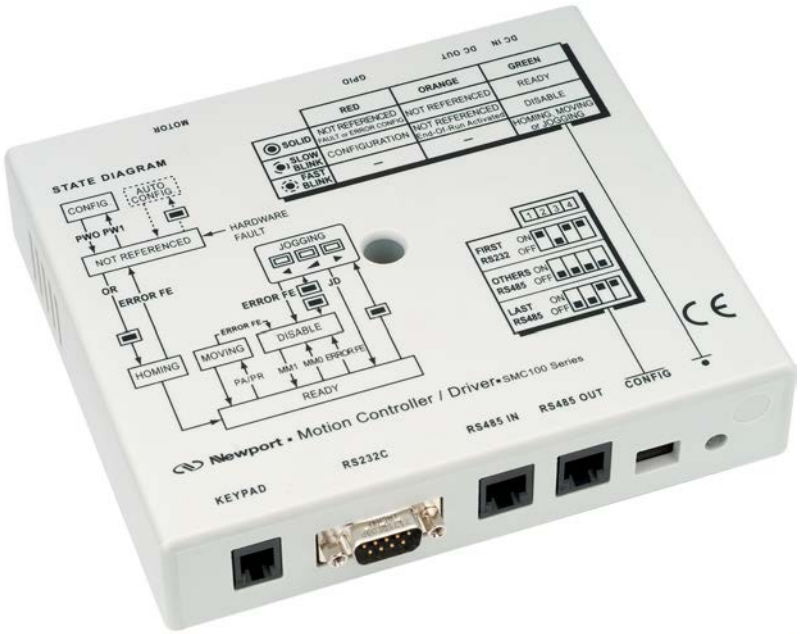
# SMC100CC & SMC100PP

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





ESP COMPATIBLE

RoHS Compliant

Newport®

Experience | Solutions

# User's Manual

**Firmware V3.0**

Precision Motion – *Guaranteed*™

# Table of Contents

# Warranty

Newport Corporation warrants that this product will be free from defects in material and workmanship and will comply with Newport's published specifications at the time of sale for a period of one year from date of shipment. If found to be defective during the warranty period, the product will either be repaired or replaced at Newport's option.

To exercise this warranty, write or call your local Newport office or representative, or contact Newport headquarters in Irvine, California. You will be given prompt assistance and return instructions. Send the product, freight prepaid, to the indicated service facility. Repairs will be made and the instrument returned freight prepaid. Repaired products are warranted for the remainder of the original warranty period or 90 days, whichever occurs last.

**Limitation of Warranty**

The above warranties do not apply to products which have been repaired or modified without Newport's written approval, or products subjected to unusual physical, thermal or electrical stress, improper installation, misuse, abuse, accident or negligence in use, storage, transportation or handling.

THIS WARRANTY IS IN LIEU OF ALL OTHER WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR USE. NEWPORT CORPORATION SHALL NOT BE LIABLE FOR ANY INDIRECT, SPECIAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM THE PURCHASE OR USE OF ITS PRODUCTS.

# EU Declaration of Conformity

## SMC100 Series

Newport
Experience | Solutions

**Year CE mark affixed: 2015**

## EC Declaration of Conformity

**The manufacturer:**

MICRO-CONTROLE Spectra-Physics,
9 rue du Bois Sauvage
F-91055 Evry FRANCE

**Hereby declares that the product:**

Description: "SMC100"
Function: Single-Axis Motion Controller/Driver
Type of equipment: Electrical equipment for measurement, control and laboratory use
Models: SMC100CC/PP; SMC-RC/-232/-USB/-PS80/-CB1/-CB3

– complies with all the relevant provisions of the Directive 2014/30/EU relating to electro-magnetic compatibility (EMC).

– was designed and built in accordance with the following harmonised standards:
NF EN 61326-1:2013 « Electrical equipment for measurement, control and laboratory use – EMC requirements – Part 1: General requirements »
NF EN 55011:2010/A1:2011 Class A

– was designed and built in accordance with the following other standards:
NF EN 61000-4-2
NF EN 61000-4-3
NF EN 61000-4-4
NF EN 61000-4-6
NF EN 61000-4-11

**Date : 26/06/2015**

Dominique DEVIDAL
Quality Director

*MICRO-CONTROLE Spectra-Physics*
*Zone Industrielle*
*F-45340 Beaune La Rolande, France*

DC2-EN rev:A

# Preface

## Confidentiality & Proprietary Rights

**Reservation of Title**

The Newport Programs and all materials furnished or produced in connection with them ("Related Materials") contain trade secrets of Newport and are for use only in the manner expressly permitted. Newport claims and reserves all rights and benefits afforded under law in the Programs provided by Newport Corporation.

Newport shall retain full ownership of Intellectual Property Rights in and to all development, process, align or assembly technologies developed and other derivative work that may be developed by Newport. Customer shall not challenge, or cause any third party to challenge, the rights of Newport.

**Preservation of Secrecy and Confidentiality and Restrictions to Access**

Customer shall protect the Newport Programs and Related Materials as trade secrets of Newport, and shall devote its best efforts to ensure that all its personnel protect the Newport Programs as trade secrets of Newport Corporation. Customer shall not at any time disclose Newport's trade secrets to any other person, firm, organization, or employee that does not need (consistent with Customer's right of use hereunder) to obtain access to the Newport Programs and Related Materials. These restrictions shall not apply to information (1) generally known to the public or obtainable from public sources; (2) readily apparent from the keyboard operations, visual display, or output reports of the Programs; (3) previously in the possession of Customer or subsequently developed or acquired without reliance on the Newport Programs; or (4) approved by Newport for release without restriction.

## Sales, Tech Support & Service

**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 Evry Cedex
France

**Sales France**
Tel.: +33 (0)1.60.91.68.68
e-mail: france@newport.com

**Sales Germany**
Tel.: +49 (0) 61 51 / 708 – 0
e-mail: germany@newport.com

**Sales UK**
Tel.: +44 (0)1635.521757
e-mail: uk@newport.com

**Technical Support**
e-mail: tech_europe@newport.com

**Service & Returns**
Tel.: +33 (0)2.38.40.51.55

## Service Information

The user should not attempt any maintenance or service of the SMC100 Controller/Driver and its accessories beyond the procedures outlined in this manual. Any problem that cannot be resolved should be referred to Newport Corporation. When calling Newport regarding a problem, please provide the Tech Support representative with the following information:

- Your contact information.
- System serial number or original order number.
- Description of problem.
- Environment in which the system is used.
- State of the system before the problem.
- Frequency and repeatability of problem.
- Can the product continue to operate with this problem?
- Can you identify anything that may have caused the problem?

## Newport Corporation RMA Procedures

Any SMC100 Controller/Driver being returned to Newport must have been assigned an RMA number by Newport. Assignment of the RMA requires the item serial number.

## Packaging

SMC100CC/PP Controller/Driver being returned under an RMA must be securely packaged for shipment. If possible, reuse the original factory packaging.

# SMC100
# Single-Axis Motion Controller

## 1.0 Introduction

### 1.1 Definitions and Symbols

The following terms and symbols are used in this documentation and also appear on the SMC100 Controller/Driver where safety-related issues occur.

#### 1.1.1 General Warning or Caution

*Figure 1: General Warning or Caution Symbol.*

The Exclamation Symbol in Figure 1 may appear in Warning and Caution tables in this document. This symbol designates an area where personal injury or damage to the equipment is possible.

#### 1.1.2 Electric Shock

*Figure 2: Electrical Shock Symbol.*

The Electrical Shock Symbol in Figure 2 may appear on labels affixed to the SMC100 Controller/Driver. This symbol indicates a hazard arising from dangerous voltage. Any mishandling could result in irreparable damage to the equipment, in personal injury, or death.

#### 1.1.3 European Union CE Mark

*Figure 3: CE Mark.*

The presence of the CE Mark on Newport Corporation equipment means that it has been designed, tested and certified as complying with all applicable European Union (CE) regulations and recommendations.

## 1.2 Warnings and Cautions

The following are definitions of the Warnings, Cautions and Notes that may be used in this manual to call attention to important information regarding personal safety, safety and preservation of the equipment, or important tips.

**WARNING**

**Situation has the potential to cause bodily harm or death.**

**CAUTION**

**Situation has the potential to cause damage to property or equipment.**

**NOTE**

**Additional information the user or operator should consider.**

## 1.3 General Warnings and Cautions

The following general safety precautions must be observed during all phases of operation of this equipment.

Failure to comply with these precautions or with specific warnings elsewhere in this manual violates safety standards of design, manufacture, and intended use of the equipment.

- Heed all warnings on the unit and in the operating instructions.
- To prevent damage to the equipment, read the instructions in this manual.
- Only plug the power supply to a grounded power outlet.
- Assure that the power supply is properly grounded to earth ground through the grounding lead of the AC power connector
- Route power cords and cables where they are not likely to be damaged.
- Disconnect or do not plug in the AC power cord in the following circumstances:
  - If the AC power cord or any other attached cables are frayed or damaged.
  - If the power plug or receptacle is damaged.
  - If the unit is exposed to rain or excessive moisture, or liquids are spilled on it.
  - If the unit has been dropped or the case is damaged.
  - If the user suspects service or repair is required.
- Keep air vents free of dirt and dust.
- Keep liquids away from unit.
- Do not expose equipment to excessive moisture (>85% humidity)
- Do not operate this equipment in an explosive atmosphere.
- Disconnect power before cleaning the Controller/Driver unit. Do not use liquid or aerosol cleaners.
- Do not open the SMC100CC/PP Controller/Driver. There are no user-serviceable parts inside.
- Return equipment to Newport Corporation for service and repair.
- Dangerous voltages associated with the 100-240 VAC power supply are present inside the power supply. To avoid injury, do not touch exposed connections or components while power is on.
- Follow precautions for static-sensitive devices when handling electronic circuits.

# 2.0 System Overview

## 2.1 General Description

The SMC100CC/PP is a single axis motion controller/driver for DC servo or stepper motors up to 48 VDC at 1.5 A rms. It provides a very compact and low-cost solution for driving a variety of Newport and other manufacturers motorized stages from a PC or from the optional SMC-RC remote control.

Communication with the SMC100CC/PP is achieved via a RS-232-C, or from a USB port using the external adapter SMC-USB (requires Windows™ operating system). A Windows™ based software supports all configurations and enables basic motion. Advanced application programming is simplified by an ASCII command interface and a set of two letter mnemonic commands.

When used with Newport ESP enhanced positioners, the SMC100CC/PP will detect the connected product automatically and provides easy configuration using the supplied Windows-based utility software. This exclusive Newport feature reduces configuration time and provides the best protection of your equipment from any accidental damages.

Up to 31 controllers can be networked through the internal RS-485 communication link. This internal multi-drop full-duplex serial link simplifies communication to several units, without the need for sending “address selection commands”. This results in enhanced multi-axes management with improved program readability and faster communication compared to alternative systems based on a RS-232-C chain. The typical execution time for a tell position command is only about 10 ms for the first controller and only about 16 ms for the other controllers. The SMC100CC/PP also features advanced “multi-axes” commands such as “Stop all” or “start a motion of all axes” and performs at a 57600 bauds rate communication speed. Furthermore, for an efficient process control, the SMC100CC/PP features dedicated digital outputs for "In Motion" and for "Not referenced".

## 2.2 Part Numbers

| Product | Description |
|---|---|
| **SMC100CC** | Single-axis motion controller/driver for DC servo motors.<br>Includes 0.2 m long power and RS-485 cable. |
| **SMC100PP** | Single-axis motion controller/driver for stepper motors.<br>Includes 0.2 m long power and RS-485 cable. |
| **SMC-RC** | Remote control keypad for SMC100CC/PP. |
| **SMC-PS80** | 80 W power supply for SMC100CC/PP. |
| **SMC-232** | RS-232-C cable, 3 m length (DB9F to DB9F). |
| **SMC-USB** | USB interface, Includes one USB to COM port adapter and one RS-232-C cable.<br>Requires Windows™ operating system. |
| **SMC-CB1** | 1 m RS-485 cable (only required when RS-485 cable supplied with SMC100CC/PP is too short). |
| **SMC-CB3** | 3 m RS-485 cable (only required when RS-485 cable supplied with SMC100CC/PP is too short). |

## 2.3 SMC100CC/PP

### 2.3.1 Contents of Delivery

- SMC100CC/PP Controller box
- SMC-PSC0.2 Power cable, 0.2 m length
- SMC-CB0.2 RS-485 network cable, 0.2 m length

### 2.3.2 Specifications

| | |
|---|---|
| General Description | Single-axis motion controller/driver for DC servo motors (DC version) and for stepper motors (stepper version) |
| Control Capability | DC servo motors, open or closed loop operation (DC version)<br>Stepper motors control, open loop operation only (stepper version) |
| Motor Output Power | – 48 VDC at 1.5 A rms, 3 A peak (DC version)<br>– 48 VDC at 1.1 A peak per phase (stepper version)<br>– 100 kHz PWM switching frequency |
| Control loop | – Floating point digital PID loop with velocity and friction feedforward<br>– 2 kHz servo rate<br>– Backlash compensation |
| Motion | Point-to-point motion with S-gamma profile and jerk time control |
| Computer interface | – RS-232-C with 57,600 baud rate<br>– USB compatible with external adapter SMC-USB (requires Windows™ operating system)<br>– RS-485 internal link for chaining up to 31 controllers from the same COM port |
| Programming | – 40+ intuitive, 2 letter ASCII commands<br>– Command set includes software limits, user units, synchronized motion start, stop all |
| General purpose I/O | – 4 TTL out (Open collector, 30 V/40 mA Max.)<br>– 4 TTL in (2.21 kΩ pull up to 5 V)<br>– 1 analog input, ±10 V, 8-Bit |
| Dedicated inputs | – RS-422 differential encoder inputs for A, B, and I, max. 2 MHz rate<br>– Forward and reverse limit, home switch and index pulse |
| Dedicated outputs | – 1 open-collector output for “In Motion”<br>– 1 open collector output for “Not Referenced” |
| Status display | Two color LED |
| Internal safety feature | Watchdog timer |

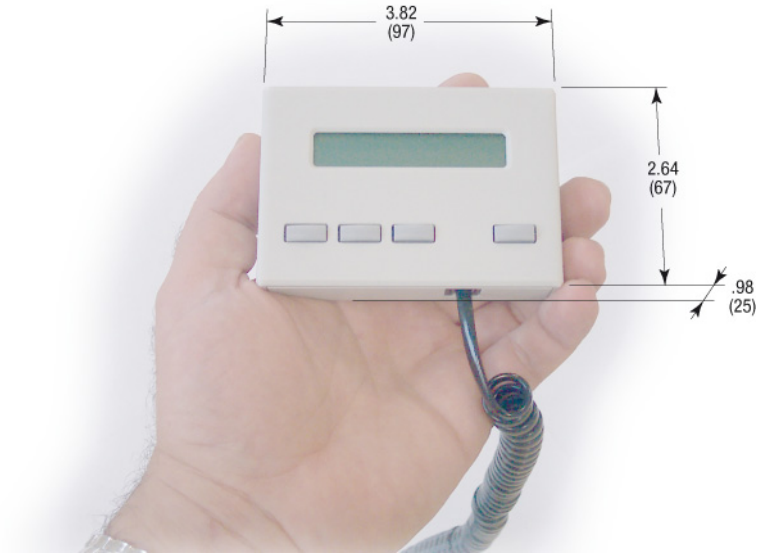
### 2.3.3 Dimensions

## 2.4 SMC-RC

### 2.4.1 Specifications

| | |
|---|---|
| General Description | Remote control keypad for SMC100CC/PP |
| Display | 1 line x 16 characters LCD display for position and short action description of Exec. button depending on controllers state |
| Function of push buttons | (from left to right)<br>– Jog left<br>– High jog velocity (when pressed together with left or jog right)<br>– Jog right<br>– Exec. (function as indicated in display depending on controllers state) |
| Cable | 0.5 m helix cable, both sides terminated with RJ11-4/4 connectors |

