

## PZ241E E-871 PIShift Controller User Manual

Version: 1.0.0

Date: 20.02.2013



This document describes the following product:

- **E-871.1A1**  
PIShift Controller, 1 Channel, Linear Encoder



Physik Instrumente (PI) GmbH & Co. KG is the owner of the following trademarks:  
PI®, PIC®, PICMA®, Picoactuator®, PIFOC®, PILine®, PInano®, PiezoWalk®,  
NEXACT®, NEXLINE®, NanoCube®, NanoAutomation®

The following designations are protected company names or registered trademarks of third parties:

Microsoft, Windows, LabVIEW

© 2013 Physik Instrumente (PI) GmbH & Co. KG, Karlsruhe, Germany. The text, photographs and drawings in this manual are protected by copyright. With regard thereto, Physik Instrumente (PI) GmbH & Co. KG retains all the rights. Use of said text, photographs and drawings is permitted only in part and only upon citation of the source.

Original instructions

First printing: 20.02.2013

Document number: PZ241E, BRo, version 1.0.0

Subject to change without notice. This manual is superseded by any new release. The latest release is available for download (p. 5) on our website.



# Contents

<b>1</b>	<b>About this Document</b>	<b>1</b>
1.1	Goal and Target Audience of this User Manual .....	1
1.2	Symbols and Typographic Conventions .....	1
1.3	Definition .....	3
1.4	Other Applicable Documents .....	4
1.5	Downloading Manuals .....	5
<b>2</b>	<b>Safety</b>	<b>7</b>
2.1	Intended Use .....	7
2.2	General Safety Instructions .....	7
2.3	Organizational Measures .....	8
<b>3</b>	<b>Product Description</b>	<b>9</b>
3.1	Features and Applications .....	9
3.2	Product View .....	10
3.2.1	Front Panel .....	10
3.2.2	Rear Panel .....	12
3.3	Scope of Delivery .....	13
3.4	Accessories .....	13
3.5	Functional Principles .....	15
3.5.1	Block Diagram .....	15
3.5.2	Commandable Items .....	15
3.5.3	Important Components of the Firmware .....	18
3.5.4	Operating Modes .....	19
3.5.5	Physical Units .....	22
3.5.6	Motion Triggering .....	23
3.5.7	Servo Algorithm and Other Control Value Corrections .....	26
3.5.8	On-Target State .....	29
3.5.9	Reference Point Switch Detection .....	30
3.5.10	Limit Switch Detection .....	31
3.5.11	Travel Range and Soft Limits .....	32
3.5.12	Reference Point Definition .....	37
3.5.13	ID Chip Detection .....	41
3.6	Communication Interfaces .....	42
3.7	Overview of PC Software .....	43
3.8	Stage Databases .....	44

4	Unpacking	47
5	Quick Start	49
6	Installation	55
6.1	General Notes on Installation .....	55
6.2	Installing the PC Software .....	55
6.2.1	Performing the Initial Installation .....	55
6.2.2	Installing Updates .....	56
6.2.3	Installing a Custom Stage Database .....	59
6.3	Ensuring Ventilation.....	59
6.4	Mounting the E-871 .....	60
6.5	Connecting the E-871 to the Protective Earth Conductor .....	61
6.6	Connecting the Power Supply to the E-871 .....	62
6.7	Connecting the Stage .....	63
6.8	Connecting the PC .....	63
6.8.1	Connecting to the RS-232 Interface .....	64
6.8.2	Connecting to the USB Interface .....	64
6.8.3	Setting Up a Daisy Chain Network .....	65
6.9	Connecting an HID Device .....	66
6.10	Connecting Digital In- and Outputs .....	68
6.10.1	Connecting the Outputs.....	68
6.10.2	Connecting the Inputs.....	69
6.11	Connecting Analog Signal Sources.....	70
7	Start-Up	71
7.1	General Notes on Start-Up.....	71
7.2	Adapting the DIP Switch Settings.....	72
7.2.1	General Procedure .....	72
7.2.2	Controller Address.....	73
7.2.3	Baud Rate.....	74
7.2.4	Update Mode .....	74
7.3	Switching on the E-871.....	75
7.4	Establishing Communication .....	76
7.4.1	Establishing Communication via RS-232 .....	76
7.4.2	Establishing Communication via USB .....	77
7.4.3	Establishing Communication for Networked Controllers .....	79
7.5	Starting Motions.....	84
7.6	Setting the Notch Filter .....	89
7.7	Optimizing the Servo-Control Parameters.....	94
8	Operation	99
8.1	Motion Errors .....	99
8.1.1	Protective Functions of the E-871 .....	99

8.1.2	Re-establishing Readiness for Operation.....	99
8.2	Data Recorder .....	100
8.2.1	Data Recorder Properties .....	100
8.2.2	Setting up the Data Recorder .....	100
8.2.3	Starting the Recording .....	101
8.2.4	Reading Out Recorded Data .....	102
8.3	Digital Output Signals .....	102
8.3.1	Commands for Digital Outputs .....	102
8.3.2	Setting Up "Position Distance" Trigger Mode .....	104
8.3.3	Setting Up "On Target" Trigger Mode.....	106
8.3.4	Setting Up "Motion Error" Trigger Mode .....	107
8.3.5	Setting Up "In Motion" Trigger Mode .....	108
8.3.6	Setting Up "Position + Offset" Trigger Mode .....	108
8.3.7	Setting Up "Single Position" Trigger Mode .....	110
8.3.8	Setting Signal Polarity .....	111
8.4	Digital Input Signals.....	111
8.4.1	Commands and Parameters for Digital Inputs .....	112
8.4.2	Using Digital Input Signals in Macros .....	114
8.4.3	Using Digital Input Signals as Switch Signals .....	114
8.4.4	Using Digital Input Signals for the HID Control .....	116
8.5	Analog Input Signals.....	116
8.5.1	Commands for Analog Inputs .....	117
8.5.2	Using Analog Input Signals in Macros.....	118
8.6	Control with an HID Device .....	118
8.6.1	Functionality of the HID Control.....	118
8.6.2	Commands and Parameters for HID Devices .....	119
8.6.3	Testing an HID Device.....	120
8.6.4	Setting Up and Enabling the HID Control.....	122
8.6.5	Calibrating Axes of HID Devices .....	124
8.6.6	Permanently Saving the Configuration of the HID Control .....	127
8.6.7	Available HID devices.....	129
8.7	Controller Macros .....	131
8.7.1	Overview: Macro Functionalities and Example Macros.....	131
8.7.2	Commands and Parameters for Macros .....	132
8.7.3	Working with Macros .....	133
8.7.4	Macro Example: Stopping Motion by Pushbutton .....	143
8.7.5	Macro example: HID control alternating with relative motions .....	145
9	GCS Commands .....	149
9.1	Notation .....	149
9.2	GCS Syntax for Syntax Version 2.0 .....	150
9.3	Target and Sender Address .....	152
9.4	Variables.....	153
9.5	Command Overview .....	155
9.6	Command Descriptions for GCS 2.0 .....	159
9.7	Error Codes .....	258

10	Adapting Settings	271
10.1	Changing Parameters in the E-871 .....	271
10.1.1	General Commands for Parameters.....	272
10.1.2	Saving Parameter Values in a Text File .....	272
10.1.3	Changing Parameters: General Procedure .....	273
10.2	Creating or Modifying a Stage Type .....	277
10.3	Parameter Overview .....	280
11	Maintenance	291
11.1	Cleaning the E-871 .....	291
11.2	Updating Firmware .....	291
12	Troubleshooting	297
13	Customer Service	301
14	Technical Data	303
14.1	Specifications.....	303
14.1.1	Data Table .....	303
14.1.2	Maximum Ratings.....	304
14.1.3	Ambient Conditions and Classifications .....	305
14.2	System Requirements .....	306
14.3	Dimensions.....	306
14.4	Pin Assignment.....	308
14.4.1	Motor.....	308
14.4.2	Sensor .....	309
14.4.3	I/O .....	310
14.4.4	C-170.IO Cable for Connecting to the I/O Socket .....	310
14.4.5	Joystick .....	311
14.4.6	C-819.20Y Cable for C-819.20 Joystick .....	312
14.4.7	RS-232 In and RS-232 Out .....	313
14.4.8	Power supply connector 24 V DC.....	314
15	Old Equipment Disposal	315

# 1 About this Document

## In this Chapter

Goal and Target Audience of this User Manual .....	1
Symbols and Typographic Conventions .....	1
Definition .....	3
Other Applicable Documents .....	4
Downloading Manuals .....	5

## 1.1 Goal and Target Audience of this User Manual

This manual contains information on the intended use of the E-871.

It assumes that the reader has a fundamental understanding of basic servo systems as well as motion control concepts and applicable safety procedures.

The latest versions of the user manuals are available for download (p. 5) on our website.

## 1.2 Symbols and Typographic Conventions

The following symbols and typographic conventions are used in this user manual:

### CAUTION



#### Dangerous situation

If not avoided, the dangerous situation will result in minor injury.



- Actions to take to avoid the situation.

### NOTICE



#### Dangerous situation


If not avoided, the dangerous situation will result in damage to the equipment.

- Actions to take to avoid the situation.

**INFORMATION**

Information for easier handling, tricks, tips, etc.

