FC Series

Intelligent Stepper Motor Stages

Newport®
Experience | Solutions

Command Interface Manual

Version 1.0.x

For Motion, Think Newport™

Table of Contents

FC Series
Intelligent Stepper Motor Stages

1.0 Introduction

1.1 Purpose

The purpose of this document is to provide the method syntax of each command to communicate with the FC series device.

1.2 Overview

The Command Interface is the wrapper class that maintains a list of FC series stages. It exposes methods to communicate with any FC series device.

NOTE

Each function name is defined with the command code "AA".

For each command function, refer to the FC Series programmer's manual.

2.0 Command Interface

2.1 Constructor

FCStepper

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

2.2 Functions

2.2.1 General

2.2.1.1 OpenInstrument

Syntax

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

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

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

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

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 synchrounous AC Get command which is used to Get acceleration.

2.2.2.2 AC_Set

Syntax

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 synchrounous 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)

controllerAddress: Address of Controller

outBacklash: outBacklash

errString: The failure reason

return: 0 in success and -1 on failure

Description

This function is used to process synchrounous 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)

controllerAddress: Address of Controller

inBacklash: inBacklash.

errString: The failure reason

return: 0 in success and -1 on failure

Description

This function is used to process synchrounous 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)

controllerAddress: Address of Controller

outHysteresis: outHysteresis

errString: The failure reason

return: 0 in success and -1 on failure

Description

This function is used to process synchrounous BH Get command which is used to Get hysteresis compensation.

2.2.2.6 BH_Set

Syntax

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

controllerAddress: Address of Controller

inHysteresis: inHysteresis.

errString: The failure reason

return: 0 in success and -1 on failure

Description

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

2.2.2.1 FRM_Get

Syntax

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

controllerAddress : controller address

MicroStepPerFullStepFactor: Micro Step per Full Step Factor

errString: The failure reason

return: 0 in success and -1 on failure

Description

This function is used to process synchrounous FRM Get command which is used to get Micro Step per Full Step Factor. Refer to the Controller's manual to get the command description.

2.2.2.2 FRM_Set

Syntax

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

controllerAddress : controller address

MicroStepPerFullStepFactor: Micro Step per Full Step Factor

errString: The failure reason

return: 0 in success and -1 on failure

Description

This function is used to process synchrounous FRM Set command which is used to set Micro Step per Full Step Factor. Refer to the Controller's manual to get the command description.

2.2.2.3 FRS_Get

Syntax

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

controllerAddress : controller address

DistancePerMotorFullStep: Distance per Motor Full Step

errString: The failure reason

return: 0 in success and -1 on failure

Description

This function is used to process synchrounous FRS Get command which is used to get the distance per Motor Full Step. Refer to the Controller's manual to get the command description.

2.2.2.4 FRS_Set

Syntax

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

controllerAddress : controller address

DistancePerMotorFullStep: Distance per Motor Full Step

errString: The failure reason

return: 0 in success and -1 on failure

Description

This function is used to process synchrounous FRS Set command which is used to set the distance per Motor Full Step. Refer to the Controller's manual to get the command description.

2.2.2.5 HT_Get

Syntax

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 synchrounous HT Get command which is used to Get HOME search type.

2.2.2.6 HT_Set

Syntax

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 synchrounous HT Set command which is used to Set HOME search type.

2.2.2.7 ID_Get

Syntax

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 synchrounous ID Get command which is used to Get stage identifier.

2.2.2.8 ID_Set

Syntax

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 synchrounous ID Set command which is used to Set stage identifier.

2.2.2.9 JR_Get

Syntax

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 synchrounous JR Get command which is used to Get jerk time.

2.2.2.10 JR_Set

Syntax

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 synchrounous JR Set command which is used to Set jerk time.

2.2.2.11 MM_Set

Syntax

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 synchrounous MM Set command which is used to Enter/Leave DISABLE state.

2.2.2.12 OH_Get

Syntax

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 synchrounous OH Get command which is used to Get HOME search velocity.

2.2.2.13 OH_Set

Syntax

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 synchrounous OH Set command which is used to Set HOME search velocity.

2.2.2.14 OR

Syntax

int OR(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 synchrounous OR Set command which is used to Execute HOME search.

2.2.2.15 OT_Get

Syntax

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

controllerAddress: Address of Controller

outHomeTimeOut: outHomeTimeOut

errString: The failure reason

return: 0 in success and -1 on failure

Description

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

2.2.2.16 OT_Set

Syntax

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

controllerAddress: Address of Controller

inHomeTimeOut: inHomeTimeOut.

errString: The failure reason

return: 0 in success and -1 on failure

Description

This function is used to process synchrounous OT Set command which is used to Set HOME search time-out.

2.2.2.17 PA_Get

Syntax

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

controllerAddress: Address of Controller

outTargetPosition: outTargetPosition

errString: The failure reason

return: 0 in success and -1 on failure

Description

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

2.2.2.18 PA_Set

Syntax

int PA_Set(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 synchrounous PA Set command which is used to Move absolute.

2.2.2.19 PR_Get

Syntax

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

controllerAddress: Address of Controller

outStep: outStep

errString: The failure reason

return: 0 in success and -1 on failure

Description

This function is used to process synchrounous PR Get command which is used to Move relative.