### 2.4.2 Dimensions

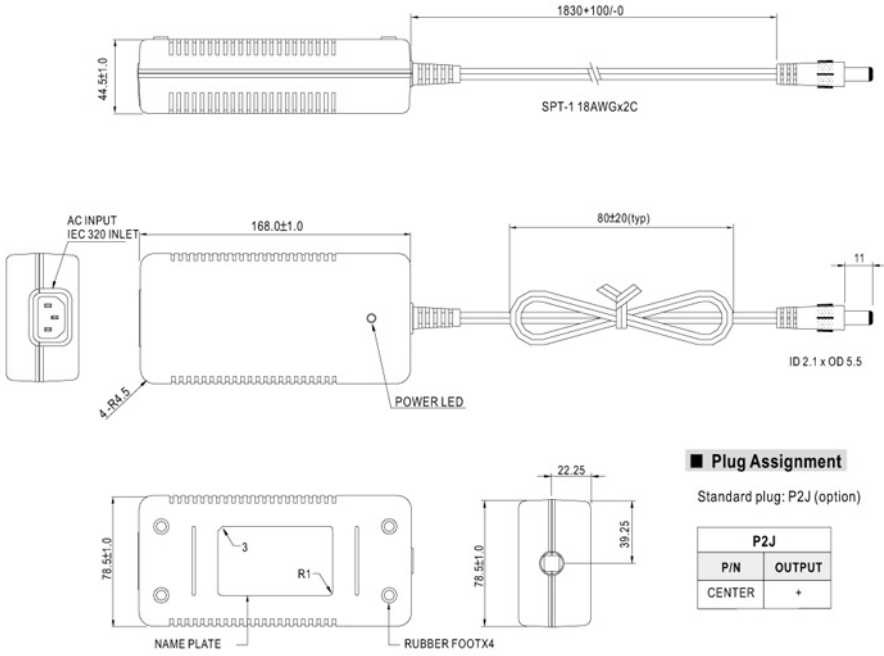


## 2.5 SMC-PS80

### 2.5.1 Specifications

| | |
|---|---|
| AC Input | 100–240 VAC, 47–63 Hz, 1.9 A |
| DC Output | 48 V, 80 W max., 1.87A, < 240mVp-p ripple and noise |
| Load and line regulation | Better than 2% |
| Connector | (male Ø 2.1 x Ø 5.5 x 11 mm) |

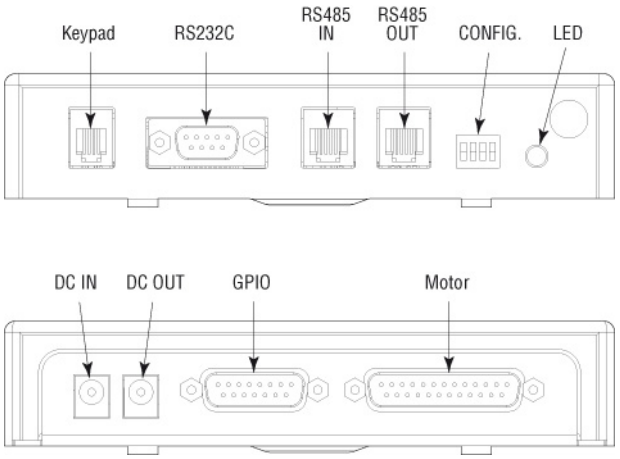
### 2.5.2 Dimensions




## 2.6 System Environmental Specifications

| | |
|---|---|
| Operating temperature | 5 °C to 40 °C |
| Operating humidity | < 85% relative humidity, non-condensing |
| Storage temperature | 0 °C to 60 °C<br>RH < 85% relative humidity, non-condensing |
| Installation category | II |
| Pollution degree | 2 |
| Use location | Indoor use only |

## 2.7 Connector Identification




### 2.7.1 Front side

| | |
|---|---|
| KEYPAD | RJ9F: For SMC-RC remote display and jog keypad. |
| RS-232-C | Sub-D9M: RS-232-C communication port for computer communication |
| RS-485 IN | RJ11F: RS-485 input for chaining several SMC100CC/PP in a multi-drop configuration |
| RS-485 OUT | RJ11F: RS-485 output for chaining several SMC100CC/PP in a multi-drop configuration |
| CONFIG. | 4 switches: Dip switches for communication setup |
| LED | LED: Status LED |

### 2.7.2 Back side

| | |
|---|---|
| DC IN | Ø 2.1 x Ø 5.5 x 11 mm: Power supply input (connect to SMC80-PS) |
| DC OUT | Ø 2.1 x Ø 5.5 x 11 mm: Power supply repeater for connecting several SMC100CC/PP to the same power supply |
| GPIO | Sub-D15F: General purpose inputs/outputs |
| MOTOR | Sub-D25F: Motor connection |

## 2.8 Serial Communication Settings

Communication parameters are preset in the SMC100CC/PP controller and do not require any configuration:

| | |
|---|---|
| Bits per second | 57,600 |
| Data bits | 8 |
| Parity | None |
| Stop bits | 1 |
| Flow control | Xon/Xoff |
| Terminator | $C_RL_F$ |

# 3.0 Getting Started

This section guides the user through the proper set-up of the SMC100CC/PP motion control system. When using the SMC100CC/PP controller ONLY in local control with the SMC-RC keypad and NOT from a computer, you can skip this section and continue reading in chapter 4.0, SMC100CC/PP with SMC-RC keypad. If not already done, carefully unpack and visually inspect the controllers and the stages for any damage. Place all components on a flat and clean surface.

**CAUTION**

**No cables should be connected to the controller at this point!**

First, the controller must be configured properly. When using several SMC100CC/PP controllers from the same COM port through the internal RS-485 communication link, an individual address must be set for each controller. Then, each controller must be configured to the connected stage. For both steps, the software supplied with the SMC100CC/PP is used.

## 3.1 Communication Settings

### 3.1.1 RS-232-C Communication (Using SMC-232 Cable)

Apply the following settings to the COM port of your PC:

| | |
|---|---|
| Bits per second | 57,600 |
| Data bits | 8 |
| Parity | None |
| Stop bits | 1 |
| Flow control | Xon/Xoff |
| Terminator | $C_RL_F$ |

### 3.1.2 USB Communication (Using SMC-USB Interface)

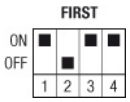
Install the software supplied with the SMC-USB on your PC. Follow the instructions supplied with the SMC-USB.

Apply the following settings to the COM port of your PC:

| | |
|---|---|
| Bits per second | 57,600 |
| Data bits | 8 |
| Parity | None |
| Stop bits | 1 |
| Flow control | Xon/Xoff |
| Terminator | $C_RL_F$ |

## 3.2 Communication to a Single SMC100CC/PP

Set the dip switches on the SMC100CC/PP to FIRST:




Connect the SMC100CC/PP to the RS-232 or to the USB port of your PC. Connect your stage to the SMC100CC/PP (MOTOR connector). Connect the power supply. The LED on the SMC100CC/PP turns RED.

## 3.3 Communication to Several SMC100CC/PP

When using several SMC100CC/PP controllers through the internal RS-485 communication link, you need to follow specific steps to be successful:

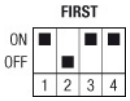
1. Apply individual addresses to each controller.
2. Connect all elements of the system together.
3. Configure each controller to drive the connected stage.

### 3.3.1 Controller Address Setting

The first thing to do is applying an individual address to each SMC100CC/PP controller.

The address of the FIRST controller connected through RS-232-C remains the address number 1. You don't need to do anything with this controller. For addressing the other controllers do the following:

Set the dip switches of ALL SMC100CC/PP to FIRST (see graphic below).




Connect ONE, and only one, SMC100CC/PP to the RS-232-C or to the USB port of your PC. It is not needed to connect any stage to the controller. Connect the power supply. The LED turns RED.

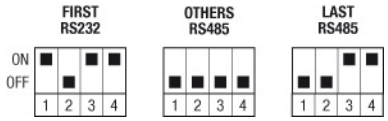
Set an address with the SMC100 applet GUI and select "Address" tab. It is recommended to note down the address of the controller somewhere. For example, use the stickers supplied with the SMC100CC/PP.

Now disconnect this controller from your PC and connect the next one instead. Select a new, not yet allocated address and press the "Set" button again. Proceed the same with all other controllers.

### 3.3.2 Building the System

When the addresses of all controllers are set, you can build your system.

Pull out all cables from all controllers. Set the dip switches of the controller with the address number 1 as FIRST. Set the dip switches of the other controllers, except one, as OTHERS, and set the dip switches of one controller as LAST. When you have only two controllers, one has to be set as FIRST (the one with the address number 1), and the other one as LAST. See below graphic for illustration.




Connect the SMC100CC/PP configured as FIRST to the RS-232-C port or to the USB port of your PC. Connect a RS-485 network cable to the RS-485 OUT of the FIRST controller and to the RS-485 IN of the next controller. Proceed the same with all other controllers. When done, you can check your system:

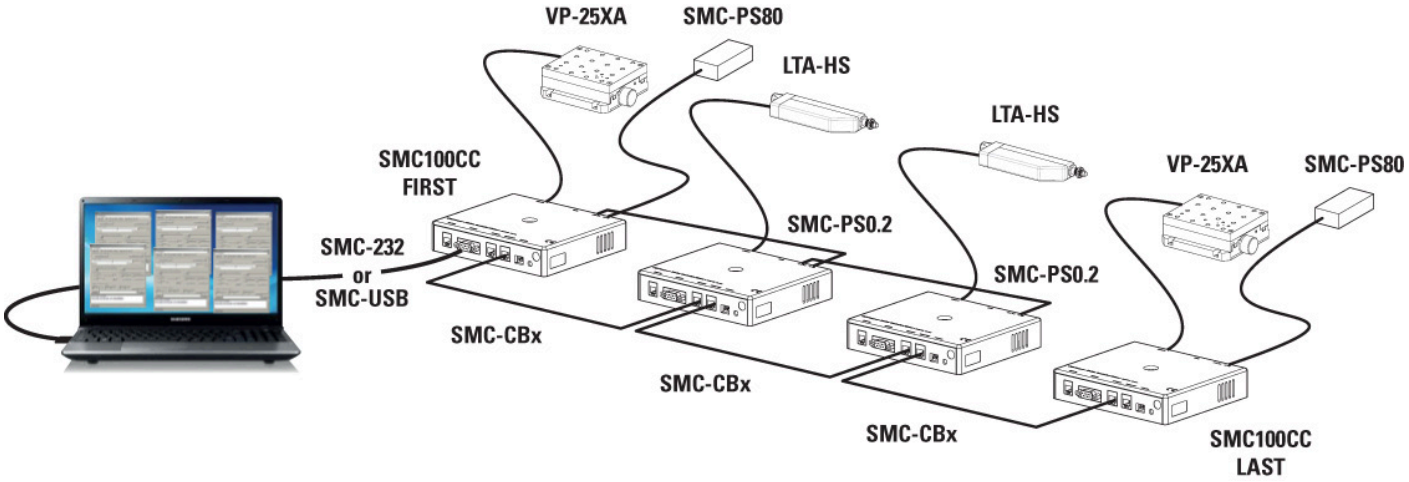
- The controller configured as FIRST should have the RS-232-C cable connected. It has the address number 1.
- All controllers configured as OTHERS should have one RS-485 network cable connected to the RS-485 IN and another one to the RS-485 OUT.
- The controller connected as LAST should have one RS-485 network cable connected to the RS-485 IN.

Connect your stages to the SMC100CC/PP's (MOTOR connector). Connect your SMC100CC/PP's to power.

The SMC100CC/PP allows chaining power from one SMC100CC/PP to another one using the SMC-PSC0.2 cable supplied with the controller. But the total power consumption of all stages connected to the same power supply should not exceed 80 W. The maximum power consumption of each Newport stage is listed in the Newport catalog and on the Newport web site. In case of questions, contact Newport.

An example: The maximum power consumption of a VP-25XA is 48 W. The maximum power consumption of an LTA-HS is 6 W. So it is possible to connect one VP-25XA and up to 5 LTA-HS to the same power supply. But it is not possible to connect two VP-25XA to the same power supply.

When done, your configuration should look as follow:




### 3.3.3 Configuring the Controller

Start the SMC100 Applet GUI and go to the "Parameters" tab.

When using the SMC100CC/PP with Newport ESP compatible stages (see label on the stage), press "Download parameters from SmartStage".

Start with the controller address 1. Press "Download parameters from SmartStage". Select the next available controller address and press "Download parameters from SmartStage" again. Proceed the same with all other controllers.

When done, your system is configured and ready to use.

**Using the SMC100CC/PP with non Newport ESP compatible stages or changing the default values**

When using the SMC100CC/PP with non Newport ESP compatible stages, you need to enter the stage parameters manually in the Parameters tab. In the "Parameters" page you can also change the configuration parameters stored in the controller. But it is not recommended doing this unless you are an experienced user. For further information about the meaning of the different parameters, please refer to the explanations at the corresponding two letter commands (see command names in brackets) in section 6.5.

# 4.0 Default Speed Setting Control for Newport Stepper Stages *(only available for SMC100PP controller)*

Due to some technical reasons, all Newport stepper stages will be set to be driven at reduced speed with the SMC100PP controller (Reduced speed = Nominal speed / 2.5).

In order to check which stages can be driven at reduced speed or full speed, please refer to the Newport web site (SMC100PP web page).

For example, an URSPP stage with a max speed of 40 °/s will be driven with a max speed of 16 °/s when controlled by the SMC100PP controller.

For stages than can be driven at full speed (please refer to the Newport web site to get the list), the defaut speed setting can be increased by the user to get the full nominal speed.

## 4.1 Irms Current Setting for SMC100PP Controller

The connection type of a stepper motor can be bipolar (full winding) or unipolar (half winding), but the SMC100PP controller always controls the stepper motor in the full winding control mode. So the Irms current in each case must be different each from other.

In the case of a unipolar motor, if the motor resistance (controlled in half winding) is R, so the same motor resistance controlled in full winding is 2R.

For the same power (and the same thermal dissipation) in all two cases, we must have:

$$R.I_{half}^2 = 2R.I_{full}^2 \quad (1)$$

Here: $I_{half}$ is the motor current in the case of half winding control (this is also Asmart: value found in the stage smart EPROM memory).

$I_{full}$ is the motor current in the case of full winding control.

From (1) we have:

$$I_{full} = I_{half} / \sqrt{2} \quad (2)$$

So in the case of a unipolar motor controlled in full winding mode (SMC100PP), the motor must not be controlled with the Asmart value, but Asmart / √2.

# 5.0 SMC100CC/PP with SMC-RC Keypad

The SMC-RC keypad allows basic use of the SMC100CC/PP controller without a computer. It features a 16 characters position display and four push buttons for configuration, jogging, homing, and enabling/disabling motors. It can be also used in parallel to a computer control.

If not already done, carefully unpack and visually inspect the SMC100CC/PP controller, the SMC-RC keypad, all stages and all accessories for any damage. Place all components on a flat and clean surface.

1. Connect the SMC-RC to the SMC100CC/PP (KEYPAD connector).
2. Connect your stage to the SMC100CC/PP (MOTOR connector).
3. Connect the SMC100CC/PP to the SMC-PS80 (DC IN connector).
4. Connect the SMC-PS80 to power.

During the initialization, the SMC100CC/PP controller checks if a SMC-RC keypad is connected. If so, it checks whether all buttons are open (not pressed). If not, an error message gets generated.

**NOTE**

**The SMC100CC/PP does not recognize an SMC-RC after the initialization. Also, disconnecting the SMC-RC from the controller and reconnecting without reinitializing the controller does not work.**

**To reinitialize the SMC100CC/PP controller, temporarily disconnect from power and reconnect again, or send the RS command (see section 6.5).**

When using the SMC100CC/PP for the first time with a Newport ESP compatible stage (see blue label on the product) a message **AUTOCONFIG ? YES** gets displayed for about 5 seconds. Press the Exec. button to configure the SMC100CC/PP to the connected stage. Once done, this message gets not displayed anymore during later initialization unless the SMC100CC/PP recognizes a different Newport ESP compatible stage than the one it is configured to. This message gets also not displayed if the controller is already configured correctly using the SMC100CC/PP software utility (see chapter 3.0).