---

Symbol/ Label	Meaning
1. 2.	Action consisting of several steps whose sequential order must be observed
➤	Action consisting of one or several steps whose sequential order is irrelevant
■	List item
p. 5	Cross-reference to page 5
RS-232	Labeling of an operating element on the product (example: socket of the RS-232 interface)
	Warning signs affixed to the product that refer to detailed information in this manual.
<b>Start &gt; Settings</b>	Menu path in the PC software (example: to open the menu, the <b>Start</b> and <b>Settings</b> menu items must be clicked in succession)
SVO?	Command line or a command from PI's General Command Set (GCS) (example: command to get the servo mode).
<b>Device S/N</b>	Parameter name (example: parameter where the serial number is stored)
5	Value that must be entered or selected via the PC software



## 1.3 Definition

Term	Explanation
PC software	Software that is installed on the PC.
Firmware	Software that is installed on the controller.
Volatile memory	RAM module in which the parameters are saved when the controller is switched on (working memory).
Nonvolatile memory	EEPROM memory chip (read-only memory), from which the default values of the parameters are loaded to the volatile memory when the controller is started.
Axis	Also referred to as "logical axis". The logical axis reflects the motion of the stage in the firmware of the E-871. For stages that allow motion in several directions (e. g. in X, Y and Z), each direction of motion corresponds to a logical axis.
Stage	<p>Mechanical system connected to the E-871. For stages having just one motion axis the designation "axis" is synonymous with "stage". Stages that allow motion in several axes are also designated as "multiaxis stages". For these stages, a distinction must be made between the individual axes.</p> <p>In this manual, actuators, i. e. drive components without a moving platform (e. g. precision linear actuators), are designated as stages as well.</p>
Incremental position sensor	Sensor (encoder) for capturing changes of position or changes of angle. Signals from the incremental position sensor are used for axis position feedback. After switching on the controller a reference point definition must be performed before absolute target positions can be commanded and reached.
Control value	<p>The control value is the input for the PIShift drive electronics integrated in the E-871. The PIShift drive electronics generates the following output signals from the control value:</p> <ul style="list-style-type: none"> <li>▪ Step mode: conversion to the modified sawtooth signal for the stage</li> <li>▪ Linear mode: "linear" conversion to an analog signal</li> </ul>
GCS	PI General Command Set; command set for PI controllers. Piezo drivers and servo controllers can be operated conjointly with minimal programming effort thanks to the GCS.

Term	Explanation
HID device	Abbreviation for "Human Interface Device". "HID device" refers to an input or output device that is connected to the controller and is intended for manual operation. Depending on the controller, the connection can be made via USB, analog or digital interfaces. Typical HID devices are joysticks and gamepads.
HID control	Control of a motion parameter of the axis of the E-871 via the displacement of an axis of the HID device.

## 1.4 Other Applicable Documents

The devices and software tools which are mentioned in this documentation are described in their own manuals.

The latest versions of the user manuals are available for download (p. 5) on our website.

Description	Document
Short version of the manual for the E-871	PZ241Equ User Manual Short Version
GCSLabVIEW driver library for E-871	PZ242E Software Manual
PI GCS 2.0 DLL for E-871	PZ243E Software Manual
GCS array data format description	SM146E Software Manual
PIMikroMove	SM148E Software Manual
PIStageEditor software for the management of stage databases	SM144E Software Manual
PI Update Finder: Search and download updates	A000T0028 Technical Note
PI Update Finder: Updating PC without Internet connection	A000T0032 Technical Note

## 1.5 Downloading Manuals

### INFORMATION

If a manual is missing on our website or if there are problems in downloading:

- Contact our customer service department (p. 301).

The current versions of the manuals are found on our website. To download a manual, proceed as follows:

1. Open the website **<http://www.pi-portal.ws>**.
2. Click **Downloads**.
3. Click the corresponding category (e. g. **E Piezo Drivers & Nanopositioning Controllers**).
4. Click the corresponding product code (e. g. **E-871**).

An overview of the available file types is shown for the selected product.

5. If **(0 Files)** is shown in the **Documents** line, log in as follows to display and download the documents:
  - a) Insert the product CD in the corresponding PC drive.
  - b) Open the **Manuals** directory.
  - c) Open the Release News (e. g. **E-871\_Releasenews\_V\_1\_0\_0.pdf**) on the CD of the product.
  - d) Find the user name and password in the **User login for software download** section in the Release News.
  - e) In the **User login** area on the left margin in the website, enter the user name and the password in the corresponding fields.
  - f) Click **Login**.

If **Documents (0 Files)** is still being displayed, no manuals are available:

- Contact our customer service department (p. 301).

6. Click **Documents**.
7. Click the desired manual and save it on the hard disk of your PC or on a data storage medium.



## 2 Safety

### In this Chapter

Intended Use .....	7
General Safety Instructions .....	7
Organizational Measures.....	8

### 2.1 Intended Use

The E-871 is a laboratory device as defined by DIN EN 61010-1. It is intended to be used in interior spaces and in an environment which is free of dirt, oil, and lubricants.

According to its type, the E-871 is intended for the operation of stages with PIShift piezo inertia drives.

The E-871 is intended for closed-loop operation with incremental position sensors. In addition, it can read and process the reference point and limit switch signals from the stage connected.

The E-871 must not be used for purposes other than those named in this user manual. In particular, the E-871 must not be used to drive ohmic or inductive loads.

### 2.2 General Safety Instructions

The E-871 is built according to state-of-the-art technology and recognized safety standards. Improper use can result in personal injury and/or damage to the E-871.

- Only use the E-871 for its intended purpose, and only use it if it is in a good working order.
- Read the user manual.
- Immediately eliminate any faults and malfunctions that are likely to affect safety.

The operator is responsible for the correct installation and operation of the E-871.

- Install the E-871 near the power source so that the power plug can be quickly and easily disconnected from the mains.
- Use the supplied components (power supply, adapter and power cord (p. 13)) to connect the E-871 to the power source.
- If one of the supplied components for connecting to the power source has to be replaced, use a sufficiently dimensioned component.

## 2.3 Organizational Measures

### User manual

- Always keep this user manual available by the E-871.  
The latest versions of the user manuals are available for download (p. 5) on our website.
- Add all information given by the manufacturer to the user manual, for example supplements or Technical Notes.
- If you pass the E-871 on to other users, also turn over this user manual as well as other relevant information provided by the manufacturer.
- Only use the device on the basis of the complete user manual. Missing information due to an incomplete user manual can result in minor injury and property damage.
- Only install and operate the E-871 after having read and understood this user manual.

### Personnel qualification

The E-871 may only be started up, operated, maintained and cleaned by authorized and qualified staff.

## 3 Product Description

### In this Chapter

Features and Applications .....	9
Product View .....	10
Scope of Delivery .....	13
Accessories .....	13
Functional Principles .....	15
Communication Interfaces.....	42
Overview of PC Software .....	43
Stage Databases .....	44

### 3.1 Features and Applications

#### Digital servo controller for PIShift piezomotors

1 channel. Integrated power amplifier and voltage generator for PIShift piezo inertia drives. Point-to-point motions. Linear mode for nanometer-precision positioning.

#### Extensive functionality

Powerful macro command language. Nonvolatile macro storage, e. g. for stand-alone functionality with autostart macro. Data recorder. ID chip for quick start-up. Parameter changes during operation. Extensive software support, e. g. for LabVIEW, shared libraries for Windows and Linux.

#### Motion controller of the Mercury class

Daisy-Chain networking for up to 16 axes operated via a common computer interface.


Interfaces: USB and RS-232 for commands. A/B (quadrature) encoder input. TTL inputs for limit and reference point switches. I/O ports (analog/digital) for automation. Interface for analog joystick.

## 3.2 Product View

### 3.2.1 Front Panel



Figure 1: E-871 PIShift controller, front view

Labeling	Type	Function
RS-232 In	Sub-D 9(m) (p. 313)	Serial connection to the PC or to the previous controller in a daisy chain network; do not connect to the PC if the USB interface is already connected.
RS-232 Out	Sub-D 9(f) (p. 313)	Serial connection to the subsequent controller in a daisy chain network
	Mini-USB type B	Universal serial bus for connecting to the PC; do not connect if <b>RS-232 In</b> is already connected.
STA	LED green/off	Controller state: <ul style="list-style-type: none"> <li>green: E-871 is ready for normal operation</li> <li>off: E-871 is not connected to the supply voltage or is in firmware update mode (selection via DIP switch 8)</li> </ul>
ERR	LED red/off	Error indicator: <ul style="list-style-type: none"> <li>Continuously lit: Error (error code <math>\neq</math> 0)</li> <li>Off: No error (error code = 0)</li> </ul> <p>The error code can be queried with the <code>ERR?</code> command. The query resets the error code to zero and the LED is switched off.</p>






Labeling	Type	Function
<b>I/O</b>	Mini-DIN socket, 9-pin (p. 310)	Digital in-/outputs: <ul style="list-style-type: none"><li>▪ Outputs: Triggering external devices</li><li>▪ Inputs: Use in macros, as switch signals or for HID control</li></ul> Analog inputs: <ul style="list-style-type: none"><li>▪ Use in macros or for scanning processes</li></ul>
<b>Joystick</b>	Mini-DIN socket, 6-pin (p. 311)	Analog HID device (joystick): <ul style="list-style-type: none"><li>▪ Inputs for signals from the axes and buttons of the HID device</li><li>▪ Output for the supply voltage of the HID device</li></ul>
<b>Mode, Baud, Addr</b>	8-bit DIP switch (p. 72)	Setting of: <ul style="list-style-type: none"><li>▪ Device address</li><li>▪ Baud rate for communication with the PC</li><li>▪ Update mode</li></ul>

### 3.2.2 Rear Panel



Figure 2: E-871 PIShift controller, rear view


Labeling	Type	Function
	Threaded bolt with fastening material for protective earth conductor	Protective earth connection (p. 61) The threaded bolt must be connected to a protective earth conductor, since the E-871 is not grounded via the power supply connector.
24 V  2.5 A	M8 panel plug, 4-pin (p. 314)	Connection for the supply voltage
Sensor	HD Sub-D 15(m) (p. 309)	Stage connection <ul style="list-style-type: none"> <li>Input of the signals of the incremental position sensor</li> <li>Input of the signals from the limit switches and reference point switch</li> <li>Input and output for signals of the ID chip</li> <li>Output of the supply voltage for position sensor, reference point and limit switches</li> </ul>
Motor  0 V to 100 V	HD Sub-D 15(f) (p. 308)	Stage connection. Only for PIShift inertia drives! <ul style="list-style-type: none"> <li>Output of the piezo voltage for the piezo actuator in the stage</li> <li>Input of the signals from the limit switches and reference point switch</li> <li>Input and output for signals of the ID chip</li> </ul>

