual to get the command description.

### 2.2.2.81 VB_Get

**Syntax**

int VB_Get(int controllerAddress, out double BaseVelocity, out string errstring)

controllerAddress: controllerAddress

BaseVelocity: BaseVelocity

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous VB Get command which is used to Get base velocity. Refer to the Controller's manual to get the command description.

### 2.2.2.82 VB_Set

**Syntax**

int VB_Set(int controllerAddress, double BaseVelocity, out string errstring)

controllerAddress: controllerAddress

BaseVelocity: BaseVelocity

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous VB Set command which is used to Set base velocity. Refer to the Controller's manual to get the command description.

### 2.2.2.83 VE

**Syntax**

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. Refer to the Controller's manual to get the command description.

### 2.2.2.84 ZT

**Syntax**

int ZT(int controllerAddress, out List<string> Parameters, out string errstring)

controllerAddress: controllerAddress

Parameters: Parameters

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous ZT Get command which is used to Get all axis parameters. Refer to the Controller's manual to get the command description.

### 2.2.2.85 ZX_Get

**Syntax**

int ZX_Get(int controllerAddress, out int Mode, out string errstring)

controllerAddress: controllerAddress

Mode: Mode

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous ZX Get command which is used to Get ESP stage configuration. Refer to the Controller's manual to get the command description.

### 2.2.2.86 ZX_Set

**Syntax**

int ZX_Set(int controllerAddress, int Mode, out string errstring)

controllerAddress: controllerAddress

Mode: Mode

errString: The failure reason

return: 0 in success and -1 on failure

**Description**

This function is used to process synchronous ZX Set command which is used to Set ESP stage configuration. Refer to the Controller's manual to get the command description.

## 3.0 Python example

```
#==============================================================
#Initialization Start
#The script within Initialization Start and Initialization End is needed for properly
#initializing IOPortClientLib and Command Interface for SMC100 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.SMC100.CommandInterface.dll to sys.path"
sys.path.append(r'C:\Program Files\Newport\MotionControl\SMC100\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.SMC100.CommandInterface.dll")
from CommandInterfaceSMC100 import *
import System
#==============================================================
# Instrument Initialization
instrument="COM5"
print 'Instrument Key=>', instrument
# create a device instance
SMC = SMC100()
result = SMC100.OpenInstrument(instrumentKey)
# Get positive software limit
result, response, errString = SMC.SR_Get(1)
if result == 0 :
  print 'positive software limit=>', response
else:
  print 'Error=>',errString
# Get negative software limit
result, response, errString = SMC.SL_Get(1)
if result == 0 :
  print 'negative software limit=>', response
else:
  print 'Error=>',errString
# Get controller revision information
result, response, errString = SMC.VE(1)
if result == 0 :
  print 'controller revision=>', response
else:
  print 'Error=>',errString
```

```
# Get current position
result, response, errString = SMC.TP(1)
if result == 0 :
  print 'position=>', response
else:
  print 'Error=>',errString
# Unregister device
SMC.UnregisterComponent();
```

# Service Form

**Your Local Representative**

Tel.: ___________________

Fax:____________________

Name: ______________________________________________

Company:____________________________________________

Address:____________________________________________

Country:____________________________________________

P.O. Number:________________________________________

Item(s) Being Returned:______________________________

Model#: ____________________________________________

Return authorization #: _______________________________

*(Please obtain prior to return of item)*

Date: ______________________________________________

Phone Number: ______________________________________

Fax Number: ________________________________________

Serial #: ___________________________________________

Description: ________________________________________________________________________

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

Newport®

Visit Newport Online at:

www.newport.com

**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

mks

Newport® Ophir® Spectra-Physics®