After successful initialization, the controller is in the NOT REFERENCED state and the display displays **+0.00000 HOM** (for more details about the SMC100CC/PP states, please refer to section 6.1). Press the Exec. button to home the stage. The stage starts moving to its home position. When done, the display shows **+0.00000 JOG**. The digital value indicates the current position of the stage. The default units for Newport positioners are millimeters for linear stages and actuators, and degrees for rotation stages.

Pressing the Exec. button again gets the controller to the JOGGING state and the display changes to **+0.00000 DIS**. The jog buttons “<”, “<< >>”, and “>” are now enabled. Pressing the “<” (jog left) or “>” (Jog right) button starts a motion at slow velocity and with slow acceleration. Releasing the button stops the motion. These slow speed motion are ideal for precise adjustments. Pressing the “<” (jog left) or “>” (Jog right) button and the “<< >>” (high speed) simultaneously starts a high speed motion. These high speed motion are ideal for coarse adjustments. The jog speed and jog acceleration settings are as follow:

| | |
|---|---|
| High jog velocity: | Equal to the default velocity (see value set in the software utility or with the VA command). |
| High jog acceleration: | High jog velocity / 2s (means final velocity is reached after 2 seconds). |
| High jog deceleration: | Equal to the default acceleration (see value set in the software utility or with the AC command). |

| | |
|---|---|
| Low jog velocity: | Equal to the default velocity (see value set in the software utility or with the VA command) divided by 1000. |
| Low jog acceleration: | Low jog velocity / 2s (means final velocity is reached after 2 seconds). |
| Low jog deceleration: | Equal to the default acceleration (see value set in the software utility or with the AC command). |

**NOTE**

**Any jog motion always respects the software limits (see settings in the software utility or with the SL and SR commands). When approaching a software limit, the controller decelerates with the programmed acceleration even if the jog buttons are pressed.**

Pressing the Exec. button when the three most right letters are DIS, gets the controller to the DISABLE state. In DISABLE state the motor is not energized and the control loop is open (for DC version). But the encoder is still read and the current position gets updated. The DISABLE state can be used for instance for manual adjustments or to make sure that no energy goes to the motor. To go from DISABLE state to the JOGGING state, press the Exec. button again.

The buttons of the keypad can get disabled by the JD command.

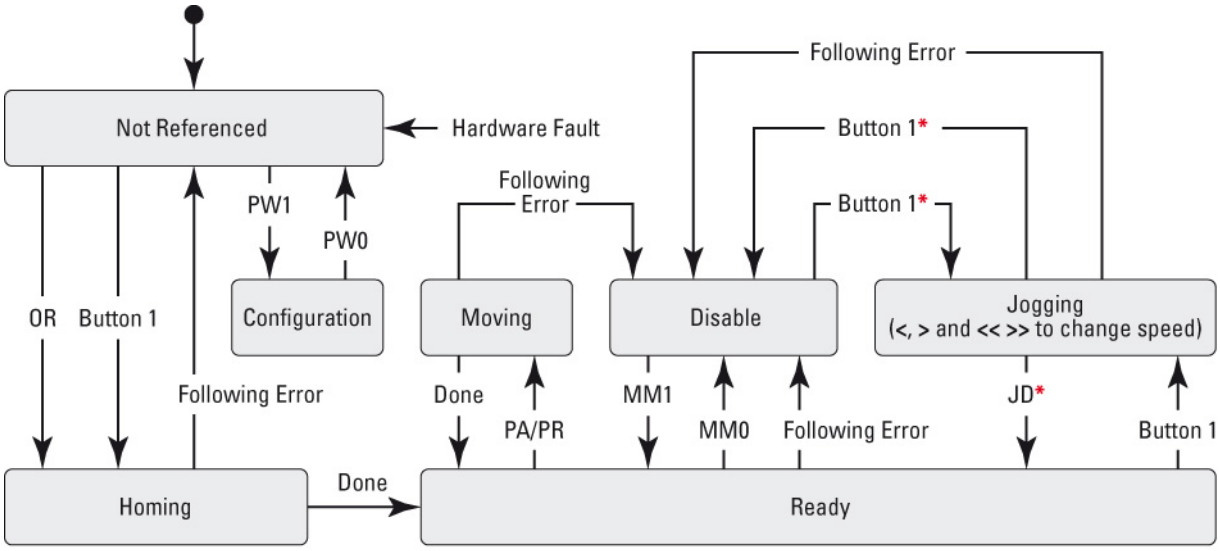
**NOTE**

**The keypad does not allow stopping any motion started from a computer (all buttons are disabled when the controller is in MOVING state). To take computer control when the controller is in JOGGING state the controller must first get to the READY state (change state from the software utility or by using the JD command).**

# 6.0 Programming

## 6.1 State Diagram

For a safe and consistent operation, the SCM100CC uses 7 different operation states: Not referenced, Configuration, Homing, Ready, Disable, Jogging and Moving. In each state, only specific commands are accepted by the SMC100CC/PP. Therefore, it is important to understand the state diagram below and which commands and actions cause transition between the different states. Also see section 6.5 for command/state information:




* No action, when jogging speed is different than zero, e.g. one of the keys "<", ">" or "<< >>" is pressed.

**End of Runs encountered in the following state:**

| | |
|---|---|
| NOT REFERENCED: | No action. |
| CONFIGURATION: | No action. |
| HOMING: | Only check at end of HOMING and then change to NOT REFERENCED state. |
| MOVING: | Abort motion and then change to NOT REFERENCED state. |
| READY: | Change to NOT REFERENCED state. |
| DISABLE: | Change to NOT REFERENCED state. |

**LED display:**

| | |
|---|---|
| NOT REFERENCED: | If everithing is OK then SOLID ORANGE. |
| NOT REFERENCED: | If hardware faults or wrong parameters then SOLID RED. |
| NOT REFERENCED: | If end of runs then SLOW BLINK ORANGE. |
| CONFIGURATION: | SLOW BLINK RED. |
| READY: | SOLID GREEN. |
| DISABLE: | SLOW BLINK GREEN. |
| HOMING: | FAST BLINK GREEN. |
| MOVING: | FAST BLINK GREEN. |
| JOGGING: | FAST BLINK GREEN. |

When connecting the SMC100CC/PP to power, the controller initializes (see section 6.2). When the initialization is successful, the controller gets to the NOT REFERENCED state. From the NOT REFERENCED state, the controller can go to the CONFIGURATION state with the PW1 command. In CONFIGURATION stage, the SMC100CC/PP allows changing all stage and motor configuration parameters like maximum motor current or travel limits. The PW0 command saves all changes to the controller's memory and returns the controller back to the NOT REFERNCED state.

To execute any move commands (PA, PR), the controller must be in READY state. To get from the NOT REFERENCED state to the READY state, the positioner must be homed first with the OR command. During homing (OR command execution), the controller is in HOMING state. When the homing is successful, the controller automatically gets to the READY state. The process for homing, and which signals are looked for during homing, can be defined with the HT command.

In READY state the motor is energized and the control loop is closed (when control loop state is closed, SC1). During a move execution (PA/PR), the controller is in MOVING state and gets automatically back to the READY state when the move is completed successfully. A following error during a move changes the controller to DISABLE state. Other errors, for instance a loss of the encoder signals, may change the controller to the NOT REFERENCED state.

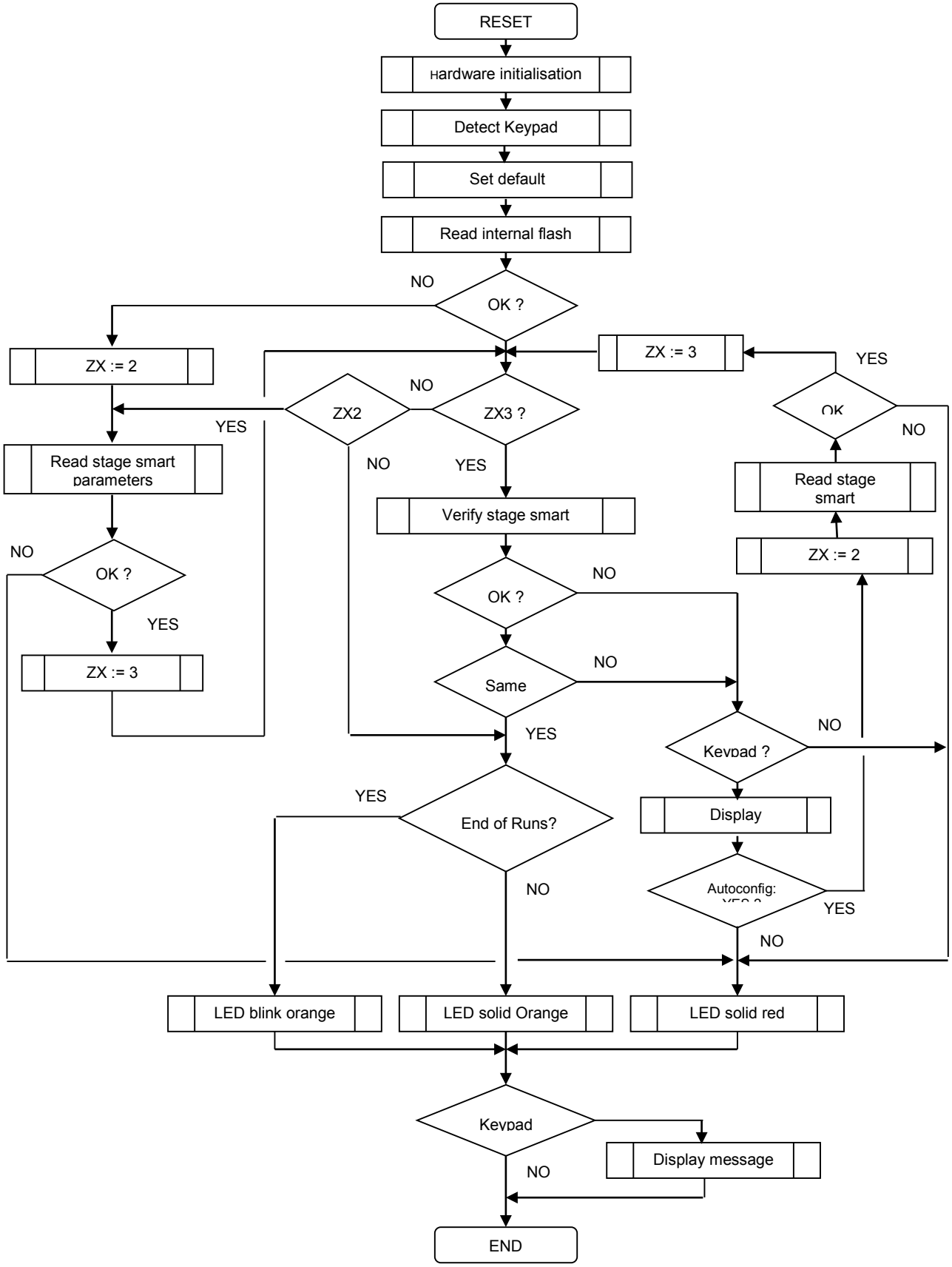
In DISABLE state the motor is not energized and the control loop is open (for DC version). But the encoder is still read and the current position gets updated (on the SMC100CC only). The DISABLE state can be used for instance for manual adjustments or to make sure that no energy goes to the motor. To go from READY state to DISABLE state and vice versa, use the MM command.

In JOGGING state the controller allows computer independent motion from the SMC-RC keypad. The controller can get to the JOGGING state ONLY by pressing the Exec. button on the SMC-RC when the controller is in the READY or in the DISABLE state. To get from JOGGING state to READY state use the JD command.

To get from READY state or DISABLE state back to the NOT REFERENCED state, for instance to make some further parameter change in CONFIGURATION state, you need to reboot the controller with the RS command.

## 6.2 Initialization

When connecting the SMC100CC/PP to power, the following initialization routine gets executed. The initialization lasts less than 5 s. For more information about system errors during initialization, refer to the TS command in section 6.5.



## 6.3 Command Syntax

The SMC100CC/PP is a command driven controller. The general format of a command is a two letter ASCII character preceded and followed by parameters specific to the command:

**Command format:**

| nn | AA | xx |
|---|---|---|

**nn** — Optional or required controller address.

**AA** — Command name.

**xx** — Optional or required value or "**?**" to query current value.

Both, upper and lower case characters are accepted. Depending on the command, it can have an optional or required prefix (**nn**) for the controller address and/or a suffix (**xx**) value or a "**?**".

**Blank spaces**

Blanks are allowed and ignored in any position, including inside a numerical value. The following two commands are equivalent, but the first example might be confusing and uses more memory:

2P A1.43 6

2PA1.436

**Decimal separator**

A dot (".") is used as decimal separator for all numerical values.

Command terminator

Commands are executed as the command terminator $C_RL_F$ (carriage-return line-feed, ASCII 13 and ASCII 10) is received. The controller will analyze the received string. If the command is valid and its parameters are in the specified range, it will be executed. Otherwise it will memorize an error.

After the execution of the command, all remaining characters in the input string, if any, will be ignored. In particular, it is not possible to concatenate several commands on a single string from the PC to the SMC100.

Each command will handle properly the memorization of related errors that can be accessed with the TE command. Please refer to the command set in section 6.5 for details.

## 6.4 Command Execution Time

The SMC100CC/PP controller interprets commands continuously as received. The typical execution time for a "tell position command" (nTP?) is about 10 ms for the first controller (controller address number 1) and about 16 ms for the other controllers. Here, command execution time means the time from sending the command until receive of the answer.

It is important to note that a move command, that may lasts for several seconds, will not suspend the controller from further command execution. So for an efficient process flow with many move commands it is recommended to use the PT command (get time for a relative move), and to query the controller status (TS command) or the current position (TP command) before any further motion command is sent. Alternative, the dedicated outputs "In Motion" and "Not Referenced" can be used for similar purposes. These will provide an even more timely accurate information of the controller state.

## 6.5 Command Set

This section describes the supported two-letter ASCII commands used to configure and operate the SMC100CC/PP. The general command format is:

**Command format:**

| nn | AA | xx |
|---|---|---|

**nn** — Optional or required controller address.

**AA** — Command name.

**xx** — Optional or required value or "**?**" to query current value.

Since multiple SMC100CC/PP may be chained through the internal RS-485 Bus, each controller uses a predetermined address (**nn**), and by decoding the address field of the incoming commands, it can determine if the command is intended for it. Some command though, can be passed without a controller address. In that case the command applies to all concerned controllers. For example: ST0 stops the motion on all controllers, 1ST0 stops the motion only on controller #1.

Most commands can be used to set a value (in that case the command name is followed by the value "**xx**") or to query the current value (in that case the command name is followed by a "**?**"). When querying a value, the controller responds with the command it received followed by the queried value. For example, a 1VA10 sets the velocity of the controller #1 to 10 units/second. A 1VA? sends the response 1VA10.