### 3.3 Scope of Delivery

Order Number	Items
E-871.1A1	PIShift controller
C-663.PS	Wide-range-input power supply 24 V / 42 W
3763	Power cord
K050B0003	Adapter for the power supply connector; barrel connector to M8 4-pin connector, A-coded
C-815.34	RS-232 null-modem cable, 3 m, 9/9-pin
C-862.CN	Cable for daisy chain, 30 cm
000014651	USB cable (type A to Mini-B) for connection to the PC
E-871.CD	Product CD with software and user manuals for the E-871
PZ241Equ	Short version of the user manual for the E-871

### 3.4 Accessories

Order Number	Description
C-862.CN2	Cable for daisy chain, 180 cm
C-819.20	Analog joystick for 2 axes, details see "Available HID Devices" (p. 129)
C-819.20Y	Y cable for connecting 2 controllers to C-819.20 joystick
C-819.30	Analog joystick for 3 axes, details "Available HID Devices" (p. 130)

Order Number	Description
C-170.PB	<p>Pushbutton box, 4 buttons and 4 LEDs.</p> <p>Connection to the <b>I/O</b> socket of the E-871, sends 4 TTL input signals and displays the state of the 4 digital outputs via the LEDs.</p> 
C-170.IO	I/O cable, 2 m, open end (p. 310)

To order, contact our customer service department (p. 301).

## 3.5 Functional Principles

### 3.5.1 Block Diagram

The E-871 controls the motion of a logical axis of a stage. The following block diagram shows how the E-871 generates the piezo voltage for the connected axis.

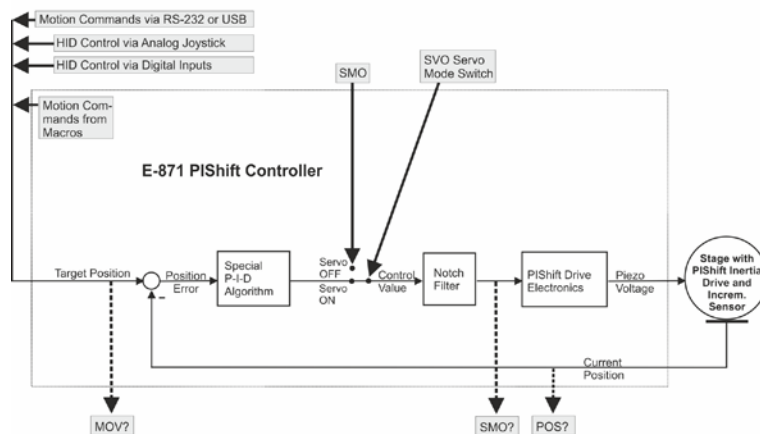


Figure 3: E-871.1A1: Control value generation

The E-871 supports stages with PIShift inertia drive and incremental sensor.

### 3.5.2 Commandable Items

The following table contains the items that can be accessed with commands of the GCS (p. 159).

Item	Number	Identifier	Description
Logical axis	1	1 (modifiable)	<p>The logical axis represents the motion of the stage in the firmware of the E-871. It corresponds to the axis of a linear coordinate system.</p> <p>Motions for logical axes are commanded in the firmware of the E-871 (i.e. for the directions of motion of a stage). The motion commands <b>MOV</b> and <b>MVR</b>, for example, are available in closed-loop operation. Motions in open-loop operation are triggered with <b>SMO</b> and <b>STE</b>.</p> <p>The axis identifier can be queried with the <b>SAI?</b> command and modified with the <b>SAI</b> command. It can comprise up to 8 characters; valid characters are 1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ_</p>

Item	Number	Identifier	Description
			<p>The new axis identifier is saved automatically in the nonvolatile memory and is thus still available even after a reboot or after the next switching-on.</p> <p>When the <b>Stage Name</b> parameter (0x3C) has the value NOSTAGE, the axis is "deactivated". A deactivated axis is not accessible for axis-related commands (e.g. motion commands or position queries). The identifier of a deactivated axis can only be queried with <code>SAI? ALL</code>.</p>
Analog inputs	8	1 to 8	<p>The analog input lines with the identifiers 1 to 4 are the inputs 1 to 4 of the <b>I/O</b> socket (p. 310). Their number is displayed with the <code>TAC?</code> command and their values can be queried with the <code>TAV?</code> command. Note that these lines can also be used as digital inputs (see below).</p> <p>Additional analog input lines are located on the <b>Joystick</b> socket (p. 311).</p> <p>These lines are not output via <code>TAC?</code> and <code>TAV?</code>.</p> <p>The values of all inputs can be recorded via record option 81 of the <code>DRC</code> command.</p>
Digital outputs	4	1 to 4	<p>1 to 4 identify digital output lines 1 to 4 of the <b>I/O</b> socket (p. 310).</p> <p>Further information see "Digital Output Signals" (p. 102).</p>
Digital inputs	4	1 to 4	<p>1 to 4 identify digital input lines 1 to 4 of the <b>I/O</b> socket (p. 310), which can also be used as analog inputs (see above).</p> <p>Further information see "Digital Input Signals" (p. 111).</p>

Item	Number	Identifier	Description
HID device	1	1	The HID device (p. 3) is used for the HID control (p. 3) of the logical axis of the E-871. Information on the axes and buttons of the HID device can be queried with the <b>HIS?</b> command.
Axes of the HID device	4	1 to 4	Two axes of the HID device can be connected at the <b>Joystick</b> socket (p. 311). Connection options: <ul style="list-style-type: none"> <li>Pin 4 (0 to 3.3 V): command as axis 1 of the HID device</li> <li>Pin 2 (-10 to 10 V): command as axis 2 of the HID device</li> </ul> Two axes of the HID device can also be connected at the <b>I/O</b> socket (p. 310). Connection options: <ul style="list-style-type: none"> <li>Pins 1 and 2 (TTL signals): command as axis 3 of the HID device</li> <li>Pins 3 and 4 (TTL signals): command as axis 4 of the HID device</li> </ul>
Buttons of the HID device	2	1, 2	The two buttons of the HID device can be connected at the <b>Joystick</b> socket (p. 311). Connection options: <ul style="list-style-type: none"> <li>Pin 5 (0 or 3.3 V): command as button 1 of the HID device</li> <li>Pin 6 (0 or 3.3 V): command as button 2 of the HID device</li> </ul> Further information see "Control with an HID Device" (p. 118). For data recorder configuration with the <b>DRC</b> command, the following data source identifiers apply, notwithstanding the above specifications: <p>5 = Axis 1 of the HID device  6 = Button 1 of the HID device  7 = Axis 2 of the HID device  8 = Button 2 of the HID device</p>
Data recorder tables	2	1, 2	The E-871 has 2 data recorder tables (query with <b>TNR?</b> ) with 1024 data points per table.
Controller address	1	1 to 16	The controller address can be set within the range of 1 to 16 with the DIP switches on the front panel of the E-871 (p. 73). In a daisy chain (p. 65), each controller must have a unique address (p. 152).

### 3.5.3 Important Components of the Firmware

The firmware of the E-871 provides the following functional units:

Firmware Component	Description
ASCII commands	<p>Communication with the E-871 can be managed using the commands of the PI General Command Set (GCS; version 2.0). The GCS is independent of the hardware (controller, stages connected).</p> <p>Examples of the use of GCS:</p> <ul style="list-style-type: none"> <li>▪ Configuring the E-871</li> <li>▪ Setting operating mode</li> <li>▪ Starting stage motions</li> <li>▪ Getting system and position values</li> </ul> <p>You can find a list of available commands in the "Command Overview " section (p. 155).</p>
Parameters	<p>Parameters reflect the properties of the connected stage (e.g. travel range) and specify the behavior of the E-871 (e.g. settings for the servo algorithm or for using the digital inputs).</p> <p>The parameters can be divided into the following categories:</p> <ul style="list-style-type: none"> <li>▪ Protected parameters whose default settings cannot be changed</li> <li>▪ Parameters that must be set by the user to adapt to the application</li> </ul> <p>Further information can be found in the "Adapting Settings" section (p. 271).</p> <p>In the case of stages with ID chip, the values of some parameters are stored on the ID chip (p. 41). They are loaded to the volatile memory when the E-871 is switched on or rebooted.</p>
Command levels	<p>The command levels determine the write permission for the parameters.</p> <p>The current command level can be changed with the <code>CCL</code> command. This may require entering a password.</p>
Servo algorithm	<p>The position error that results from the difference between the target position and the actual position (sensor feedback) runs through a PID servo algorithm. You can find further information in the "Servo Algorithm and Other Control Value Corrections" (p. 26) and "Motion Triggering" (p. 23) sections.</p>
Data recorder	<p>The E-871 contains a real-time data recorder (p. 100). The data recorder can record various signals (e. g. position, analog input) from different data sources (e. g. logical axes or input channels).</p>



Firmware Component	Description
Macros	The E-871 can save macros (p. 131). Command sequences can be defined and permanently stored in the nonvolatile memory of the device via the macro function. A start-up macro can be defined that is executed each time that the E-871 is switched on or rebooted. This simplifies stand-alone operation (operation without a connection to the PC). Further information can be found in the "Controller Macros" section (p. 131).

The firmware can be updated with a tool (p. 291).

### 3.5.4 Operating Modes

#### Servo modes

The servo mode determines whether the motion is performed in closed-loop operation or in open-loop operation.