2.2.2.20 PR_Set

Syntax

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

controllerAddress: Address of Controller

inStep: inStep.

errString: The failure reason

return: 0 in success and -1 on failure

Description

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

2.2.2.21 PT_Get

Syntax

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

controllerAddress: Address of Controller

outMotionTime: outMotionTime

errString: The failure reason

return: 0 in success and -1 on failure

Description

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

2.2.2.22 PT_Set

Syntax

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

controllerAddress: Address of Controller

inMotionTime: inMotionTime.

errString: The failure reason

return: 0 in success and -1 on failure

Description

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

2.2.2.23 PW_Get

Syntax

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 synchrounous PW Get command which is used to Enter/Leave CONFIGURATION state.

2.2.2.24 PW_Set

Syntax

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 synchrounous 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.

2.2.2.25 RS

Syntax

int RS(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 synchrounous RS Set command which is used to Reset controller.

2.2.2.26 SA_Get

Syntax

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 synchrounous SA Get command which is used to Get controller's RS-485 address.

2.2.2.27 SA_Set

Syntax

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 synchrounous SA Set command which is used to Set controller's RS-485 address.

2.2.2.28 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 synchrounous SE Set command which is used to Configure/Execute simultaneous started move.

2.2.2.29 SL_Get

Syntax

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 synchrounous SL Get command which is used to Get negative software limit.

2.2.2.30 SL_Set

Syntax

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 synchrounous SL Set command which is used to Set negative software limit.

2.2.2.31 SR_Get

Syntax

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 synchrounous SR Get command which is used to Get positive software limit.

2.2.2.32 SR_Set

Syntax

int SR_Set(int controllerAddress, double inPositiveLimit, out string errString)

controllerAddress: Address of Controller

inPositiveLimit: inPositiveLimit.

errString: The failure reason

return: 0 in success and -1 on failure

Description

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

2.2.2.33 ST

Syntax

int ST(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 synchrounous ST Set command which is used to Stop motion.

2.2.2.34 TB

Syntax

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 synchrounous TB Get command which is used to Get command error string.

2.2.2.35 TE

Syntax

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 synchrounous TE Get command which is used to Get last command error.

2.2.2.36 TH

Syntax

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 synchrounous TH Get command which is used to Get set-point position.

2.2.2.37 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 synchrounous TP Get command which is used to Get current position.

2.2.2.38 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 synchrounous TS Get command which is used to Get positioner error and controller state.

2.2.2.39 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 synchrounous VA Get command which is used to Get velocity.

2.2.2.40 VA_Set

Syntax

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

controllerAddress: Address of Controller

inVelocity: inVelocity.

errString: The failure reason

return: 0 in success and -1 on failure

Description

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

2.2.2.41 VE

Syntax

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

controllerAddress: Address of Controller

outControllerVersion: outControllerVersion

errString: The failure reason

return: 0 in success and -1 on failure

Description

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

2.2.2.42 ZT

Syntax

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

controllerAddress: Address of Controller

AxisParameters: AxisParameters

errString: The failure reason

return: 0 in success and -1 on failure

Description

This function is used to process synchrounous ZT Get command which is used to Get all axis parameters.

3.0 Python Example

```
#============================================================
# Newport Proprietary and Confidential    Newport Corporation 2013
#
# No part of this file in any format, with or without modification
# shall be used, copied or distributed without the express written
# consent of Newport Corporation.
#
# Description: This is a Python Script to access FCStepper library
#============================================================

#============================================================
#Initialization Start
#The script within Initialization Start and Initialization End is needed for properly
#initializing Command Interface DLL for FCStepper 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.FCStepper.CommandInterface.dll to sys.path"
sys.path.append(r'C:\Program Files\Newport\MotionControl\FCStepper\Bin')
sys.path.append(r'C:\Program Files (x86)\Newport\MotionControl\FCStepper\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.FCStepper.CommandInterface.dll")
from CommandInterfaceFCStepper 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
myFCStepper = FCStepper()
ret = myFCStepper.OpenInstrument(instrumentKey)
print 'OpenInstrument => ', ret
```

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

# Get negative software limit
result, response, errString = myFCStepper.SL_Get(1)
if result == 0 :
  print 'negative software limit=>', response
else:
  print 'Error=>',errString

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

# Unregister device
myFCStepper.CloseInstrument();
```

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: _____

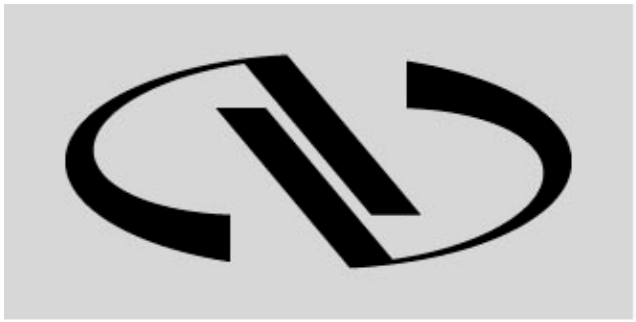
Phone Number: _____

Fax Number: _____

Serial #: _____

Description: _____

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



Newport®

Experience | Solutions

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