Not every command can be executed in all states of the SMC100CC/PP and some commands have different meaning in different states. It is therefore important to understand the state diagram of the controller, see section 6.1.

| | Not Ref. | Config. | Disable | Ready | Motion | Jogging | Description | SMC100CC/PP |
|---|---|---|---|---|---|---|---|---|
| **AC** | | ¶ | × | × | | | Set/Get acceleration | |
| **BA** | | ¶ | | | | | Set/Get backlash compensation | |
| **BH** | | ¶ | | | | | Set/Get hysteresis compensation | |
| **DV** | | ¶ | | | | | Set/Get driver voltage | Not for PP |
| **FD** | | ¶ | × | | | | Set/Get low pass filter for Kd | Not for PP |
| **FE** | | ¶ | × | | | | Set/Get following error limit | Not for PP |
| **FF** | | ¶ | × | | | | Set/Get friction compensation | Not for PP |
| **FR** | | ¶ | | | | | Set/Get stepper motor configuration | Not for CC |
| **HT** | | ¶ | | | | | Set/Get HOME search type | |
| **ID** | | ¶ | | | | | Set/Get stage identifier | |
| **JD** | | | | | | × | Leave JOGGING state | |
| **JM** | | ¶ | × | × | | | Enable/disable keypad | |
| **JR** | | ¶ | × | × | | | Set/Get jerk time | |
| **KD** | | ¶ | × | | | | Set/Get derivative gain | Not for PP |
| **KI** | | ¶ | × | | | | Set/Get integral gain | Not for PP |
| **KP** | | ¶ | × | | | | Set/Get proportional gain | Not for PP |
| **KV** | | ¶ | × | | | | Set/Get velocity feed forward | Not for PP |
| **MM** | | | × | × | | | Enter/Leave DISABLE state | |
| **OH** | | ¶ | | | | | Set/Get HOME search velocity | |
| **OR** | × | | | | | | Execute HOME search | |
| **OT** | | ¶ | | | | | Set/Get HOME search time-out | |
| **PA** | | | | × | | | Move absolute | |
| **PR** | | | | × | | | Move relative | |
| **PT** | | | × | × | × | | Get motion time for a relative move | |
| **PW** | × | × | | | | | Enter/Leave CONFIGURATION state | |
| **QI** | | ¶ | | | | | Set/Get motor's current limits | |
| **RA** | × | × | × | × | × | × | Get analog input value | |
| **RB** | × | × | × | × | × | × | Get TTL input value | |
| **RS** | × | | × | × | | | Reset controller | |
| **SA** | | ¶ | | | | | Set/Get controller's RS-485 address | |
| **SB** | | | × | × | × | × | Set/Get TTL output value | |
| **SC** | | ¶ | ¶ | | | | Set/Get control loop state | Not for PP |
| **SE** | | | | × | | | Configure/Execute simultaneous started move | |
| **SL** | | ¶ | × | × | | | Set/Get negative software limit | |
| **SR** | | ¶ | × | × | | | Set/Get positive software limit | |
| **ST** | | | × | × | × | | Stop motion | |
| **SU** | | ¶ | | | | | Set/Get encoder increment value | Not for PP |
| **TB** | × | × | × | × | × | × | Get command error string | |
| **TE** | × | × | × | × | × | | Get last command error | |
| **TH** | × | × | × | × | × | × | Get set-point position | |
| **TP** | × | × | × | × | × | × | Get current position | |
| **TS** | × | × | × | × | × | × | Get positioner error and controller state | |
| **VA** | | ¶ | × | × | | | Set/Get velocity | |
| **VB** | | ¶ | × | × | | | Set/Get base velocity | Not for CC |
| **VE** | × | × | × | × | × | × | Get controller revision information | |
| **ZT** | × | × | × | × | × | | Get all axis parameters | |
| **ZX** | | ¶ | | | | | Set/Get SmartStage configuration | |

| | |
|---|---|
| **Motion:** | Corresponds to HOMING and MOVING state (for details see state diagram, section 6.1). |
| ¶ | Changes configuration parameters. Those changes will be stored in the controller's memory with the PW1 command and remain available after switching off the controller. |
| × | Changes working parameters only. Those changes will get lost when switching off the controller. |
| × | Accepted command. |
| **Blank:** | Not accepted command (will return an error). |
| **Command:** | Command passed without preceding controller number applies to all controllers (e.g. MM0 disables all controllers). |
| **Not for PP:** | The controller will return an error indicating that the command is not allowed for SMC100PP version. |
| **Not for CC:** | The controller will return an error indicating that the command is not allowed for SMC100CC version. |

# AC — Set/Get acceleration

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | ▪ | ▪ | × | × |

**Syntax** **xxACnn** or **xxAC?**

**Parameters**

**Description**
**xx** [int] — Controller address.
**nn** [float] — Acceleration value.

**Range**
**xx** — **1** to **31**
**nn** — **> $10^{-6}$** and **< $10^{12}$**

**Units**
**xx** — None
**nn** — Preset units/$s^2$

**Defaults**
**xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.
**nn** Missing: Error C.
Out of range: Error C.

**Description**
In CONFIGURATION state, this command sets the maximum acceleration value which can than be saved in the controller's nonvolatile memory using the PW command. This is the maximum acceleration that can be applied to the mechanical system. It is also the default acceleration that will be used for all moves unless a lower value is set in DISABLE or READY state.

In DISABLE or READY state, this command sets the acceleration used for the following moves. Its value can be up to the programmed value in CONFIGURATION state. This value is not saved in the controller's memory and will be lost after reboot.

**Returns**
If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**
A — Unknown message code or floating point controller address.
B — Controller address not correct.
C — Parameter missing or out of range.
D — Execution not allowed.
H — Execution not allowed in NOT REFERENCED state.
L — Execution not allowed in HOMING state.
M — Execution not allowed in MOVING state.

**Rel. Commands**
**VA** — Set velocity.

**Example**
1AC500 | *Set controller #1 acceleration to 500 units/$s^2$.*
1AC? | *Controller returns 1AC500.*

# BA — Set/Get backlash compensation

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | × | × | × | × |

**Syntax** **xxBAnn** or **xxBA?**

**Parameters**

**Description**
**xx** [int] — Controller address.
**nn** [float] — Backlash value.

**Range**
**xx** — **1** to **31**
**nn** — **≥ 0** and **< 1E12**

**Units**
**xx** — None
**nn** — Preset units

**Defaults**
**xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.
**nn** Missing: Error C.
Out of range: Error C.

**Description** The BA command sets the backlash compensation value. This is the value that the controller moves the motor in addition to the commanded distance with any move that reverses the direction of motion without changing the current position value (TP command).

The BA command helps compensating for repeatable mechanical defects that appear when reversing the direction of motion, for instance mechanical play. The value 0 disables this function. This feature can be only used when the hysteresis compensation (BH) is disabled.

**Returns** If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**
A — Unknown message code or floating point controller address.
B — Controller address not correct.
C — Parameter missing or out of range.
D — Execution not allowed.
H — Execution not allowed in NOT REFERENCED state.
J — Execution not allowed in DISABLE state.
K — Execution not allowed in READY state.
L — Execution not allowed in HOMING state.
M — Execution not allowed in MOVING state.

**Rel. Commands** **BH** — Set hysteresis compensation.

**Example** 1BA0.005 | *Set controller #1 backlash compensation to 0.005 units.*

# BH — Set/Get hysteresis compensation

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | × | × | × | × |

**Syntax** **xxBHnn** or **xxBH?**

**Parameters**

**Description**
**xx** [int] — Controller address.
**nn** [float] — Hysteresis value.

**Range**
**xx** — **1** to **31**
**nn** — **≥ 0** and **< $10^{12}$**

**Units**
**xx** — None
**nn** — Preset units

**Defaults**
**xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.
**nn** Missing: Error C.
Out of range: Error C.

**Description** The BH command sets the hysteresis compensation value. When set to a value different than zero, the controller will issue for each move in the positive direction a move of the commanded distance plus the hysteresis compensation value, and then a second move of the hysteresis compensation value in the negative direction. This motion ensures that a final position gets always approached from the same direction and distance and helps compensating for non-repeatable mechanical defects like hysteresis or mechanical stiffness variations.

The value 0 disables this function. The BH command can not be used when the backlash compensation is enabled (BA command).

**Returns** If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**
A — Unknown message code or floating point controller address.
B — Controller address not correct.
C — Parameter missing or out of range.
D — Execution not allowed.
H — Execution not allowed in NOT REFERENCED state.
J — Execution not allowed in DISABLE state.
K — Execution not allowed in READY state.
L — Execution not allowed in HOMING state.
M — Execution not allowed in MOVING state.

**Rel. Commands** **BA** — Set backlash compensation.

**Example** 1BH0.015 | *Set controller #1 backlash compensation to 0.015 units.*

# DV — Set/Get driver voltage

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | × | × | × | × |

**Syntax** **xxDVnn** or **xxDV?**

**Parameters**

**Description**
**xx** [int] — Controller address.
**nn** [float] — Driver voltage value.

**Range**
**xx** — **1** to **31**
**nn** — ≥ **12** and ≤ **48**

**Units**
**xx** — None.
**nn** — Volts

**Defaults**
**xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.
**nn** Missing: Error C.
Out of range: Error C.

**Description** This command sets the max. output voltage of the driver to the motor.

**Returns** If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**
A — Unknown message code or floating point controller address.
B — Controller address not correct.
C — Parameter missing or out of range.
D — Execution not allowed.
H — Execution not allowed in NOT REFERENCED state.
J — Execution not allowed in DISABLE state.
K — Execution not allowed in READY state.
L — Execution not allowed in HOMING state.
M — Execution not allowed in MOVING state.

**Rel. Commands** **QI** — Set current limit.

**Example** 1DV48 | *Set controller #1 maximum output voltage to 48 V.*

# FD — Set/Get low pass filter cut off frequency for Kd

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | ▪ | × | × | × |

**Syntax** **xxFDnn** or **xxFD?**

**Parameters**

**Description**
**xx** [int] — Controller address.
**nn** [float] — Cut off frequency value.

**Range**
**xx** — **1** to **31**
**nn** — **> $10^{-6}$** and **< 2000**

**Units**
**xx** — None.
**nn** — Hertz

**Defaults**
**xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.
**nn** Missing: Error C.
Out of range: Error C.

**Description**
In CONFIGURATION state, this command sets the value for the low pass filter cut-off frequency which can than be saved in the controller's nonvolatile memory using the PW command. It is also the default value that will be used unless a different value is set in DISABLE state.

In DISABLE state, this command allows setting a new working parameter for the low pass filter cut-off frequency. This value is not saved in the controller's memory and will be lost after reboot.

**Returns**
If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**
A — Unknown message code or floating point controller address.
B — Controller address not correct.
C — Parameter missing or out of range.
D — Execution not allowed.
H — Execution not allowed in NOT REFERENCED state.
K — Execution not allowed in READY state.
L — Execution not allowed in HOMING state.
M — Execution not allowed in MOVING state.
W — Command not allowed for SMC100PP version.

**Rel. Commands**
**SC** — Set closed loop state.

**Example**
1FD1500 | *Set controller #1 Kd cut-off frequency to 1500 Hz.*

# FE — Set/Get following error limit

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | ▪ | × | × | × |

**Syntax** **xxFEnn** or **xxFE?**

**Parameters**

**Description**
**xx** [int] — Controller address.
**nn** [float] — Following error limit value.

**Range**
**xx** — **1** to **31**
**nn** — **> $10^{-6}$** and **< $10^{12}$**

**Units**
**xx** — None.
**nn** — Preset units.

**Defaults**
**xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.
**nn** Missing: Error C.
Out of range: Error C.

**Description**
In CONFIGURATION state, this command sets the value for the maximum allowed following error which can than be saved in the controller's nonvolatile memory using the PW command. It is also the default value that will be used for the closed-loop control unless a different value is set in DISABLE state.

The following error is the most important parameter to control motion. It is the difference between the set point (or theoretical) position and the current (or encoder) position. When the current following error exceeds the maximum allowed value, a following error is issued and the controller is set to DISABLE state.

In DISABLE state, this command allows setting a new working parameter for the maximum allowed following error. This value is not saved in the controller's memory and will be lost after reboot.

**Returns**
If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**
A — Unknown message code or floating point controller address.
B — Controller address not correct.
C — Parameter missing or out of range.
D — Execution not allowed.
H — Execution not allowed in NOT REFERENCED state.
K — Execution not allowed in READY state.
L — Execution not allowed in HOMING state.
M — Execution not allowed in MOVING state.
W — Command not allowed for SMC100PP version.

**Rel. Commands**
**SC** — Set closed loop state.

**Example**
1FE0.015 | *Set controller #1 following error limit to 0.015 units.*

# FF — Set/Get friction compensation

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | ▪ | × | × | × |

**Syntax** **xxFFnn** or **xxFF?**

**Parameters**

**Description**
**xx** [int] — Controller address.
**nn** [float] — Friction compensation value.

**Range**
**xx** — **1** to **31**
**nn** — **≥ 0** and **< DV**

**Units**
**xx** — None.
**nn** — Volt * second/preset units.

**Defaults**
**xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.
**nn** Missing: Error C.
Out of range: Error C.

**Description**
In CONFIGURATION state, this command sets the value for the friction compensation which can than be saved in the controller's nonvolatile memory using the PW command. It is also the default value that will be used for any move unless a different value is set in DISABLE state.

The FF command helps minimizing the following error with systems that have significant friction. The value for the friction compensation is the voltage that gets added to the output voltage whenever the set point (or theoretical) velocity is different from zero. The sign of this voltage is the same as the sign of the set point velocity.

In DISABLE state, this command allows setting a new working parameter for the friction compensation. This value is not saved in the controller's memory and will be lost after reboot.

**Returns**
If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**
A — Unknown message code or floating point controller address.
B — Controller address not correct.
C — Parameter missing or out of range.
D — Execution not allowed.
H — Execution not allowed in NOT REFERENCED state.
K — Execution not allowed in READY state.
L — Execution not allowed in HOMING state.
M — Execution not allowed in MOVING state.
W — Command not allowed for SMC100PP version.

**Rel. Commands**
**SC** — Set closed loop state.

**Example**
1FF0.15 | *Set controller #1 friction compensation to 0.15 V * s/units.*

# FR — Set/Get stepper motor configuration

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | × | × | × | × |

**Syntax** **xxFRSnn**, **xxFRM?** or **xxFRS?**

**Parameters**

**Description**

**xx** [int] — Axis number.

**Mmm** [int]— Micro-step factor.

**Snn** [float] — Full step value.

**Range**

**xx** — **1** to **31**

**mm** — **> 0** and **≤ 2000**

**nn** — **> 1E$^{-6}$** and **< 1E$^{12}$**

**Units**

**xx** — None.

**Mmm** — None.

**Snn** — None.

**Defaults**

**xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**mm** Missing: Error C.

Out of range: Error C.

**nn** Missing: Error C.

Out of range: Error C.

**Description** FRM: this command sets the micro-step per full step factor.

FRS : this command sets the motion distance per motor's full step.

**Returns** If the sign "**?**" takes place of mm or **nn**, this command returns the current programmed value.

**Errors**

A — Unknown message code or floating point controller address.

B — Controller address not correct.

C — Parameter missing or out of range.

D — Execution not allowed.

H — Execution not allowed in NOT REFERENCED state.

J — Execution not allowed in DISABLE state.

K — Execution not allowed in READY state.

L — Execution not allowed in HOMING state.

M — Execution not allowed in MOVING state.

X — Command not allowed for SMC100CC version.

**Rel. Commands** VB — Set base velocity.

**Example** 1FRS0.02 | *Set controller #1 full step value to 0.02 units.*

# HT — Set/Get HOME search type

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | × | × | × | × |

**Syntax** **xxHTnn** or **xxHT?**

**Parameters**

**Description**

**xx** [int] — Controller address.

**nn** [int] — Home type value.

**Range**

**xx** — **1** to **31**

**nn** — **0** use MZ switch and encoder Index.
**1** use current position as HOME.
**2** use MZ switch only.
**3** use EoR- switch and encoder Index.
**4** use EoR- switch only.

**Units**

**xx** — None.

**nn** — None.

**Defaults**

**xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.

**nn** Missing: Error C.
Out of range: Error C.

**Description** This command sets the type of HOME search used with the OR command.

**Returns** If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**

A — Unknown message code or floating point controller address.
B — Controller address not correct.
C — Parameter missing or out of range.
D — Execution not allowed.
H — Execution not allowed in NOT REFERENCED state.
J — Execution not allowed in DISABLE state.
K — Execution not allowed in READY state.
L — Execution not allowed in HOMING state.
M — Execution not allowed in MOVING state.

**Rel. Commands** **OR** — Execute HOME search.

**Example** 1HT0 | *Set controller #1 HOME sequence to use MZ and encoder index.*

# ID — Set/Get stage identifier

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | × | × | × | × |

**Syntax** **xxIDnn** or **xxID?**

**Parameters**

**Description**
**xx** [int] — Controller address.
**nn** [float] — Stage model number.

**Range**
**xx** — **1** to **31**
**nn** — **1** to **31** ASCII characters.

**Units**
**xx** — None
**nn** — None

**Defaults**
**xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.
**nn** Missing: Error C.
Out of range: Error C.

Description
The ID? command return the stage identifier. When used with Newport ESP compatible stages (see blue label on the product), this is the identical to the Newport product name. In CONFIGURATION mode, this command allows changing the stage identifier. However, customer should never do this when the ESP stage configuration is enabled (ZX3).

**Returns**
If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**
A — Unknown message code or floating point controller address.
B — Controller address not correct.
C — Parameter missing or out of range.
D — Execution not allowed.
H — Execution not allowed in NOT REFERENCED state.
J — Execution not allowed in DISABLE state.
K — Execution not allowed in READY state.
L — Execution not allowed in HOMING state.
M — Execution not allowed in MOVING state.