Operating Mode	Description
Closed-loop operation (servo mode On)	<p>The position error that results from the difference between the target position and the actual position (sensor feedback) runs through a PID servo algorithm (proportional integral differential).</p> <p>The commanded target position determines whether the motion is executed as a sequence of linear and step mode or only in linear mode. Further information see "Servo Algorithm and Other Control Value Corrections" (p. 26).</p>
Open-loop operation (servo mode Off)	<p>In open-loop operation, the E-871 does not evaluate the signals of the position sensor.</p> <p>The value of the parameter 0x1F000702 and the motion command sent determine whether the motion takes place in linear mode or in step mode. Further information, see "Motion Triggering" (p. 23) and "PIShift Drive Modes" (p. 20).</p>

**INFORMATION**

The E-871 is intended for closed-loop operation with incremental position sensors (servo mode On). After switching-on, open-loop operation is enabled by default (servo mode Off).

- Get the current operating mode with the `SVO?`, `#4` or `SRG?` commands.
- Enable closed-loop operation with the `SVO` command.
- If necessary, program a start-up macro that starts the E-871 via the `SVO` command in closed-loop operation; see "Setting up a start-up macro" (p. 141).

**PIShift drive modes**

In the PIShift inertia drive, a piezo actuator acts on a moving rod to generate the motion. The E-871 supports the following drive modes for PIShift inertia drives:

Operating Mode	Description
Step mode	<p>The PIShift drive electronics in the E-871 converts the control value to a modified sawtooth signal with a maximum frequency of 25 kHz and outputs the corresponding piezo voltage. The piezo voltage generates a cyclic alternation of static and sliding friction between the moving rod and the piezo actuator and thus a continuous feed of the rod. The output of one period of the modified sawtooth signal generates one "step" of the rod.</p> <p>The travel range is only limited by the physical limits of the stage.</p>
Linear mode	<p>The PIShift drive electronics in the E-871 converts the control value "linearly" to an analog signal. The output piezo voltage corresponds to 10 times this analog signal. The feed of the rod is generated by the expansion of the piezo actuator caused by the piezo voltage. The piezo actuator achieves its maximum expansion when the E-871 outputs the maximum permissible piezo voltage.</p> <p>The travel range is limited by the maximum expansion of the piezo actuator.</p>

The E-871 is configured with the parameters listed in the table for the connected PIShift inertia drive. With the exception of the value for the parameter 0x1F000702, all parameter values can be loaded from a stage database (p. 44) with the PC software from PI (recommended).

Parameter	Description and possible values
<b>PIShift Upper Supply Voltage (V)</b> 0x1F000000	Maximum value of the piezo voltage for the PIShift inertia drive The value depends on the type of the drive.
<b>PIShift Lower Supply Voltage (V)</b> 0x1F000100	Minimum value of the piezo voltage for the PIShift inertia drive The value depends on the type of the drive.
<b>PIShift Forward Current (A)</b> 0x1F000200	Maximum current consumption of the PIShift inertia drive during the forward motion in step mode. The value depends on the type of the drive.
<b>PIShift Backward Current (A)</b> 0x1F000300	Maximum current consumption of the PIShift inertia drive during the backward motion in step mode. The value depends on the type of the drive.
<b>PIShift Frequency (Hz)</b> 0x1F000400	Frequency of the piezo voltage for the step mode of the PIShift inertia drive (= frequency of the modified sawtooth signal; "step frequency") Determines the velocity of the drive in step mode. If the parameter 0x1F000702 has the value 0 and motions are started in open-loop operation with the SMO command (p. 236): The step frequency directly depends on the control value commanded with SMO and is limited by the parameters 0x1F000400 and 0x9.
<b>PIShift Charge Cycle</b> 0x1F000500	Duty cycle of the current source during the output of one period of the modified sawtooth in step mode 0 to 1 The value depends on the type of the drive.

Parameter	Description and possible values
<b>PIShift Open-Loop Driving Mode</b> 0x1F000702	PIShift drive mode in open-loop operation 0 = step mode 1 = linear mode (default) This parameter is evaluated when motions are started with the SMO command. The value of the parameter has no effect on motions that are started with the STE command (p. 244) in open-loop operation.

Special settings for closed-loop operation, see "Control Algorithm and Other Control Value Corrections" (p. 26).

#### INFORMATION

The PIShift drive can develop noises in step mode. The noise development depends on the current step frequency.

### 3.5.5 Physical Units

The E-871 supports various units of length for positions. The adaptation is made via a factor with which the counts of the incremental encoder are converted into the physical unit of length required. The conversion factor is set with the following parameters:

Parameter	Description and possible values
<b>Numerator Of The Counts-Per-Physical-Unit Factor</b> 0xE	Numerator and denominator of the factor for counts per physical length unit 1 to 1,000,000 for each parameter. The factor for the counts per physical unit of length specifies the unit of length for position queries and motion commands in closed-loop operation.
<b>Denominator Of The Counts-Per-Physical-Unit Factor</b> 0xF	The values of every parameter whose unit is either the physical unit of length itself or a unit of measurement based on it are automatically adapted to the set factor. The factor for the counts per physical unit of length has no impact on the stability of the servo loop but is used for the input and output scaling of position values.

The unit symbol can be customized for display purposes with the following parameter:

Parameter	Description and possible values
<b>Axis Unit</b> 0x07000601	Unit symbol Maximum of 20 characters. The unit symbol is "MM", for example, if the factor for the counts per physical unit of length is set with the 0xE and 0xF parameters so that the encoder counts are converted into millimeters. The unit symbol for rotation stages normally is "deg". The value of the parameter 0x07000601 is not evaluated by the E-871 but is only used by the PC software for display purposes. Examples: 1 encoder count = 100 nm Counts per physical length unit: 10000:1 → Unit symbol: mm 1 encoder count = 0.254 mm Counts per physical length unit: 100:1 → Unit symbol: in

### 3.5.6 Motion Triggering

#### Motions in closed-loop operation

Trigger of the motion	Commands	Description
Motion commands, sent by the command line or via the PC software	MOV, MVR	Motion to absolute or relative target position
	GOH	Motion to zero position
	STE	Starts performing a step of a specified distance and records the step response
	FNL, FPL, FRF	Starting reference moves
	FED	Starting moves to signal edges

Trigger of the motion	Commands	Description
Controller macros with motion commands	MAC  Additional macro commands and information see "Controller Macros" (p. 131).	Calls a macro function. Permits recording, deleting and running macros on the controller.  Any commands can be sent from the command line when a macro is running on the controller. The macro content and motion commands received from the command line can overwrite each other.
HID control	HIN  HIA        SST  For further commands, see "Control with an HID Device" (p. 118).	Enables or disables the control of the axis of the E-871 by the axes of the HID devices ("HID control").  Configures the HID control for the axis of the E-871. The following motion parameters of the E-871 axis can be controlled via the axes of HID devices: <ul style="list-style-type: none"> <li>▪ Absolute target position</li> <li>▪ Relative target position (specification of how many motions are to be executed by the respective same distance)</li> </ul> Sets the distance to be travelled for relative motions that are triggered by HID devices.  Motion commands are not allowed when the HID control is enabled for the axis.

#### INFORMATION

Absolute target positions can be commanded only if the reference point definition for the axis has been performed before; see "Reference Point Definition" (p. 37).

### Motions during open-loop operation

Motions are triggered by the following commands:

Commands	Description
STE	<p>Starts a motion of a specified number of steps in step mode (p. 20) and records the step response</p> <p>The step frequency is determined by the value of the parameter 0x1F000400.</p>
SMO	<p>Directly defines the control value for the PIShift drive electronics in the E-871.</p> <p>The value of the parameter 0x1F000702 determines in which PIShift drive mode the motion takes place:</p> <ul style="list-style-type: none"> <li>▪ 0x1F000702 has the value 0: The motion takes place in step mode. The control value set with SMO determines the step frequency and thus the velocity of the axis. The travel range is only limited by the physical limits of the stage.</li> <li>▪ 0x1F000702 has the value 1: The motion takes place in linear mode (p. 20). The control value set with SMO defines the output piezo voltage and thus the expansion of the piezo actuator in the PIShift drive. The travel range is limited by the maximum expansion of the piezo actuator.</li> </ul>

HID control is not possible in open-loop operation.

### 3.5.7 Servo Algorithm and Other Control Value Corrections

In closed-loop operation, the control value for the PIShift drive electronics integrated in the E-871 and thus the settling behavior of the system is optimized by a PID servo algorithm (**p**roportional **i**ntegral **d**ifferential). Independent of the servo mode, the control value is also corrected by a notch filter in linear mode.

The motion is executed in closed-loop operation as follows:

Target position can be reached in linear mode?	Sequence of the motion
No	Sequence of the following four phases: 1. Linear mode 2. Fast step mode 3. Slow individual steps 4. Linear mode The control value correction via the PID servo algorithm and the notch filter is only carried out during linear mode in phase 4 of the motion.
Yes	Linear mode with control value correction via PID servo algorithm and notch filter

#### Settings for the 4-phase motion sequence

The motion sequence consisting of linear and step mode carried out in closed-loop operation can be configured with the following parameters:

Parameter	Description and possible values
<b>PIShift Step Size</b> <b>(Phys. Unit)</b> 0x1F000700	Size of the slow individual steps Also serves as a criterion for switching between the following phases of the motion sequence: <ul style="list-style-type: none"> <li>▪ Fast step mode &gt; slow individual steps</li> <li>▪ Slow individual steps &gt; linear mode</li> </ul>



Parameter	Description and possible values
<b>PIShift Delay (ms)</b> 0x1F000701	Delay time Specifies the following: <ul style="list-style-type: none"> <li>Length of time between the end of the linear mode in phase 1 and the beginning of fast step mode</li> <li>Length of time between the end of the fast step mode and the beginning of the slow individual steps</li> <li>Length of time between slow individual steps</li> <li>Length of time between the last slow individual step and the beginning of linear mode</li> </ul>

### Settings for the servo algorithm

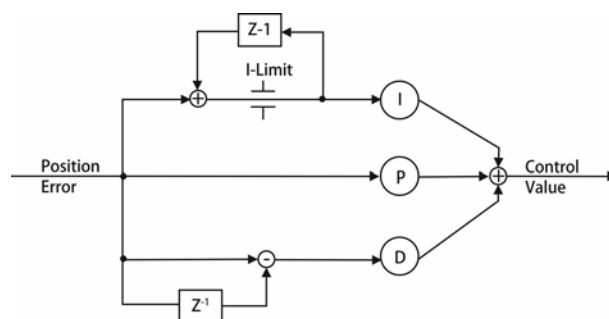


Figure 4: PID algorithm

The position error that runs through the PID servo algorithm results from the difference between the target position and the actual position (sensor feedback).