**Rel. Commands**
**ZX** — Set SmartStage configuration.

**Example**
1ID? | *Get stage identifier for controller #1.*
| *Controller returns URS100CC.*

# JD — Leave JOGGING state

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | × | × | × | × | ▪ |

**Syntax** **xxJD**

**Parameters**

**Description** **xx** [int] — Controller address.

**Range** **xx** — **1** to **31**

**Units** **xx** — None

**Defaults** **xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.

**Description** In JOGGING STATE, when no jog buttons are pressed and the stage velocity is 0 the xxJD command sets the controller's state to READY.

**Errors**

| | | |
|---|---|---|
| A | — | Unknown message code or floating point controller address. |
| B | — | Controller address not correct. |
| D | — | Execution not allowed. |
| H | — | Execution not allowed in NOT REFERENCED state. |
| I | — | Execution not allowed in CONFIGURATION state. |
| J | — | Execution not allowed in DISABLE state. |
| K | — | Execution not allowed in READY state. |
| L | — | Execution not allowed in HOMING state. |
| M | — | Execution not allowed in MOVING state. |

**Rel. Commands** **JM** — Enable/Disable keypad.

**Example** 1JD | *Controller #1 leaves jogging state.*

# JM — Enable/Disable keypad

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | ▪ | ▪ | × | × |

**Syntax** **xxJMnn** or **xxJM?**

**Parameters**

**Description**
**xx** [int] — Controller address.
**nn** [float] — Jog state.

**Range**
**xx** — **1** to **31**
**nn** — **0** or **1**

**Units**
**xx** — None
**nn** — None

**Defaults**
**xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.
**nn** Missing: Error B.
Out of range: Error A.

**Description** The JM1 command enables the SMC-RC keypad buttons (default setting). The JM0 command disables the SMC-RC keypad buttons.

Sending the JM command when the controller is in DISABLE or READY state only temporarily applies the setting. With the next boot of the controller the default setting will get applied again. Whereas sending the JM command when the controller is in CONFIGURATION state saves the setting in the controller's non-volatile memory).

**Returns** If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**
A — Unknown message code or floating point controller address.
B — Controller address not correct.
D — Execution not allowed.
H — Execution not allowed in NOT REFERENCED state.
L — Execution not allowed in HOMING state.
M — Execution not allowed in MOVING state.

**Rel. Commands** **JD** — Leave JOGGING state.

**Example** 1JM1 | *Enable keypad for controller #1.*

# JR — Set/Get jerk time

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | ▪ | ▪ | × | × |

**Syntax** **xxJRnn** or **xxJR?**

**Parameters**

**Description**

**xx** [int] — Controller address.

**nn** [float] — Jerk time value.

**Range**

**xx** — 1 to 31

**nn** — **> 0.001** and **< $10^{12}$**

**Units**

**xx** — None.

**nn** — Seconds.

**Defaults**

**xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**nn** Missing: Error C.

Out of range: Error C.

**Description**

In CONFIGURATION state, this command sets the value for the maximum jerk time which can than be saved in the controller's nonvolatile memory using the PW command. It is also the default value that will be used unless a different value is set in DISABLE or READY state.

Jerk is the derivative of acceleration. The jerk time defines the time to reach the needed acceleration. A longer jerk time reduces stress to the mechanics and smoothes motion.

In DISABLE or READY state, this command allows setting a new working parameter for the maximum jerk time. This value is not saved in the controller's memory and will be lost after reboot.

**Returns**

If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**

A — Unknown message code or floating point controller address.

B — Controller address not correct.

C — Parameter missing or out of range.

D — Execution impossible (axis in movement).

H — Execution not allowed in NOT REFERENCED state.

L — Execution not allowed in HOMING state.

M — Execution not allowed in MOVING state.

**Rel. Commands**

**AC** — Set positioner acceleration.

**Example**

1JR0.05 | *Set controller #1 jerk time to 0.05 seconds.*

# KD — Set/Get derivative gain

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | ▪ | × | × | × |

**Syntax** **xxKDnn** or **xxKD?**

**Parameters**

**Description**

**xx** [int] — Controller address.

**nn** [float] — Derivative gain value.

**Range**

**xx** — **1** to **31**

**nn** — **≥ 0** and **< $10^{12}$**

**Units**

**xx** — None.

**nn** — Volt * second/preset unit.

**Defaults**

**xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**nn** Missing: Error C.

Out of range: Error C.

**Description**

In CONFIGURATION state, this command sets the derivative gain of the PID control loop which can than be saved in the controller's nonvolatile memory using the PW command. It is also the default value that will be used unless a different value is set in DISABLE state.

In DISABLE state, this command allows setting a new working parameter for the derivative gain. This value is not saved in the controller's memory and will be lost after reboot.

**Returns**

If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**

A — Unknown message code or floating point controller address.

B — Controller address not correct.

C — Parameter missing or out of range.

D — Execution not allowed.

H — Execution not allowed in NOT REFERENCED state.

K — Execution not allowed in READY state.

L — Execution not allowed in HOMING state.

M — Execution not allowed in MOVING state.

W — Command not allowed for SMC100PP version.

**Rel. Commands**

**SC** — Set closed loop state.

**KI** — Set integral gain.

**KP** — Set proportional gain.

**KV** — Set velocity feed forward.

**Example** 1KD0.015 | *Set controller #1 derivative gain to 0.015.*

# KI — Set/Get integral gain

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | ▪ | × | × | × |

**Syntax** **xxKInn** or **xxKI?**

**Parameters**

**Description**

**xx** [int] — Controller address.

**nn** [float] — Integral gain value.

**Range**

**xx** — **1** to **31**

**nn** — **≥ 0** and **< $10^{12}$**

**Units**

**xx** — None.

**nn** — Volt * preset unit/second.

**Defaults**

**xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**nn** Missing: Error C.

Out of range: Error C.

**Description**

In CONFIGURATION state, this command sets the integral gain of the PID control loop which can than be saved in the controller's nonvolatile memory using the PW command. It is also the default value that will be used unless a different value is set in DISABLE state.

In DISABLE state, this command allows setting a new working parameter for the derivative gain. This value is not saved in the controller's memory and will be lost after reboot.

**Returns**

If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**

A — Unknown message code or floating point controller address.

B — Controller address not correct.

C — Parameter missing or out of range.

D — Execution not allowed.

H — Execution not allowed in NOT REFERENCED state.

K — Execution not allowed in READY state.

L — Execution not allowed in HOMING state.

M — Execution not allowed in MOVING state.

W — Command not allowed for SMC100PP version.

**Rel. Commands**

**SC** — Set closed loop state.

**KD** — Set derivative gain.

**KP** — Set proportional gain.

**KV** — Set velocity feed forward.

**Example** 1KI0.015 | *Set controller #1 integral gain to 0.015.*

# KP — Set/Get proportional gain

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | ▪ | × | × | × |

**Syntax** **xxKPnn** or **xxKP?**

**Parameters**

**Description**

**xx** [int] — Controller address.

**nn** [float] — Proportional gain value.

**Range**

**xx** — **1** to **31**

**nn** — **≥ 0** and **< $10^{12}$**

**Units**

xx — None.

**nn** — Volt/preset unit

**Defaults**

**xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**nn** Missing: Error C.

Out of range: Error C.

**Description**

In CONFIGURATION state, this command sets the proportional gain of the PID control loop which can than be saved in the controller's nonvolatile memory using the PW command. It is also the default value that will be used unless a different value is set in DISABLE state.

In DISABLE state, this command allows setting a new working parameter for the derivative gain. This value is not saved in the controller's memory and will be lost after reboot.

**Returns**

If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**

A — Unknown message code or floating point controller address.

B — Controller address not correct.

C — Parameter missing or out of range.

D — Execution not allowed.

H — Execution not allowed in NOT REFERENCED state.

K — Execution not allowed in READY state.

L — Execution not allowed in HOMING state.

M — Execution not allowed in MOVING state.

W — Command not allowed for SMC100PP version.

**Rel. Commands**

**SC** — Set closed loop state.

**KD** — Set derivative gain.

**KI** — Set integral gain.

**KV** — Set velocity feed forward.

**Example** 1KP0.015 | *Set controller #1 proportional gain to 0.015.*

# KV — Set/Get velocity feed forward

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | ▪ | × | × | × |

**Syntax** **xxKVnn** or **xxKV?**

**Parameters**

**Description**

**xx** [int] — Controller address.

**nn** [float] — Velocity feed forward value.

**Range**

**xx** — **1** to **31**

**nn** — **≥ 0** and **< $10^{12}$**

**Units**

**xx** — None.

**nn** — Volt * second/preset unit

**Defaults**

**xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**nn** Missing: Error C.

Out of range: Error C.

**Description**

In CONFIGURATION state, this command sets the velocity feed forward of the PID control loop which can than be saved in the controller's nonvolatile memory using the PW command. It is also the default value that will be used unless a different value is set in DISABLE state.

In DISABLE state, this command allows setting a new working parameter for the derivative gain. This value is not saved in the controller's memory and will be lost after reboot.

**Returns**

If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**

A — Unknown message code or floating point controller address.

B — Controller address not correct.

C — Parameter missing or out of range.

D — Execution not allowed.

H — Execution not allowed in NOT REFERENCED state.

K — Execution not allowed in READY state.

L — Execution not allowed in HOMING state.

M — Execution not allowed in MOVING state.

W — Command not allowed for SMC100PP version.

**Rel. Commands**

**SC** — Set closed loop state.

**KD** — Set derivative gain.

**KI** — Set integral gain.

**KP** — Set proportional gain.

**Example**

1KV0.015 | *Set controller #1 velocity feed forward to 0.015.*

# MM — Enter/Leave DISABLE state

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | × | ▪ | ▪ | × | × |

**Syntax** **xxMMnn** or **xxMM?**

**Parameters**

**Description**
**xx** [int] — Controller address.
**nn** [float] — Velocity feed forward value.

**Range**
**xx** — **0** to **31**
**nn** — **0** changes state from READY to DISABLE.
**1** changes state from DISABLE to READY.

**Units**
**xx** — None.
**nn** — None.

**Defaults**
**xx** Missing: Change to 0.
Out of range: Error B.
Floating point: Error A.
**nn** Missing: Error C.
Out of range: Error C.

**Description** When the MM command is sent without preceding controller number or the controller number is 0, the MM command gets executed on all controllers.

MM0 changes the controller's state from READY to DISABLE. In DISABLE state the control loop is open and the motor is not energized . The encoder, though, is still read and the current position gets updated (on the SMC100CC only).

MM1 changes the controller's state from DISABLE to READY. The controller's set point position is set equal to its current position and the control loop gets closed (depending on the closed-loop state). The residual following error gets cleared from the buffer and the motor gets energized.

**Returns** If the sign "**?**" takes place of **nn**, this command returns the current state.

**Errors**
A — Unknown message code or floating point controller address.
B — Controller address not correct.
C — Parameter missing or out of range.
D — Execution not allowed.
H — Execution not allowed in NOT REFERENCED state.
I — Execution not allowed in CONFIGURATION state.
L — Execution not allowed in HOMING state.
M — Execution not allowed in MOVING state.

**Rel. Commands** **PW** — Enter/leave CONFIGURATION state.

**Example**
MM0 | *All controllers go to DISABLE state.*

# OH — Set/Get HOME search velocity

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | × | × | × | × |

**Syntax** **xxOHnn** or **xxOH?**

**Parameters**

**Description**
**xx** [int] — Controller address.
**nn** [float] — HOME high velocity.

**Range**
**xx** — **1** to **31**
**nn** — **> $10^{-6}$** and **< $10^{12}$**

**Units**
**xx** — None.
**nn** — Preset units/s.

**Defaults**
**xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.
**nn** Missing: Error C.
Out of range: Error C.

**Description** This command sets the maximum velocity used by the controller for the HOME search.

**Returns** If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**
A — Unknown message code or floating point controller address.
B — Controller address not correct.
C — Parameter missing or out of range.
D — Execution not allowed.
H — Execution not allowed in NOT REFERENCED state.
J — Execution not allowed in DISABLE state.
K — Execution not allowed in READY state.
L — Execution not allowed in HOMING state.
M — Execution not allowed in MOVING state.

**Rel. Commands**
**OR** — Execute HOME search.
**OT** — Set HOME search time-out.

**Example** 1OH50 | *Set controller #1 HOME search velocity to 50 units/s.*

# OR — Execute HOME search

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | ▪ | × | × | × | × | × |

**Syntax** **xxOR**

**Parameters**

**Description** **xx** [int] — Controller address.

**Range** **xx** — **1** to **31**

**Units** **xx** — None.

**Defaults** **xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.
**nn** Missing: Error C.
Out of range: Error C.

**Description** This command starts the execution of the HOME search as defined by the HT command.

When in NOT REFERENCED state, for instance after system start, any positioner must first get homed with the OR command before further motion commands can get executed.

The OR command gets accepted only in NOT REFERENCED state and only with no present hardware errors, except for end-of-run maybe. Refer to the TS command to get more information on the possible hardware errors.

**Errors**

| | | |
|---|---|---|
| A | — | Unknown message code or floating point controller address. |
| B | — | Controller address not correct. |
| C | — | Parameter missing or out of range. |
| D | — | Execution not allowed. |
| E | — | home sequence already started. |
| I | — | Execution not allowed in CONFIGURATION state. |
| J | — | Execution not allowed in DISABLE state. |
| K | — | Execution not allowed in READY state. |
| L | — | Execution not allowed in HOMING state. |
| M | — | Execution not allowed in MOVING state. |

**Rel. Commands**

| | | |
|---|---|---|
| **HT** | — | Set HOME search type. |
| **OH** | — | Set HOME search velocity. |
| **OT** | — | Set HOME search time-out. |

**Example** 1OR | *Execute HOME search with controller #1.*

# OT — Set/Get HOME search time-out

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | × | × | × | × |

**Syntax** **xxOTnn** or **xxOT?**

**Parameters**

**Description**

**xx** [int] — Controller address.

**nn** [float] — HOME time-out.

**Range**

**xx** — **1** to **31**

**nn** — **> 1** and **< $10^3$**

**Units**

**xx** — None.

**nn** — Seconds

**Defaults**

**xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**nn** Missing: Error C.

Out of range: Error C.

**Description** This command sets the time-out value for the HOME search. When the HOME search does not finish successfully before this time elapses, the HOME search will be aborted and an error gets recorded.

**Returns** If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**

A — Unknown message code or floating point controller address.

B — Controller address not correct.

C — Parameter missing or out of range.

D — Execution not allowed.

H — Execution not allowed in NOT REFERENCED state.

J — Execution not allowed in DISABLE state.

K — Execution not allowed in READY state.

L — Execution not allowed in HOMING state.

M — Execution not allowed in MOVING state.

**Rel. Commands**

**HT** — Set HOME search type.

**OH** — Set HOME search velocity.

**OR** — Execute HOME search.

**Example** 1OT2.2 | *Set controller #1 HOME time-out to 2.2 seconds.*

# PA — Move absolute

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | × | × | ▪ | × | × |

**Syntax** **xxPAnn** or **xxPA?**

**Parameters**

**Description**

**xx** [int] — Controller address.

**nn** [float] — New target position.

**Range**

**xx** — **1** to **31**

**nn** — **> SL** and **< SR**

**Units**

**xx** — None.

**nn** — Preset units.

**Defaults**

**xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**nn** Missing: Error C.

Out of range: Error C.

**Description**

The PA command initiates an absolute move. When received, the positioner will move, with the predefined acceleration and velocity, to the new target position specified by **nn**.

The PA command gets only accepted in READY state, AND when the new target position is higher or equal to the negative software limit (SL), AND lower or equal to the positive software limit (SR).

To avoid any mismatch, the controller always rounds the new target position to the closest encoder position.

**Returns**

If the sign "**?**" takes place of **nn**, this command returns the target position value.

**Errors**

A — Unknown message code or floating point controller address.

B — Controller address not correct.

C — Parameter missing or out of range.

D — Execution not allowed.

G — Target position out of limits.

H — Execution not allowed in NOT REFERENCED state.

I — Execution not allowed in CONFIGURATION state.

J — Execution not allowed in DISABLE state.

**Rel. Commands**

**PR** — Move relative.

**TH** — Get set-point position.

**TP** — Get current position.

**SU** — Set encoder increment value.

**Example**

1PA2.2 | *Move positioner on controller #1 to absolute position 2.2 units.*

# PR — Move relative

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | × | × | ▪ | × | × |

**Syntax** **xxPRnn** or **xxPR?**

**Parameters**

**Description**

**xx** [int] — Controller address.

**nn** [float] — Displacement.

**Range**

**xx** — **1** to **31**

**nn** — **> SL** and **< SR**

**Units**

**xx** — None.

**nn** — Preset units.

**Defaults**

**xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**nn** Missing: Error C.

Out of range: Error C.

**Description**

The PR command initiates a relative move. When received, the positioner will move, with the predefined acceleration and velocity, to a new target position **nn** units away from the current target position.

The PR command gets only accepted in READY state, AND when the distance of the positioner to the end of runs is larger than the commanded displacement.

To avoid any mismatch, the controller always rounds the new target position to the closest encoder position.

**Returns**

If the sign "**?**" takes place of **nn**, this command returns the target position value.

**Errors**

A — Unknown message code or floating point controller address.

B — Controller address not correct.

C — Parameter missing or out of range.

D — Execution not allowed.

G — Displacement out of limits.

H — Execution not allowed in NOT REFERENCED state.

I — Execution not allowed in CONFIGURATION state.

J — Execution not allowed in DISABLE state.

**Rel. Commands**

**PA** — Move absolute.

**TH** — Get set-point position.

**TP** — Get current position.

**SU** — Set encoder increment value.

**Example**

1PR2.2 | *Move positioner on controller #1 to a new position 2.2 units away from the current target position.*

# PT — Get motion time for a relative move

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | × | ▪ | ▪ | ▪ | × |

**Syntax** **xxPTnn**

**Parameters**

**Description**
**xx** [int] — Controller address.
**nn** [float] — Displacement.

**Range**
**xx** — **1** to **31**
**nn** — > $\mathbf{10^{-6}}$ and < $\mathbf{10^{12}}$

**Units**
**xx** — None.
**nn** — Preset units.

**Defaults**
**xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.
**nn** Missing: Error C.
Out of range: Error C.

**Description**
The PT commands helps evaluating move times for an efficient program flow.

When receiving the PT command, the controller returns the time, in seconds, necessary to execute a relative move of the displacement **nn** with the current working parameters (velocity, acceleration, etc.). The controller does not execute any motion.

**Errors**
A — Unknown message code or floating point controller address.
B — Controller address not correct.
C — Parameter missing or out of range.
D — Execution not allowed.
H — Execution not allowed in NOT REFERENCED state.
I — Execution not allowed in CONFIGURATION state.

**Rel. Commands**
**PA** — Move absolute.
**PR** — Move relative.
**TH** — Get set-point position.
**TP** — Get current position.
**SU** — Set encoder increment value.

**Example**
1PT2.2 | *Get time to move positioner on controller #1 by 2.2 units.*
| *Controller returns: 1PT0.25, means 0.25 seconds.*

# PW — Enter/Leave CONFIGURATION state

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | ▪ | ▪ | × | × | × | × |

**Syntax** **xxPWnn** or **xxPW?**

**Parameters**

**Description**

**xx** [int] — Controller address.

**nn** [float] — Velocity feed forward value.

**Range**

**xx** — **1** to **31**

**nn** — **1**: Go from NOT REFERENCED state to CONFIGURATION state.

**0**: Go from CONFIGURATION state to NOT REFERENCED state.

**Units**

**xx** — None.

**nn** — None.

**Defaults**

**xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**nn** Missing: Error C.

Out of range: Error C.

**Description**

PW1 changes the controller's state from NOT REFERENCED to CONFIGURATION. In Configuration state all parameter settings are saved in the controller's memory and remain available after switching off the controller. In addition, some settings are only possible in CONFIGURATION state (e.g. set drive voltage, set Backlash compensation, etc.).

PW0 checks all stage parameters, and if they are acceptable, saves them in the flash memory of the controller. After that, it changes the controller's state from CONFIGURATION to NOT REFERENCED.

The execution of a PW0 command may take up to 10 seconds. During that time the controller will not respond to any other command.

**Returns**

If the sign "**?**" takes place of **nn**, this command returns the current state.

**Errors**

A — Unknown message code or floating point controller address.

B — Controller address not correct.

C — Parameter missing or out of range.

D — Execution not allowed.

J — Execution not allowed in DISABLE state.

K — Execution not allowed in READY state.

L — Execution not allowed in HOMING state.

M — Execution not allowed in MOVING state.

**Rel. Commands**

**MM** — Enter/Leave DISABLE state.

**Example**

1PW1 | *Changes controller #1 to CONFIGURATION state.*

# QI — Set/Get motor's current limits

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | × | × | × | × |

**Syntax** **xxQILnn**, **xxQIRnn**, **xxQITnn**, **xxQIL?**, **xxQIR?** or **xxQIT?**

**Parameters**

**Description**

**xx** [int] — Controller address.

**Lmm** [float]— Motor's peak current limit.

**Rnn** [float]— Motor's rms current limit.

**Tpp** [float]— Motor's rms current averaging time.

**Range**

**xx** — **1** to **31**

**mm** — ≥ **0.05** and ≤ **3.0**

**nn** — ≥ **0.05** and ≤ **1.5** and ≤ **mm**

**pp** — > **0.01** and ≤ **100**

**Units**

**xx** — None.

**mm** — Amperes.

**nn** — Amperes.

**pp** — Seconds.

**Defaults**

**xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**mm** Missing: Error C.

**nn** Missing: Error C.

**pp** Missing: Error C.

Out of range: Error C.

**Description**

**QIL**: Sets the controller's maximum or peak output current limit to the motor. When the controller detects a higher current than the peak current limit, it will generate a hardware error and a fault will be recorded.

**QIR**: Sets the controller's rms output current limit to the motor. The rms current limit must be lower than the peak current limit. When the controller's output current exceeds the rms current limit, it will generate a hardware error and a fault will be recorded.

**QIT**: Sets the controller's averaging period for rms current calculation. In general, the QIT command defines for how long time the actual motor current is allowed to exceed the rms output current limit.

**Returns** If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**

A — Unknown message code or floating point controller address.

B — Controller address not correct.

C — Parameter missing or out of range.

D — Execution not allowed.

H — Execution not allowed in NOT REFERENCED state.

J — Execution not allowed in DISABLE state.

K — Execution not allowed in READY state.

L — Execution not allowed in HOMING state.

M — Execution not allowed in MOVING state.

**Rel. Commands** DV — Set driver input voltage.

**Example**

1QIL0.75 | *Set controller #1 current limit to 0.75 A.*

1QIR0.25 | *Set controller #1 rms current limit to 0.25 A.*

1QIT2.5 | *Set controller #1 rms averaging period to 2.5 s.*

# RA — Get analog input value

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | . | . | . | . | . | . |

**Syntax** **xxRA**

**Parameters**

**Description** **xx** [int] — Controller address.

**Range** **xx** — **1** to **31**

**Units** **xx** — None.

**Defaults** **xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.

**Description** The RA command returns the value of the ±10 volts analog input. The converter is a ±7 bits analog to digital converter with ±0.15 volts of maximum offset and 5% full scale linearity. The resolution is 0.078125 volts.

**Errors**
A — Unknown message code or floating point controller address.
B — Controller address not correct.
D — Execution not allowed.
H — Execution not allowed in NOT REFERENCED state.
I — Execution not allowed in CONFIGURATION state.

**Rel. Commands** **SB** — Get TTL inputs.

**Example**
1RA | *Get controller axis #1 analog input.*
| *Controller returns: 1RA7.8125, means 7.8125 V.*

# RB — Get TTL input value

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | . | . | . | . | . | . |

**Syntax** **xxRB**

**Parameters**

**Description** **xx** [int] — Controller address.

**Range** xx — **1** to **31**

**Units** xx — None.

**Defaults** **xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**Description** The RB command returns the value of the TTL inputs. The returned decimal number represents the binary word made of all 4 inputs, where bit 0 is input 1, bit 1 is input 2, bit 2 is input 3, and bit 3 is input 4.

The TTL input value is 1 when the corresponding voltage on the pin is larger than 2.4 volts, and it is 0 when the corresponding voltage is below 0.8 volt. When the voltage is between these two values, the result is unreliable and can be 1 or 0.

**Errors**

| | | |
|---|---|---|
| A | — | Unknown message code or floating point controller address. |
| B | — | Controller address not correct. |
| D | — | Execution not allowed. |
| H | — | Execution not allowed in NOT REFERENCED state. |
| I | — | Execution not allowed in CONFIGURATION state. |

**Rel. Commands** **RA** — Get analog input value.

**Example**

| | | |
|---|---|---|
| 1RB | \| | *Get TTL input value for controller #1.* |
| | \| | *Controller returns: 1RB5, means input 0 and 2 are high, all others are low.* |

# RS — Reset controller

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | ▪ | ▪ | ▪ | ▪ | ▪ | × |

**Syntax** **xxRS**

**Parameters**

**Description** **xx** [int] — Controller address.

**Range** **xx** — **1** to **31**

**Units** xx — None.

**Defaults** **xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.

**Description** The RS command issues a hardware reset of the controller, equivalent to a power-up.

To go from DISABLE or READY state to CONFIGURATION state, it is also needed to first reset the controller with the RS command, and then to change the controller's state with the PW1 command from NOT REFERENCED to CONFIGURATION.

**Errors**

| | | |
|---|---|---|
| A | — | Unknown message code or floating point controller address. |
| B | — | Controller address not correct. |
| D | — | Execution not allowed. |
| H | — | Execution not allowed in NOT REFERENCED state. |
| I | — | Execution not allowed in CONFIGURATION state. |
| L | — | Execution not allowed in HOMING state. |
| M | — | Execution not allowed in MOVING state. |

**Example** 1RS | *Reset controller #1.*

# SA — Set/Get controller's RS-485 address

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | × | × | × | |
| | × | | | | | |

**Syntax** **xxSAnn** or **xxSA?**

**Parameters**

**Description**

**xx** [int] — Axis number.

**nn** [int] — Controller's axis number.

**Range**

**xx** — **1**

**nn** — **2** to **31**

**Units**

**xx** — None.

**nn** — None.

**Defaults**

**xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**nn** Missing: Error C.

Out of range: Error C.

**Description**

The SA command sets the controller's RS-485 address. This address is ONLY used when the controller is configured for RS-485 communication.

The SA command can only be sent to a controller configured for RS-232-C communication. In this configuration, the controller's address is 1. Only one controller can be configured for RS-232-C communication.

Newport recommends using the supplied utility software for all controller configurations. The SA command is of practical use only when not using this software.

**Returns**

If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**

A — Unknown message code or floating point controller address.

B — Controller address not correct.

C — Parameter missing or out of range.

D — Execution not allowed.

H — Execution not allowed in NOT REFERENCED state.

J — Execution not allowed in DISABLE state.

K — Execution not allowed in READY state.

L — Execution not allowed in HOMING state.

M — Execution not allowed in MOVING state.

**Example**

1SA3 | *Set controller's RS-485 address to 3.*

# SB — Set/Get TTL output value

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | × | ▪ | ▪ | ▪ | ▪ |

**Syntax** **xxSBnn** or **xxSB?**

**Parameters**

**Description**

**xx** [int] — Controller address.

**nn** [int] — TTL output value.

**Range**

**xx** — **1** to **31**

**nn** — **0** to **15**

**Units**

**xx** — None.

**nn** — None.

**Defaults**

**xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**nn** Missing: Error C.

Out of range: Error C.

**Description**

The SB command sets the value of the TTL outputs. The decimal number **nn** represents thereby the binary word made of all 4 outputs, where bit 0 is output 1, bit 1 is output 2, bit 2 is output 3, and bit 3 is output 4.

A 1 closes the open collector output transistor of the output. A 0 blocks the open collector output transistor of the output.

**Returns**

If the sign "**?**" takes place of **nn**, this command returns the current TTL outputs value.

**Errors**

A — Unknown message code or floating point controller address.

B — Controller address not correct.

C — Parameter missing or out of range.

D — Execution not allowed.

H — Execution not allowed in NOT REFERENCED state.

I — Execution not allowed in CONFIGURATION state.

**Rel. Commands**

**RB** — Get TTL input value.

**Example**

1SB3 | *Close controller #1 TTL outputs 1 & 2 and open outputs 3 & 4.*

# SC — Set/Get control loop state

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | × | × | × | × |

**Syntax** **xxSCnn** or **xxSC?**

**Parameters**

**Description**

**xx** [int] — Controller address.

**nn** [int] — Closed loop state.

**Range**

**xx** — **1** to **31**

**nn** — **1**: CLOSED loop control.

**0**: OPEN loop control.

**Units**

**xx** — None.

**nn** — None.

**Defaults**

**xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**nn** Missing: Error C.

Out of range: Error C.

**Description** SC1 sets the controller to CLOSED loop control. This is the default.

SC0 sets the controller to OPEN loop control. Open loop control might be useful for defining stage parameters like friction compensation or velocity feed forward.

**Returns** If the sign "**?**" takes place of **nn**, this command returns the current state.

**Errors**

A — Unknown message code or floating point controller address

B — Controller address not correct.

C — Parameter missing or out of range.

D — Execution not allowed.

H — Execution not allowed in NOT REFERENCED state.

J — Execution not allowed in DISABLE state.

K — Execution not allowed in READY state.

L — Execution not allowed in HOMING state.

M — Execution not allowed in MOVING state.

W — Command not allowed for SMC100PP version.

**Rel. Commands**

**KD** — Set derivative gain.

**KI** — Set integral gain.

**KP** — Set proportional gain.

**KV** — Set velocity feed forward.

**Example** 1SC1 | *Set controller #1 to closed loop control.*

# SE — Configure/Execute simultaneous started move

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | × | × | ▪ | × | × |

**Syntax** **xxSEnn**, **xxSE?** or **SE**

**Parameters**

**Description**

**xx** [int] — Controller address.

**nn** [float] — New target position.

**Range**

**xx** — **0** to **31**

**nn** — **> SL** and **< SR**

**Units**

**xx** — None.

**nn** — Preset units.

**Defaults**

**xx** Missing: Change to 0.

Out of range: Error B.

Floating point: Error A.

**nn** Missing: Error C.

Out of range: Error C.

**Description**

The SE command allows starting a move on different controllers at the same time.

The command xxSEnn sets a new target position for the controller **nn**. But different than the PA command, the move does not get executed immediately, but only after receipt of an SE command without preceding controller number and without following position value. When receiving the SE command, all controllers start a move to their new target position.

The xxSEnn command gets only accepted in READY state, AND when the new target position is higher or equal to the negative software limit (SL), AND lower or equal to the positive software limit (SR). To avoid any mismatch, the controller always rounds the new target position to the closest encoder position.

The SE command should not be confused with a synchronized move. With a synchronized move, all positioners start their motion simultaneously and have velocities, accelerations and jerk times which are limited to a rate which make all positioners start and complete their moves at the same time. The emphasis here is that they all start AND stop at the same time. The SE command starts a move on all controllers at the same time, but each positioner moves with its individually defined velocity and acceleration. So naturally, the different positioners don't complete their motion at the same time.

**Returns**

If the sign "**?**" takes place of **nn**, this command returns the target position value set by the SE command, which is not necessarily the same as the target position set by the PA command.

**Errors**

A — Unknown message code or floating point controller address.

B — Controller address not correct.

C — Parameter missing or out of range.

D — Execution not allowed.

H — Execution not allowed in NOT REFERENCED state.

I — Execution not allowed in CONFIGURATION state.

J — Execution not allowed in DISABLE state.

L — Execution not allowed in HOMING state.