The servo algorithm uses the following servo-control parameters. The optimum servo-control parameter setting depends on your application and your requirements; see "Optimizing Servo-Control Parameters" (p. 94).

Parameter	Description and possible values
<b>P-Term</b> 0x1	Proportional constant (dimensionless) 0 to 32767 Aim: Rapid correction of the position error
<b>I-Term</b> 0x2	Integration constant (dimensionless) 0 to 32767 Aim: Reduction of the static position error

Parameter	Description and possible values
<b>D-Term</b> 0x3	Differential constant (dimensionless) 0 to 32767 The default setting of this parameter value is 0 and should not be changed.
<b>I-Limit</b> 0x4	Limit of the integration constant (dimensionless) 0 to 32767
<b>D-Term Delay (No. Of Servo Cycles)</b> 0x71	D-term delay The default setting of this parameter value is 0 and should not be changed.

The input of the servo algorithm can be configured for the E-871 with the following parameters:

Parameter	Description and possible values
<b>Numerator Of The Servo-Loop Input Factor</b> 0x5A	Numerator and denominator of the servo loop input factor 1 to 1,000,000 for both parameters The servo loop input factor decouples the servo-control parameters from the encoder resolution.
<b>Denominator Of The Servo-Loop Input Factor</b> 0x5B	The servo loop input factor is independent of the factor for counts per physical length unit (0xE and 0xF). Numerator and denominator of the servo loop input factor should not be changed.

### Settings for the notch filter

The notch filter corrects the control value in linear mode. The corrections by the notch filter take place in closed-loop and open-loop operation. The notch filter can be configured with the following parameters:

Parameter	Description and possible values
<b>Notch Filter Frequency 1 (Hz)</b> 0x94	Frequency of the first notch filter 40 to 20000 Hz The appropriate frequency component is reduced in the control value to compensate for undesired resonances in the mechanical system. An adjustment can be particularly useful in the case of very high loads.

Parameter	Description and possible values
<b>Notch Filter Edge 1</b> 0x95	Rise of the edge of the first notch filter (dimensionless) 0.1 to 10  The greater the value of this parameter, the narrower the notch filter bandwidth.

### 3.5.8 On-Target State

In closed-loop operation, the on-target state can be used to check whether the target position has been reached:

- On-target state = true (1): the target position is considered as reached
- On-target state = false (0): the target position is considered as not reached

The E-871 determines the on-target state based on the following criteria:

- Settling window around the target position (parameter 0x36)
- Delay time for setting the on-target state (parameter 0x3F)

The on-target state has the value **true** in the following cases:

- The current position is inside the settling window and stays there at least for the duration of the delay time.
- If the value for the delay time is set to 0: The current position is in the settling window.

The on-target state can be read with the `ONT?`, `#4` and `SRG?` commands.

In the *On Target* trigger mode (p. 106) the on-target state of the selected axis is output at the selected trigger output.

Parameter	Description and possible values
<b>Settling Time(s)</b> 0x3F	Delay time for setting the on-target state 0 to 1.000 s
<b>Settling Window (encoder counts)</b> 0x36	Settling window around the target position 0 to $2^{31}$ counts of the incremental encoder  Specifies the window limits. If the current position exits the settling window, the target position is no longer considered as reached.  The parameter value corresponds to half the width of the window. It can be changed only if the servo mode is switched off.

### 3.5.9 Reference Point Switch Detection

The E-871 receives signals from reference point switches on the following lines:

- **Sensor** panel plug (p. 309), pin 1: reference point switch, single-ended
- **Sensor** panel plug, pins 3 and 8: reference point switch, differential
- **Motor** socket (p. 308), pin 10: reference point switch, single-ended

The reference point switch detection by the E-871 can be configured with the following parameters:

Parameter	Description and possible values
<b><i>Invert Reference?</i></b> 0x31	Should the reference signal be inverted? 0 = Reference signal not inverted 1 = Reference signal inverted This parameter is used for inverting the reference signal whose source can be either the reference point switch or a digital input which is used instead of the reference switch (p. 114).
<b><i>Has Reference?</i></b> 0x14	Does the stage have a reference point switch? 0 = No reference point switch installed 1 = Reference point switch present (signal input at <b>Sensor</b> or <b>Motor</b> connection) This parameter enables or disables reference moves to the reference point switch installed.
<b><i>Reference Signal Type</i></b> 0x70	Reference signal type 0 = Direction-sensing reference point switch. The signal level changes when passing the reference point switch. 1 = Pulse signal with a pulse width of several nanoseconds (parameter 0x47 must be set correctly). 2 = Index pulse. The approach takes place via the negative limit switch or hard stop (default setting). 3 = Index pulse. The approach takes place via the positive limit switch or hard stop (default setting).

The signal from the reference point switch of the stage can be used for reference moves. After a reference move to the reference point switch, the controller knows the absolute axis position; see "Reference Point Definition" (p. 37).

**INFORMATION**

The stages with PIShift inertia drives currently offered by PI miCos supply the signal of the reference point switch on pins 3 and 8 of the **Sensor** panel plug. If this signal is to be used for reference moves, the value of the **Reference Signal Type** parameter (ID 0x70) must be set to 2 or 3.

### 3.5.10 Limit Switch Detection

The E-871 receives limit switch signals at the following connections:

- Positive limit switch: pin 6 of the **Sensor** panel plug (p. 309), pin 5 of the **Motor** socket (p. 308)
- Negative limit switch: pin 7 of the **Sensor** panel plug, pin 4 of the **Motor** socket

The limit switch detection by the E-871 can be configured with the following parameters:

Parameter	Description and possible values
<b>Limit Mode</b> 0x18	Signal logic of the limit switches 0 = Positive limit switch active high (pos-HI), negative limit switch active high (neg-HI) 1 = Positive limit switch active low (pos-LO), neg-HI 2 = pos-HI, neg-LO 3 = pos-LO, neg-LO
<b>Has No Limit Switches?</b> 0x32	Does the stage have limit switches? 0 = Stage has limit switches (signal inputs at the <b>Sensor</b> and <b>Motor</b> connections) 1 = Stage has no limit switches This parameter enables or disables a stop of the motion at the limit switches installed.
<b>Use Limit Switches Only For Reference Moves?</b> 0x77	Should the limit switches only be used for reference moves? 0 = Use limit switches for stopping at the end of the travel range and for reference moves (default) 1 = Use limit switches only for reference moves This parameter is intended for use with rotary stages. This parameter is only evaluated when the parameter 0x32 has the value 0.

The signals from the limit switches (also end-of-travel sensors) of a linear positioning stage are used to stop the motion prior to the hard stop at both ends of the travel range. In addition, soft limits (p. 32) can be set via parameters of the E-871.

The limit switch signals can also be used for reference moves. After a reference move to a limit switch, the controller knows the absolute axis position; see "Reference Point Definition" (p. 37).

### 3.5.11 Travel Range and Soft Limits

The physical limits of the travel range can be represented by the following items of the stage:

- Limit switches
- If the stage does not have any integrated limit switches: hard stops

The following parameters of the E-871 reflect the physical travel range of the stage and define soft limits:

Parameter	Description and possible values
<b>Maximum Travel In Positive Direction (Phys. Unit)</b> 0x15	Soft limit in positive direction (physical unit) Based on the zero position. If this value is smaller than the position value for the positive physical limit of the travel range (which results from the sum of the parameters 0x16 and 0x2F), the positive physical limit of the travel range cannot be used for reference moves. The value can be negative.
<b>Value At Reference Position (Phys. Unit)</b> 0x16	Position value at the reference point switch (physical unit) The current position is set to this value if the axis has executed a reference move to the reference point switch (start with <b>FRF</b> ). The parameter value is also used for calculating the position values which are set after reference moves to the physical limits of the travel range; this also applies when the mechanical system does not have a reference point switch.
<b>Distance From Negative Limit To Reference Position (Phys. Unit)</b> 0x17	Distance between reference point switch and negative physical limit of the travel range (physical unit) If the axis has executed a reference move to the negative physical limit of the travel range (start with <b>FNL</b> ), the current position is set to the difference between the values of parameters 0x16 and 0x17.

Parameter	Description and possible values
<b>Distance From Reference Position To Positive Limit (Phys. Unit)</b> 0x2F	Distance between reference point switch and positive physical limit of the travel range (physical unit)  If the axis has executed a reference move to the positive physical limit of the travel range (start with <b>FPL</b> ), the current position is set to the sum of the values of parameters 0x16 and 0x2F.
<b>Maximum Travel In Negative Direction (Phys. Unit)</b> 0x30	Soft limit in negative direction (physical unit) Based on the zero position. If this value is larger than the position value for the negative physical limit of the travel range (which results from the difference of the parameters 0x16 and 0x17), the negative physical limit of the travel range cannot be used for reference moves. The value can be negative.
<b>Range Limit Min</b> 0x07000000	Additional soft limit for the negative direction of motion (physical unit)  If the current position reaches this value in either closed-loop or open-loop operation, the control value is set to zero and the motion is stopped as a result. The axis can move again as soon as the value for the soft limit has been decreased.
<b>Range Limit Max</b> 0x07000001	Additional soft limit for the positive direction of motion (physical unit)  If the current position reaches this value in either closed-loop or open-loop operation, the control value is set to zero and the motion is stopped as a result. The axis can move again as soon as the value for the soft limit has been increased.