M — Execution not allowed in MOVING state.

| | | | |
|---|---|---|---|
| **Rel. Commands** | **PR** | — | Move relative. |
| | **TH** | — | Get set-point position. |
| | **TP** | — | Get current position. |
| | **SU** | — | Set encoder increment value. |
| **Example** | 1SE2.2 | \| | *Prepare controller #1 to move to absolute position 2.2 units.* |
| | 2SE3.3 | \| | *Prepare controller #2 to move to absolute position 3.3 units.* |
| | SE | \| | *All controllers start their programmed move, if any.* |

# SL — Set/Get negative software limit

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | ▪ | ▪ | × | × |

**Syntax** **xxSLnn** or **xxSL?**

**Parameters**

**Description**

**xx** [int] — Controller address.

**nn** [float] — Negative software limit.

**Range**

**xx** — **1** to **31**

**nn** — **> -10$^{12}$** and **≤ 0**

**Units**

**xx** — None.

**nn** — Preset units.

**Defaults**

**xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**nn** Missing: Error C.

Out of range: Error C.

**Description**

In CONFIGURATION state, this command sets the negative software limit which can than be saved in the controller's nonvolatile memory using the PW command. It is also the default value that will be used unless a different value is set in DISABLE or READY state.

In DISABLE or READY state, this command allows setting a new working parameter for the negative software limit. It must be lower or equal to the set-point position. This value is not saved in the controller's memory and will be lost after reboot.

The software limits are useful to limit the travel range of a positioner. There is no possibility to disable software limits. For an almost infinite motion, for instance with a rotation stage, set the lowest possible value, which is: -2147000000 * "encoder increment value" (see SU command). For instance if the encoder increment value is 0,0005, this limit is -1073500.

**Returns**

If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**

A — Unknown message code or floating point controller address.

B — Controller address not correct.

C — Parameter missing or out of range.

D — Execution not allowed.

H — Execution not allowed in NOT REFERENCED state.

L — Execution not allowed in HOMING state.

M — Execution not allowed in MOVING state.

**Rel. Commands** **SR** — Set positive software limit.

**Example** 1SL-100 | *Set controller #1 negative software limit to –100 units.*

# SR — Set/Get positive software limit

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | ▪ | ▪ | × | × |

**Syntax** **xxSRnn** or **xxSR?**

**Parameters**

**Description**
**xx** [int] — Controller address.
**nn** [float] — Positive software limit.

**Range**
**xx** — **1** to **31**
**nn** — **≥ 0** and **< $10^{12}$**

**Units**
**xx** — None.
**nn** — Preset units.

**Defaults**
**xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.
**nn** Missing: Error C.
Out of range: Error C.

**Description**
In CONFIGURATION state, this command sets the positive software limit which can than be saved in the controller's nonvolatile memory using the PW command. It is also the default value that will be used unless a different value is set in DISABLE or READY state.

In DISABLE or READY state, this command allows setting a new working parameter for the positive software limit. It must be larger or equal to the set-point position. This value is not saved in the controller's memory and will be lost after reboot.

The software limits are useful to limit the travel range of a positioner. There is no possibility to disable software limits. For an almost infinite motion, for instance with a rotation stage, set the largest possible value, which is: 2147000000 * "encoder increment value" (see SU command). For instance if the encoder increment value is 0,0005, this limit is 1073500.

**Returns**
If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**
A — Unknown message code or floating point controller address.
B — Controller address not correct.
C — Parameter missing or out of range.
D — Execution not allowed.
H — Execution not allowed in NOT REFERENCED state.
L — Execution not allowed in HOMING state.
M — Execution not allowed in MOVING state.

**Rel. Commands** **SL** — Set negative software limit.

**Example** 1SR100 | *Set controller #1 positive software positive to 100 units.*

# ST — Stop motion

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | × | ▪ | ▪ | ▪ | × |

**Syntax** **[xx]ST**

**Parameters**

**Description** **xx** [int] — Controller address.

**Range** **xx** — **0** to **31**

**Units** **xx** — None.

**Defaults** **xx** Missing: Change to 0.

Out of range: Error B.

Floating point: Error A.

**Description** The ST command is a safety feature. It stops a move in progress by decelerating the positioner immediately with the acceleration defined by the AC command until it stops.

The xxST command with preceding controller address stops a move in progress on controller xx. The ST command without preceding controller address stops the moves on ALL controllers.

**Errors**

| | | |
|---|---|---|
| A | — | Unknown message code or floating point controller address. |
| B | — | Controller address not correct. |
| D | — | Execution not allowed. |
| H | — | Execution not allowed in NOT REFERENCED state. |
| I | — | Execution not allowed in CONFIGURATION state. |

**Example**

| | |
|---|---|
| ST \| | *Stop moves on all controllers.* |

# SU — Set/Get encoder increment value

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | × | × | × | × |

**Syntax** **xxSUnn** or **xxSU?**

**Parameters**

**Description**
**xx** [int] — Controller address.
**nn** [float] — Equivalent units to one encoder count.

**Range**
**xx** — **1** to **31**
**nn** — **> $10^{-6}$** and **< $10^{12}$**

**Units**
**xx** — None.
**nn** — Units.

**Defaults**
**xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.
**nn** Missing: Error C.
Out of range: Error C.

**Description** The SU command sets the value for one encoder count. It defines also the system of units for all other parameters like travel limits, velocities, accelerations, etc. Therefore, it is the first parameter to be defined for any positioner.

Example: For a positioner with an encoder resolution of 1 µm, the command xxSU0.001 sets 1 encoder count = 1 µm = 0.001 unit or 1 unit = 1 mm.

**Returns** If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**
A — Unknown message code or floating point controller address.
B — Controller address not correct.
C — Parameter missing or out of range.
D — Execution not allowed.
H — Execution not allowed in NOT REFERENCED state.
J — Execution not allowed in DISABLE state.
K — Execution not allowed in READY state.
L — Execution not allowed in HOMING state.
M — Execution not allowed in MOVING state.
W — Command not allowed for SMC100PP version.

**Example** 1SU7.5e-6 | *Set controller #1 encoder increment to 7.5 * $10^{-6}$ units.*

# TB — Get command error string

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | . | . | . | . | . | . |

**Syntax** **xxTBnn**

**Parameters**

**Description** **xx** [int] — Controller address.

**Range** **xx** — **1** to **31**

**nn** [char] — Error code (refer to TE command).

**Units** **xx** — None.

**Defaults** **xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**nn** Missing: Returns explanation of current error.

Out of range: Error C.

**Description** The TB command returns a string that explains the meaning of the error code **nn** (see TE command for complete list).

**Errors** A — Unknown message code or floating point controller address.

B — Controller address not correct.

C — Parameter missing or out of range.

D — Execution not allowed.

**Rel. Commands** **TE** — Get error code.

**Example** 1TB@ | *Get explanation to error code @.*

| *Controller returns: 1TB@ No error, @ means no error.*

# TE — Get last command error

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | ▪ | ▪ | ▪ | ▪ | ▪ | × |

**Syntax** **xxTE**

**Parameters**

**Description** **xx** [int] — Controller address.

**Range** **xx** — **1** to **31**

**Units** **xx** — None.

**Defaults** **xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**Description** The TE command returns the currently memorized error. When a command is not executable, it memorizes an error. This error can be read with the TE command. After the execution of a TE command, the error buffer gets erased and another TE command will return @, means no error. When a new command error is generated before the previous command error is read, the new command error will overwrite the current memorized error.

For a safe program flow it is recommended to always query the command error after each command execution.

**Errors**

A — Unknown message code or floating point controller address.

B — Controller address not correct.

D — Execution not allowed.

**Rel. Commands** **TB** — Get error string.

**Example**

1TE | *Get last error memorized on controller #1.*

| *Controller returns: 1TE@, means no error.*

List of errors and corresponding strings (see TB command):

@ — No error.

A — Unknown message code or floating point controller address.

B — Controller address not correct.

C — Parameter missing or out of range.

D — Command not allowed.

E — Home sequence already started.

F — ESP stage name unknown.

G — Displacement out of limits.

H — Command not allowed in NOT REFERENCED state.

I — Command not allowed in CONFIGURATION state.

J — Command not allowed in DISABLE state.

K — Command not allowed in READY state.

L — Command not allowed in HOMING state.

M — Command not allowed in MOVING state.

N — Current position out of software limit.

S — Communication Time Out.

U — Error during EEPROM access.

V — Error during command execution.

W — Command not allowed for PP version.

X — Command not allowed for CC version.

# TH — Get set-point position

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | . | . | . | . | . | . |

**Syntax** **xxTH**

**Parameters**

**Description** **xx** [int] — Controller address.

**Range** **xx** — **1** to **31**

**Units** **xx** — None.

**Defaults** **xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.

**Description** The TH command returns the value of the set-point or theoretical position. This is the position where the positioner should be. In MOVING state, the set-point position changes according to the calculation of the motion profiler. In READY state, the set-point position is equal to the target position.

**Errors**
A — Unknown message code or floating point controller address.
B — Controller address not correct.
D — Execution not allowed.
H — Execution not allowed in NOT REFERENCED state.
I — Execution not allowed in CONFIGURATION state.

**Rel. Commands** TP — Get current position.

**Example**
1TH | *Get set-point position of controller #1.*
| *Controller returns: 1TH0, set-point position = 0 units.*

# TP — Get current position

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | . | . | . | . | . | . |

**Syntax** **xxTP**

**Parameters**

**Description** **xx** [int] — Controller address.

**Range** **xx** — **1** to **31**

**Units** **xx** — None.

**Defaults** **xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**Description** The TP command returns the value of the current position. This is the position where the positioner actually is according to his encoder value. In MOVING state, this value always changes. In READY state, this value should be equal or very close to the set-point and target position.

Together with the TS command, the TP command helps evaluating whether a motion is completed.

**Errors**

| | | |
|---|---|---|
| A | — | Unknown message code or floating point controller address. |
| B | — | Controller address not correct. |
| D | — | Execution not allowed |
| H | — | Execution not allowed in NOT REFERENCED state. |
| I | — | Execution not allowed in CONFIGURATION state. |

**Rel. Commands** **TH** — Get set-point position.

**Example**

| | | |
|---|---|---|
| 1TP | \| | *Get current position of controller #1.* |
| | \| | *Controller returns: 1TP0, actual position = 0 units.* |

# TS — Get positioner error and controller state

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | . | . | . | . | . | . |

**Syntax** **xxTS**

**Parameters**

**Description** **xx** [int] — Controller address.

**Range** **xx** — **1** to **31**

**Units** **xx** — None.

**nn** — None.

**Defaults** **xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**Description** The TS command returns the positioner error and the current controller state.

**Returns** The TS command returns six characters (1TSabcdef). The first 4 characters (abcd) represent the positioner error in Hexadecimal. The last two characters (ef) represent the controller state.

**Error code (abcd):** Convert each hexadecimal to a binary:

| F | E | D | C | B | A | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1111 | 1110 | 1101 | 1100 | 1011 | 1010 | 1001 | 1000 | 0111 | 0110 | 0101 | 0100 | 0011 | 0010 | 0001 | 0000 |

F

Each bit represents one possible error:

| A | | | | B | | | | C | | | | D | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| • Not used | • Not used | • Not used | • Not used | • Not used | • Not used | • 80 W output power exceeded | • DC voltage too low | • Wrong ESP stage | • Homing time out | • Following error | • Short circuit detection | • RMS current limit | • Peak current limit | • Positive end of run | • Negative end of run |

Examples:

- Error map 0000 = No errors
- Error map 0013 = Short circuit detection, Positive end of run, negative end of run
- Error map 004C = Homing time out, RMS current limit, Peak current limit

**Controller states (ef):**

- **0A**: NOT REFERENCED from reset.
- **0B**: NOT REFERENCED from HOMING.
- **0C**: NOT REFERENCED from CONFIGURATION.
- **0D**: NOT REFERENCED from DISABLE.
- **0E**: NOT REFERENCED from READY.
- **0F**: NOT REFERENCED from MOVING.
- **10**: NOT REFERENCED ESP stage error.
- **11**: NOT REFERENCED from JOGGING.
- **14**: CONFIGURATION.
- **1E**: HOMING commanded from RS-232-C.
- **1F**: HOMING commanded by SMC-RC.
- **28**: MOVING.
- **32**: READY from HOMING.
- **33**: READY from MOVING.
- **34**: READY from DISABLE.
- **35**: READY from JOGGING.
- **3C**: DISABLE from READY.
- **3D**: DISABLE from MOVING.
- **3E**: DISABLE from JOGGING.
- **46**: JOGGING from READY.
- **47**: JOGGING from DISABLE.

**NOTES**

**THE ERROR BUFFER GETS UPDATED PERIODICALLY, APPROX. EVERY 1 MS.**

**THE TS COMMAND READS THE ERROR BUFFER AND CLEARS THE ERROR BUFFER AT THE SAME TIME (SAME AS FOR COMMANDS TE, TB). SO WHEN LAUNCHING THE TS COMMAND, IT IS IMPORTANT TO PROCESS THE TS FEEDBACK ACCORDINGLY.**

**THE ERROR "WRONG ESP STAGE" GETS ONLY DETECTED DURING THE BOOTING OF THE CONTROLLER. WHEN READ THE ERROR IS CLEARED.**

**With no errors in the error buffer the color of the LED will change from red to either green or orange depending on the controller state.**

| | | | |
|---|---|---|---|
| **Errors** | A | — | Unknown message code or floating point controller address. |
| | B | — | Controller address not correct. |
| **Rel. Commands** | **TE** | — | Get last error. |
| **Example** | 1TS | \| | *Get error and state of controller #1.* |
| | | \| | *Controller returns: 1TS00000A, no errors and NOT REFERENCED from reset.* |

# VA — Set/Get velocity

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | ▪ | ▪ | × | × |

**Syntax** **xxVAnn** or **xxVA?**

**Parameters**

**Description**
**xx** [int] — Controller address.
**nn** [float] — Velocity value.

**Range**
**xx** — **1** to **31**
**nn** — > $\mathbf{10^{-6}}$ and < $\mathbf{10^{12}}$

**Units**
**xx** — None.
**nn** — Preset units/s.

**Defaults**
**xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.
**nn** Missing: Error C.
Out of range: Error C.

**Description** In CONFIGURATION state, this command sets the maximum velocity value which can than be saved in the controller's nonvolatile memory using the PW command. This is the maximum velocity that can be applied to the mechanical system. It is also the default velocity that will be used for all moves unless a lower value is set in DISABLE or READY state.

In DISABLE or READY state, this command sets the velocity used for the following moves. Its value can be up to the programmed value in CONFIGURATION state. This value is not saved in the controller's memory and will be lost after reboot.

**Returns** If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**
A — Unknown message code or floating point controller address.
B — Controller address not correct.
C — Parameter missing or out of range.
D — Execution not allowed.
H — Execution not allowed in NOT REFERENCED state.
L — Execution not allowed in HOMING state.
M — Execution not allowed in MOVING state.

**Rel. Commands** **AC** — Set positioner acceleration.

**Example** 1VA50 | *Set controller #1 velocity to 50 units/s.*

# VB — Set/Get base velocity

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | ▪ | ▪ | × | × |

**Syntax** **xxVBnn** or **xxVB?**

**Parameters**

**Description**
**xx** [ int ] — Axis number.
**nn** [int] — Base velocity.

**Range**
**xx** — **1** to **31**
**nn** — **≤ 0** and **≥ value fixed by VA command**.

**Units**
**xx** — None.
**nn** — Units.

**Defaults**
**xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.
**nn** Missing: Error C.
Out of range: Error C.

**Description** This command sets the profile generator base velocity.

**Returns** If the sign "**?**" takes place of **nn**, this command returns the current programmed value.

**Errors**
A — Unknown message code or floating point controller address.
B — Controller address not correct.
C — Parameter missing or out of range.
D — Execution not allowed.
H — Execution not allowed in NOT REFERENCED state.
L — Execution not allowed in HOMING state.
M — Execution not allowed in MOVING state.
X — Command not allowed for SMC100CC version.

**Rel. Commands** **VA** — Set velocity.

EXAMPLE 1VB0.1 | *Set axis #1 base velocity to 0.1 units/s.*

# VE — Get controller revision information

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | . | . | . | . | . | . |

**Syntax** **xxVE**

**Parameters**

**Description** **xx** [int] — Controller address.

**nn** [string] — Action.

**Range** **xx** — **1** to **31**

**Units** **xx** — None.

**Defaults** **xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**Description** This command returns the controller's revision information.

**Errors** A — Unknown message code or floating point controller address.

B — Controller address not correct.