**INFORMATION**

The E-871 supports two parameter pairs for establishing soft limits. They are intended for different applications:

- 0x15 (**Maximum Travel In Positive Direction (Phys. Unit)**) and 0x30 (**Maximum Travel In Negative Direction (Phys. Unit)**):
  - The limits establish the permissible travel range in closed-loop operation.
  - Motion commands are executed only if the commanded position is within these soft limits.
  - The limits always refer to the current zero position.
  - Appropriate values are loaded when the stage type is selected from the stage database.
- 0x07000000 (**Range Limit Min**) and 0x07000001 (**Range Limit Max**):
  - Using these limits is recommended only if open-loop motions are required. Here, for logical reasons, the values are outside the soft limits which are specified via 0x15 and 0x30.
  - Apply both in closed-loop and open-loop operation.
  - Motions are stopped abruptly once the current position reaches a limit.
  - The limits are independent of the current zero position.
  - The values are not loaded from the stage database and are set in the default settings so that the limits are disabled.

**Examples**

The following examples refer to a stage with incremental sensor, reference point switch and limit switches.

The distance between the negative and positive limit switches of the stage is 20 mm. The reference point switch has a distance of 8 mm to the negative limit switch and a distance of 12 mm to the positive limit switch.

This switch setup of the stage is reflected in the following parameters:

- Parameter 0x17: Distance between negative limit switch and reference point switch = 8 mm
- Parameter 0x2F: Distance between reference point switch and positive limit switch = 12 mm

**INFORMATION**

The switch setup of the stage can be determined with the `FED` and `POS?` commands.



**Example 1: Maximum travel range available**

After reference moves (p. 37), the current position is to have the following values:

- Move to the negative limit switch (start with **FNL**): current position = 0
- Move to the reference point switch (start with **FRF**): current position = 8
- Move to the positive limit switch (start with **FPL**): current position = 20

As a result, parameter 0x16, which, during reference moves, specifies the position value for the reference point switch and is included in the calculation of the position values for the limit switches, has the value 8.

The travel range is not to be limited by soft limits. As a result, the respective parameters are set as follows:

- Parameter 0x15 = 20
- Parameter 0x30 = 0

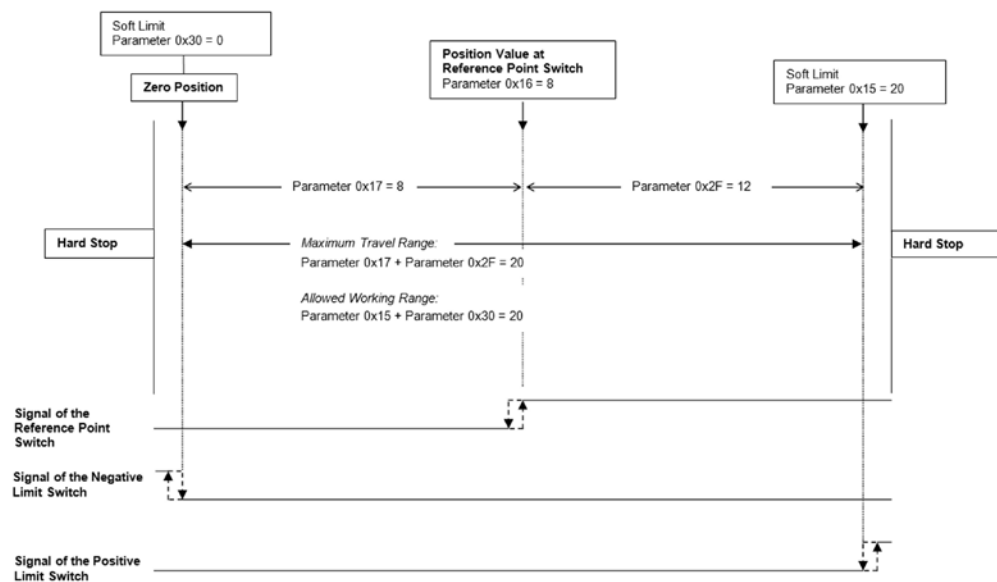


Figure 5: The travel range of the stage is not limited by soft limits.

After a reference move of the stage to the reference point switch (**FRF** command), query commands return the following responses:

- **TMN?** returns the value 0
- **TMX?** returns the value 20
- **POS?** returns the value 8

### Example 2: Travel range limited by soft limits

The zero position should be located at approximately a third of the distance between the negative limit switch and the reference point switch. As a result, parameter 0x16 now has the value 5.4.

A safety distance is to be put in place at both ends of the travel range by establishing soft limits. As a result, the soft limits are set as follows:

- Parameter 0x15 = 16.4
- Parameter 0x30 = -2.1

As a result, the stage can move from the zero position 16.4 mm in the positive direction and 2.1 mm in the negative direction, respectively. The limit switches can no longer be used for reference moves.

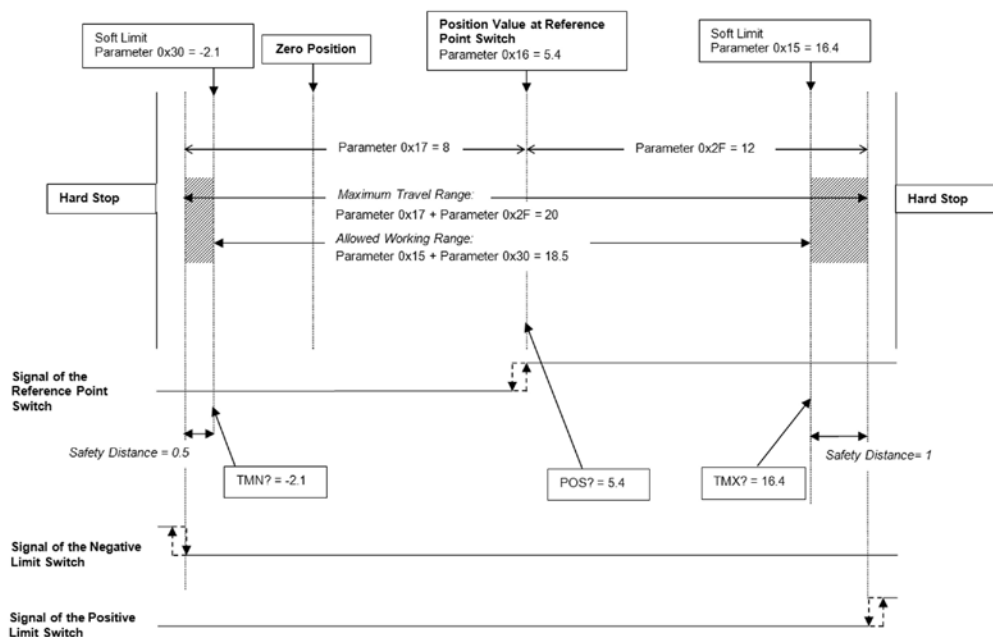


Figure 6: The travel range of the stage is limited by soft limits.

After a reference move of the stage to the reference point switch (**FRF** command), query commands return the following responses:

- **TMN?** returns the value -2.1
- **TMX?** returns the value 16.4
- **POS?** returns the value 5.4

### 3.5.12 Reference Point Definition

The incremental sensors that are used for axis position feedback only return relative motion information. As a result, the controller does not know the absolute position of an axis upon switching-on. So that absolute target positions can be commanded and reached, a reference point definition must be performed beforehand.

The reference point definition can be performed in different ways:

- **Reference move** (default): A reference move moves the axis to a permanently defined point, e.g. to the reference point switch, to a limit switch or to a hard stop. At this point, the current position is set to a defined value. The controller now knows the absolute axis position.
- **Manually setting the absolute position:** If this type of reference point definition was enabled with the `RON` command (p. 229), you can set the current position of the axis at an arbitrary point to an arbitrary value using the `POS` command (p. 227). Here the axis is not moved. The controller knows the absolute axis position afterwards.

#### INFORMATION

During start-up using PIMikroMove, the reference point definition is performed via a reference move by default. Knowledge of the commands and parameters described here is not needed for reference point definition using PIMikroMove.

### Commands

The following commands are available for reference point definition:

Command	Syntax	Function
<code>RON</code>	<code>RON {&lt;AxisID&gt; &lt;ReferenceOn&gt;}</code>	Sets mode of reference point definition: <ul style="list-style-type: none"> <li>▪ <code>&lt;ReferenceOn&gt; = 0</code>: To define the reference point of the axis, an absolute position value can be assigned with <code>POS</code> or a reference move can be started with <code>FRF</code>, <code>FNL</code> or <code>FPL</code>.</li> <li>▪ <code>&lt;ReferenceOn&gt; = 1</code> (default): To define the reference point of the axis, a reference move must be started with <code>FRF</code>, <code>FNL</code> or <code>FPL</code>. Using <code>POS</code> is not permitted.</li> </ul>

Command	Syntax	Function
RON?	RON? [{<AxisID>}]	Gets mode of reference point definition.
FRF	FRF [{<AxisID>}]	<p>Starts a reference move to the reference point switch. The approach depends on the value of the <b>Reference Signal Type</b> parameter (ID 0x70):</p> <ul style="list-style-type: none"> <li>0x70 = 0 or 1: The approach always takes place from the same side irrespective of the axis position when the command is sent.</li> <li>0x70 = 2: The approach takes place via the negative limit switch or hard stop.</li> <li>0x70 = 3: The approach takes place via the positive limit switch or hard stop.</li> </ul>
FRF?	FRF? [{<AxisID>}]	<p>Queries whether the reference point for an axis has already been defined.</p> <p>1 = Reference point has been defined 0 = Reference point has not been defined</p>
FNL	FNL [{<AxisID>}]	Starts a reference move to the negative limit switch or hard stop.
FPL	FPL [{<AxisID>}]	Starts a reference move to the positive limit switch or hard stop.
POS	POS {<AxisID> <Position>}	Sets the current position (does not trigger a motion) and thus defines the reference point.

## Parameters

Reference moves can be configured with the following parameters:

Parameter	Description and possible values
<b>Reference Travel Direction</b> 0x47	Default direction for the reference move 0 = automatic detection 1 = negative direction 2 = positive direction

Parameter	Description and possible values
<b>Distance Between Limit And Hard Stop (Phys. Unit)</b> 0x63	Distance between internal limit switch and hard stop Is used for stages without limit switches for reference moves that are started with FNL or FPL: Defines the distance that the axis moves away from the hard stop after reaching it. The reference move is not finished until this distance has been travelled.
<b>Distance From Limit To Start Of Ref. Search (Phys. Unit)</b> 0x78  <b>Distance For Reference Search (Phys. Unit)</b> 0x79	Distance between limit switch or hard stop and the starting position for the motion to the index pulse  Maximum distance for the motion to the index pulse  The parameters 0x78 and 0x79 are used for reference moves when the two following conditions are met: <ul style="list-style-type: none"> <li>▪ The reference move is started with FRF.</li> <li>▪ The <b>Reference Signal Type</b> parameter has the value 2 or 3.</li> </ul> Sequence of the reference move: <ol style="list-style-type: none"> <li>1. The axis moves to the corresponding limit switch or hard stop.</li> <li>2. The axis moves the distance given by the parameter 0x78 away from the limit switch or hard stop.</li> <li>3. The axis moves to the index pulse and thus travels the distance specified by the parameter 0x79 at the maximum.</li> </ol>
<b>Use Hard Stops For Referencing?</b> 0x7A	Should the hard stops be used for reference moves? 0 = Do not use for reference moves 1 = Use for reference moves

#### INFORMATION

- For maximum repeatability always execute the reference move in the same way.

**INFORMATION**

The limit switches or the hard stops can be used for reference moves only if the travel range is not limited by soft limits (p. 32).

**INFORMATION**

For reference moves, you can also use the digital inputs of the E-871 as the source of the reference signal, the negative limit switch signal or the positive limit switch signal. See "Using Digital Input Signals as Switch Signals" (p. 114) for more information.

**INFORMATION**

The stages with PIShift inertia drives currently offered by PI miCos supply the signal of the reference point switch on pins 3 and 8 of the **Sensor** panel plug. If this signal is to be used for reference moves, the value of the **Reference Signal Type** parameter (ID 0x70) must be set to 2 or 3. Details see "Reference Point Switch Detection" (p. 30).

**INFORMATION**

If the absolute position of the axis is defined manually with the POS command, conflicts with the settings for the soft limits can occur (parameter 0x15, query with `TMX?`, and 0x30, query with `TMN?`).

- Set the absolute position of the axis manually only if reference point definition is not possible otherwise.

**INFORMATION**

If the current parameter settings of the E-871 are written to the nonvolatile memory in PIMikroMove or by entering the WPA command using the password 100 or 101, the axis will no longer be considered "referenced" (the response to FRF? is 0).

### 3.5.13 ID Chip Detection

The stages with PIShift inertia drives offered from PI miCos contain an ID chip in the motor or sensor connection on which the following data is saved as parameters:

- Information on the stage: type, serial number, date of manufacture, version of the hardware
- Settings for the sensor: interpolation rate, corrections of hysteresis as well as of phase and offset, gain values

The data of the connected stage is loaded to the volatile memory of the E-871 from the ID chip when the E-871 is switched on (p. 75) or rebooted.

The parameter values in the volatile memory of the E-871 can be queried and written to the nonvolatile memory, see "Adapting Settings" (p. 271).

#### **INFORMATION**

The ID chip only contains some of the information that is required to operate the stage with the E-871. When you use the PC software from PI, further information is loaded as parameter values from a stage database (p. 44) into the volatile memory of the E-871.

Parameters that are loaded from the ID chip or from a stage database are marked in color in the parameter overview (p. 280).

## 3.6 Communication Interfaces

### Communication interfaces available

The E-871 can be controlled with ASCII commands from a PC: Connecting to the PC can be effected via a direct connection or via a daisy chain network. The following interfaces of the E-871 can be used for direct connection to the PC:

- Serial RS-232 connection
- USB connection

Only one of the two interfaces may be connected to the PC at all times.

### Default communication settings

Interface	Property	Default value
RS-232	Baud rate	115200 Settings for DIP switches 5 and 6; see "Baud Rate" (p. 74) Other: 8 data bits and 1 stop bit, without parity; internal buffers do not require a handshake

### Daisy chain network

Using a daisy chain network, up to 16 controllers can be connected to the PC via a single RS-232 or USB connection. Interlinking occurs in series. See also "Definition" (p. 3).



### 3.7 Overview of PC Software

The following table shows the PC software that is included in the product CD. The given operating systems stand for the following versions:

- Windows: Versions XP (Service Pack 3), Vista (Service Pack 1) and 7
- Linux: Kernel 2.6, GTK 2.0, glibc 2.4 (configuration used to develop the PC software)

PC software	Operating system	Short description	Recommended use
Dynamic program library for GCS	Windows, Linux (USB only in Windows)	Allows software programming for the E-871 with programming languages such as e. g. C++. The functions in the dynamic program library are based on the PI General Command Set (GCS).	For users who would like to use a dynamic program library for their application. Is required for PIMikroMove. Is required for LabVIEW drivers if communication is to be established via USB or a daisy chain network.
LabVIEW drivers	Windows, Linux	LabVIEW is a software for data acquisition and process control (must be ordered separately from National Instruments). The E-871 LabVIEW software is a collection of virtual instrument drivers (VI drivers) for the E-871 controller. These drivers support the GCS.	For users who want to use LabVIEW to program their application.
PIMikroMove	Windows	Graphic user interface for Windows with which the E-871 and other controllers from PI can be used: <ul style="list-style-type: none"> <li>▪ The system can be started without programming effort</li> <li>▪ Graph of motions in open-loop and closed-loop operation</li> <li>▪ Macro functionality for storing command sequences on the PC (host macros)</li> <li>▪ Support of HID devices</li> <li>▪ Complete environment for command entry, for trying out different commands</li> </ul>	For users who want to perform simple automation tasks or test their equipment before or instead of programming an application. A log window showing the commands sent makes it possible to learn how to use the commands.

PC software	Operating system	Short description	Recommended use
		No command knowledge is necessary to operate PIMikroMove. PIMikroMove uses the dynamic program library to supply commands to the controller.	
PI Terminal	Windows	Terminal program that can be used for nearly all PI controllers (see the description of the <b>Command Entry</b> window in the PIMikroMove user manual).	For users who want to send GCS commands directly to the controller.
PI Stage Editor	Windows	Program for opening and editing stage databases.	For users who want to deal intensively with the contents of stage databases.
PI Update Finder	Windows	Checks the PI software installed on the PC. If more current versions of the PC software are available on the PI server, downloading is offered.	For users who want to update the PC software.
PI Firmware Updater	Windows	Program for user support when updating firmware of the E-871.	For users who want to update the firmware.
USB driver	Windows	Driver for the USB interface	For users who want to connect the controller to the PC via the USB interface.

### 3.8 Stage Databases

You can select a parameter set appropriate for your stage from a stage database in the PC software from PI. The PC software transfers the values of the selected parameter set to the volatile memory of the controller.

Database file name	Editable?	Description
PI MicosStages2.dat	No, updates can be downloaded from the PI website (p. 56).	Standard stage database: Includes parameter sets for all standard stages from PI miCos; is automatically saved on the PC when the PC software is installed.
PI Stages2.dat	No, updates can be downloaded from the PI website (p. 56).	Standard stage database: Includes parameter sets for all standard stages from PI; is automatically saved to the PC when the PC software is installed.

Database file name	Editable?	Description
PI_UserStages2.dat	Yes, new parameter sets can be created, edited and saved (p. 277).	Is automatically created when you make a connection to your stage using the PC software for the first time (i. e. when selecting the stage in PIMikroMove or when using the commands VST? or CST from the dynamic program library).
X-xxx.dat	No, you receive updates from our customer service department (p. 301).	Includes the parameter set for a custom stage, for installation see "Installing a Custom Stage Database" (p. 59).

The parameter values in the volatile memory of the E-871 can be queried and written to the nonvolatile memory, see "Adapting Settings" (p. 271).

#### **INFORMATION**

Stage databases only contain some of the information that is required to operate a stage with the E-871. Further information is loaded as parameter values to the volatile memory of the E-871 from the ID chip (p. 41) of the stage when the E-871 is switched on or rebooted.

Parameters that are loaded from a stage database or from the ID chip are marked in color in the parameter overview (p. 280).

Further information on stage databases can also be found in the user manuals for PIMikroMove, PIStageEditor and the PI GCS program library.



## 4 Unpacking

1. Unpack the E-871 with care.
2. Compare the contents against the items covered by the contract and against the packing list.
3. Inspect the contents for signs of damage. If parts are missing or you notice signs of damage, contact PI immediately.
4. Keep all packaging materials in case the product needs to be returned.



## 5 Quick Start

### CAUTION



#### **Risk of electric shock if the protective earth conductor is not connected!**

If a protective earth conductor is not or not properly connected, dangerous touch voltages can occur on the E-871 in the case of malfunction or failure of the system. If touch voltages exist, touching the E-871 can result in minor injuries due to electric shock.