**Rel. Commands** **TP** — Get current position.

**Example** 1VE | *Get controller #1 revision information.*

| *Controller returns 1VE SMC - Controller-driver version 1.00r.*

# ZT — Get all configuration parameters

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | ▪ | ▪ | ▪ | ▪ | ▪ | × |

**Syntax** **xxZT**

**Parameters**

**Description** **xx** [int] — Controller address.

**Range** **xx** — **1** to **31**

**Units** **xx** — None.

**Defaults** **xx** Missing: Error B.
Out of range: Error B.
Floating point: Error A.

**Description** The ZT command returns the list of all current configuration parameters.

The ZT command allows a quick review of all current stage parameter and simplifies the configuration of non Newport stages, for instance by using Hyper Terminal file transfer.

**Errors** A — Unknown message code or floating point controller address
B — Controller address not correct

**Rel. Commands** **TE** — Get error code.

**Example**

```
1ZT     | Get controller #1 configuration data.
1PW1
1AC320.000000
1BA0.000000
…
1VA80.000000
1ZX3
1PW1
```

# ZX — Set/Get ESP stage configuration

| Usage | Not Ref. | Config. | Disable | Ready | Motion | Jogging |
|---|---|---|---|---|---|---|
| | × | ▪ | × | × | × | × |

**Syntax** **xxZXnn** or **xxZX?**

**Parameters**

**Description** **xx** [int] — Controller address.

**Range** **xx** — **1** to **31**

**nn** — **1** disable ESP stage check.

**2** update ESP stage information.

**3** enable ESP stage check.

**Units** **xx** — None.

**nn** — None.

**Defaults** **xx** Missing: Error B.

Out of range: Error B.

Floating point: Error A.

**nn** Missing: Error C.

Out of range: Error C.

**Description** The ZX command allows loading ESP stage data to the controller's flash memory and enables/disables ESP stage check during power-up. ESP refers to Newport stages with an EEPROM (called ESP chip), that contains all stage information like motor type, travel limits, maximum velocity, maximum acceleration, etc.

The command ZX2 reads the parameters from the ESP stage and saves them to the controller's flash memory. When using the SMC100CC/PP controller with Newport ESP compatible stages this is the fastest way of doing the stage configuration. When not using the Newport supplied utility software, just send the ZX2 command, and you're done.

The command ZX3 enables the ESP stage check. When enabled, the controller checks at each power-up whether the connected stage is the same as the one recorded in the controller flash memory. If not, it memorizes an error. The ESP stage check is recommended with all Newport ESP compatible stages.

The command ZX1 disables the ESP stage check. When disabled, the controller will not check the connected stage and the stage reference is set to UNKNOWN.

**Returns** If the sign "**?**" takes place of **nn**, this command returns the current stage reference.

**Errors**

| | | |
|---|---|---|
| A | — | Unknown message code or floating point controller address. |
| B | — | Controller address not correct. |
| C | — | Parameter missing or out of range. |
| D | — | Execution not allowed. |
| H | — | Execution not allowed in NOT REFERENCED state. |
| J | — | Execution not allowed in DISABLE state. |
| K | — | Execution not allowed in READY state. |
| L | — | Execution not allowed in HOMING state. |
| M | — | Execution not allowed in MOVING state. |

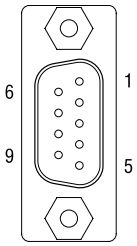
**Example** 1ZX? | *Controller returns: 1ZXURS100CC, means URS100CC stage.*

# 7.0 Connector Pinout

## 7.1 DC IN and DC OUT (Female Ø 2.1 x Ø 5.5 x 11 mm)

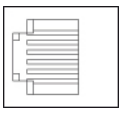
| Pin # | Description |
|---|---|
| Center | +48 VDC |
| Outer | GND |

## 7.2 RS-232-C (Male Sub-D9)



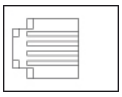

| Pin # | Description |
|---|---|
| **1** | Shorted together with 4 and 6 |
| **2** | TX |
| **3** | RX |
| **4** | Shorted together with 1 and 6 |
| **5** | GND |
| **6** | Shorted together with 1 and 4 |
| **7** | Shorted together with 8 |
| **8** | Shorted together with 7 |
| **9** | Not connected |

## 7.3 RS-485 IN and RS-485 OUT (Female RJ11-6/6)



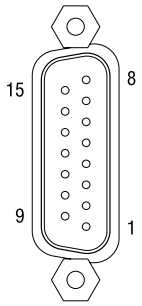
| Pin # | Description |
|---|---|
| **1** | GND |
| **2** | RX+ |
| **3** | RX- |
| **4** | TX- |
| **5** | TX+ |
| **6** | GND |

## 7.4 Keypad (Female RJ9-4/4)



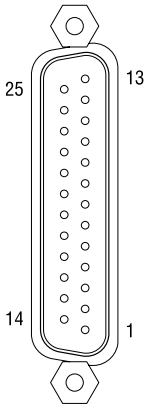
| Pin # | Description |
|---|---|
| **1** | +12 VDC |
| **2** | Tx |
| **3** | Rx |
| **4** | GND |

## 7.5 GPIO (Female Sub-D15)



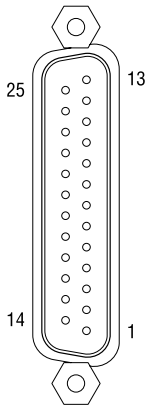

| Pin # | Description |
|---|---|
| **1** | Analog in |
| **2** | GND |
| **3** | OUT1 (Open collector, 30 V/40 mA Max.) |
| **4** | OUT2 (Open collector, 30 V/40 mA Max.) |
| **5** | OUT3 (Open collector, 30 V/40 mA Max.) |
| **6** | OUT4 (Open collector, 30 V/40 mA Max.) |
| **7** | GND |
| **8** | IN1 (2.21 kΩ pull up to 5 V) |
| **9** | IN2 (2.21 kΩ pull up to 5 V) |
| **10** | IN3 (2.21 kΩ pull up to 5 V) |
| **11** | IN4 (2.21 kΩ pull up to 5 V) |
| **12** | GND |
| **13** | In Motion (Open collector) |
| **14** | Not Referenced (Open collector) |
| **15** | GND |

## 7.6 DC Motor (Female Sub-D25)



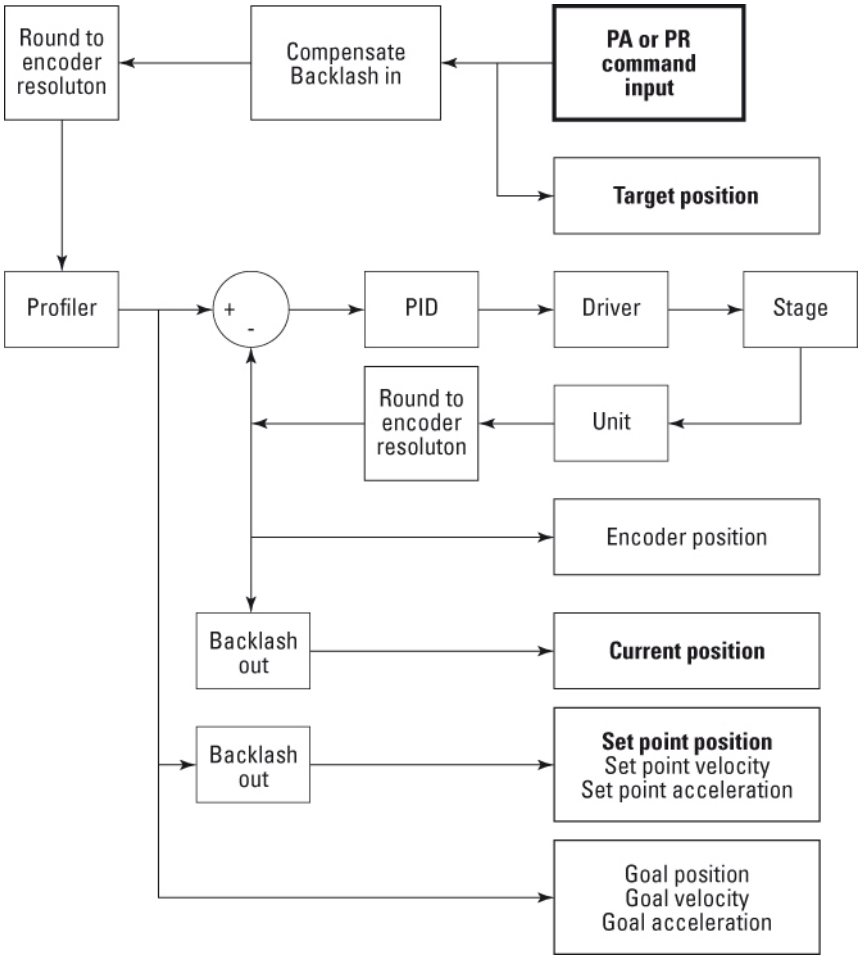

| Pin # | Description |
|---|---|
| **1** | Not connected |
| **2** | Not connected |
| **3** | Not connected |
| **4** | Not connected |
| **5** | MOTOR+ |
| **6** | MOTOR+ |
| **7** | MOTOR- |
| **8** | MOTOR- |
| **9** | Not connected |
| **10** | Not connected |
| **11** | Not connected |
| **12** | Not connected |
| **13** | ZM |
| **14** | GND |
| **15** | VI |
| **16** | GVD |
| **17** | EoR+ |
| **18** | EoR- |
| **19** | VA |
| **20** | VB |
| **21** | +5 V |
| **22** | GVD |
| **23** | /VA |
| **24** | /VB |
| **25** | /VI |

## 7.7 Stepper Motor (Female Sub-D25)




| Pin # | Description |
|---|---|
| **1** | Winding 1+ |
| **2** | Winding 1+ |
| **3** | Winding 1- |
| **4** | Winding 1- |
| **5** | Winding 2+ |
| **6** | Winding 2+ |
| **7** | Winding 2- |
| **8** | Winding 2- |
| **9** | Not connected |
| **10** | Not connected |
| **11** | Not connected |
| **12** | Not connected |
| **13** | ZM |
| **14** | GND |
| **15** | VI or N.C. if no encoder |
| **16** | GND |
| **17** | EoR+ |
| **18** | EoR- |
| **19** | VA or N.C. if no encoder |
| **20** | VB or N.C. if no encoder |
| **21** | +5 V |
| **22** | GND |
| **23** | /VA or N.C. if no encoder |
| **24** | /VB or N.C. if no encoder |
| **25** | /VI or N.C. if no encoder |

## 8.0 Backlash Compensation

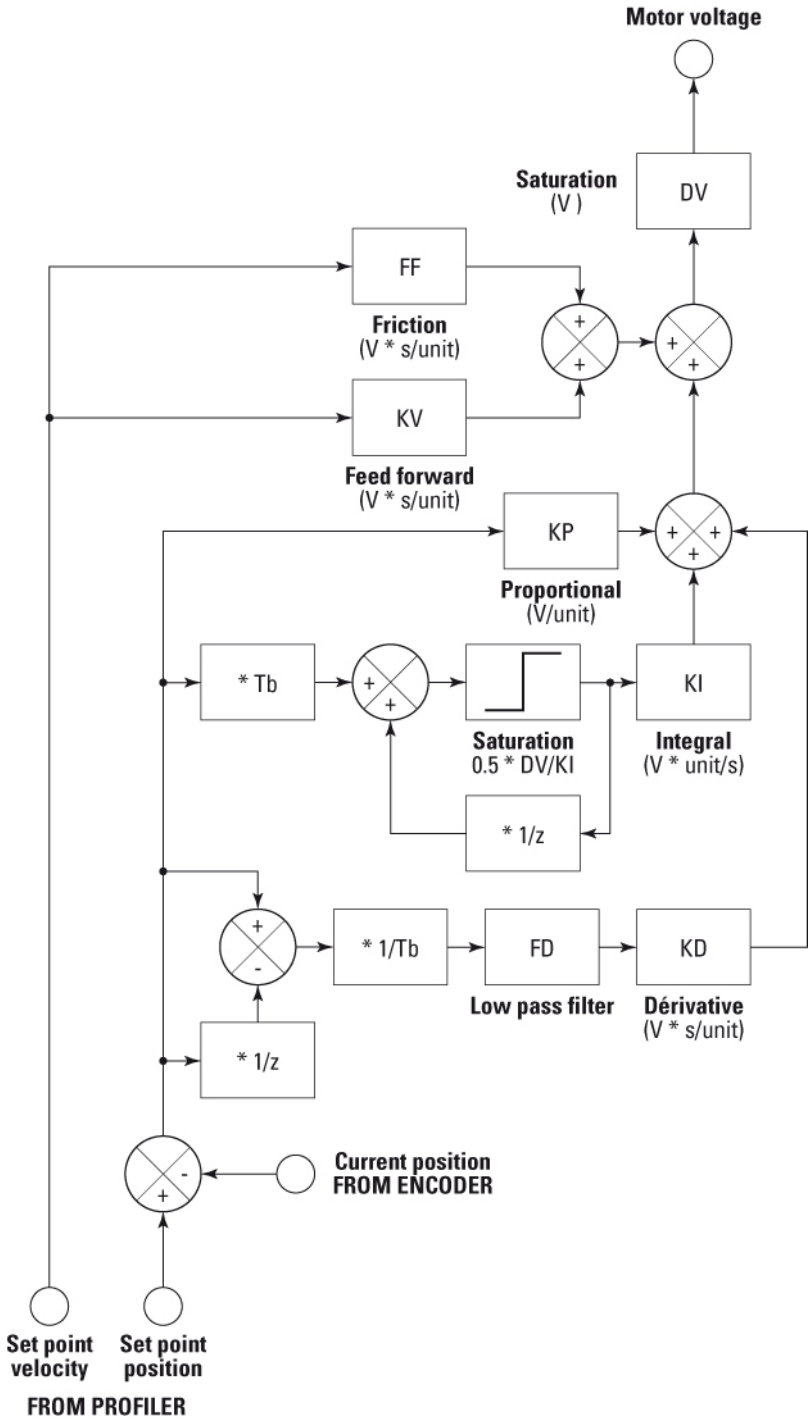



- Target position is read by PA command.
- Current position is read by TP command.
- Set-point position is read by TH command.
- Encoder resolution is set/read by the SU command.
- Backlash is set/read by the BA command.

## 9.0 ESP Stages

ESP refers to Newport stages with an EEPROM (ESP chip), that contains all stage information like motor type, travel limits, maximum speeds, etc. The SMC100CC/PP is capable reading this information from the stage and can save it to the controller's flash memory. This minimizes the stage configuration time and possible errors during configuration. The SMC100CC/PP can also be configured to confirm at each power-up that the connected stage is the same as the one recorded in the controller's memory, which is another safety feature.

## 10.0 PID Control Loop Structure



# 11.0 Maintenance and Service

## 11.1 Enclosure Cleaning

The SMC100CC/PP Controller/Driver should only be cleaned with a lightly damped cloth or sponge with a soapy water solution. Do not use an acetone or alcohol solution, this will damage the finish of the enclosure.

## 11.2 Obtaining Service

The SMC100CC/PP Controller/Driver contains no user serviceable parts. To obtain information regarding factory service, contact Newport Corporation or your Newport representative. Please have the following information available:

- Instrument model number (on front panel).
- Instrument serial number (on rear panel) or original order number.
- Description of the problem.

If the instrument is to be returned to Newport Corporation, you will be given a Return Number, which you should reference in your shipping documents.

Complete a copy of the Service Form as represented on the next page and include it with your shipment.

# 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: _____

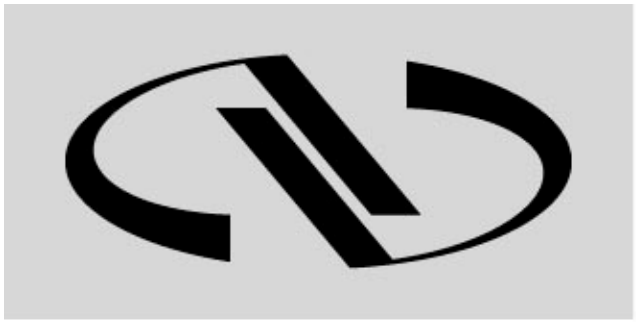
Phone Number: _____

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