- Connect the E-871 to a protective earth conductor before start-up (p. 61).
- Do **not** remove the protective earth conductor during operation.
- If the protective earth conductor has to be removed temporarily (e. g. in the case of modifications), reconnect the E-871 to the protective earth conductor before starting it up again.

### NOTICE



#### **Incorrect wiring!**

Connecting the USB and RS-232 interfaces of the controller to the PC at the same time can damage the PC or the controller.

- Connect either the USB or the RS-232 interface to the PC.

**NOTICE****Oscillations!**

Unsuitable settings of the notch filter and the servo-control parameters of the E-871 can cause the stage to oscillate. Oscillations can damage the stage and/or the load affixed to it.

- Secure the stage and all loads adequately.
- If the stage is oscillating (unusual operating noise), immediately switch off the servo mode or disconnect the E-871 from the power source.
- Only switch on the servo mode after you have modified the settings of the notch filter and the servo-control parameters of the E-871; see „Setting the Notch Filter“ (p. 89) and "Optimizing Servo-Control Parameters" (p. 94).
- If, due to a very high load, oscillations occur already during the reference move, follow the instructions for the reference move in "Troubleshooting" (p. 297).

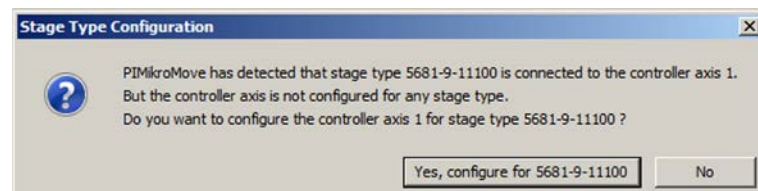
The aim of quick start is to start initial test motions of a stage that is connected to a non-networked E-871 in the PIMikroMove PC software.

1. Install the following on the PC:
  - The PC software and the USB drivers from the product CD
  - Updates for PC software and the stage databases (PIMicosStages2.dat and PISTages2.dat).
  - If provided separately by PI: custom stage database(s)
 Details see "Installing the PC Software" (p. 55).
2. Install the E-871:
  - Observe the general information on installation (p. 55).
  - Ensure the ventilation (p. 59).
3. Connect the E-871 to the protective earth conductor (p. 61) via the threaded bolt marked with .
4. Connect the following to the E-871:
  - The included wide-range-input power supply (**not** connected to the power socket via the power cord) to the **24 V** **2.5 A** connection. Details see "Connecting the Power Supply to the E-871" (p. 62).
  - The stage to the **Motor** socket and, if a sensor connection is present, to the **Sensor** panel plug. Details see "Connecting the Stage" (p. 63).



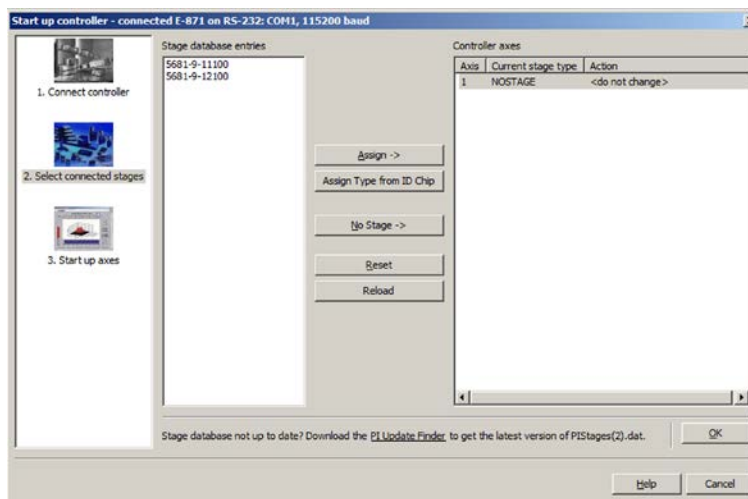
- The PC via the RS-232 interface (**RS-232 In** socket) **or** via the USB interface. Details see "Connecting the PC" (p. 63).
- 5. Check the DIP switch settings (p. 72) and adapt them to your application if necessary. Controller address 1 must be set.
- 6. Switch on the E-871 (p. 75) by connecting the power cord of the wide-range-input power supply to the power socket.
- 7. Start PIMikroMove on the PC.
- 8. Establish communication between the E-871 and the PC in PIMikroMove via the RS-232 interface or the USB interface. Details see "Establishing Communication" (p. 76).
- 9. If one of the two following points applies, configure the E-871 for the connected stage:
  - The **Stage Type Configuration** window has opened.
  - In the **Start up controller** window, the **Select connected stages** step is displayed.

If the **Stage Type Configuration** window has opened:

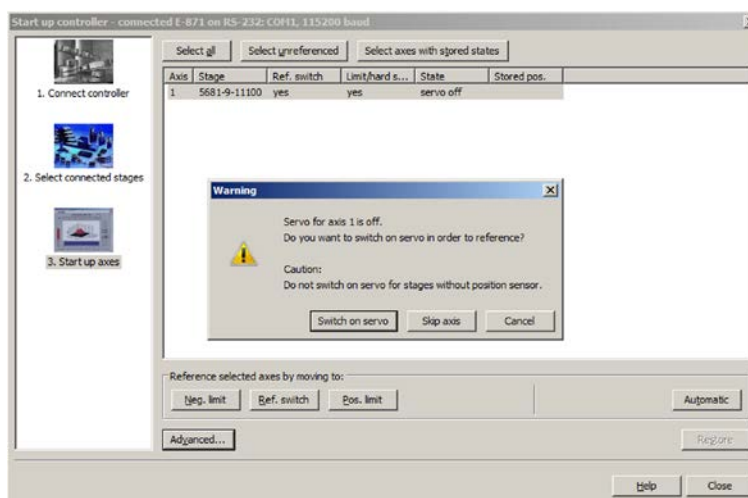


- Click the **Yes, configure for ...** button to load the appropriate parameter set from a stage database to the volatile memory of the E-871. The **Stage Type Configuration** window closes, and the **Start up controller** window goes to the **Start up axes** step.

If the **Select connected stages** step is displayed in the **Start up controller** window:



- a) Select the appropriate stage type: Click **Assign Type from ID Chip** or mark the appropriate stage type in the **Stage database entries** list.
  - b) If you have marked the appropriate stage type in the **Stage database entries** list in step a, click **Assign**.
  - c) Confirm the selection with **OK** to load the parameter settings for the selected stage type from the stage database to the volatile memory of the E-871. The **Start up controller** window goes to the **Start up axes** step.
10. In the **Start up axes** step, execute the reference move for the axis so that the controller knows the absolute axis position:



- If you want to start the reference move to the reference point switch, click on **Ref. switch**.

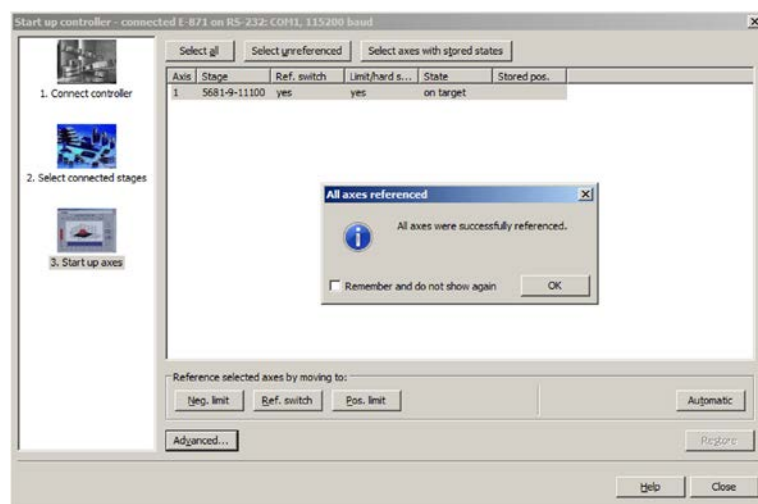
- If you want to start the reference move to the negative physical limit of the travel range, click **Neg. limit**.
- If you want to start the reference move to the positive physical limit of the travel range, click **Pos. limit**.

If a warning message appears indicating that the servo mode is switched off:

- Switch on the servo mode by clicking on the **Switch on servo** button (closed-loop operation).

The axis executes the reference move.

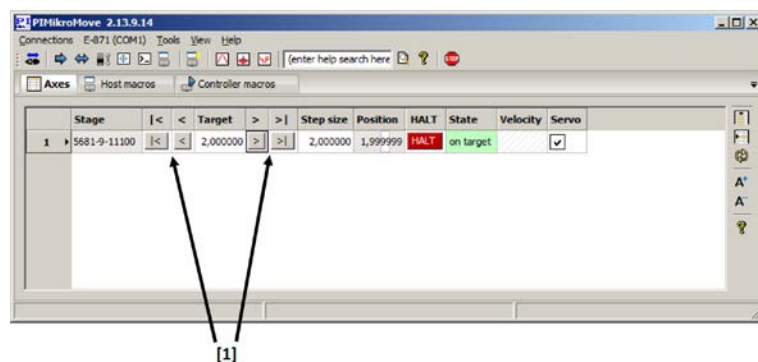
11. After a successful reference move, click **OK > Close**.



The main window of PIMikroMove opens.

12. Start a few test motions of the axis.

In the main window of PIMikroMove, you can execute, for example, motions of a particular distance (specification in **Step size** column) or to the limits of the travel range by clicking the corresponding arrow keys [1] for the axis.





## 6 Installation

### In this Chapter

General Notes on Installation .....	55
Installing the PC Software .....	55
Ensuring Ventilation.....	59
Mounting the E-871 .....	60
Connecting the E-871 to a Protective Earth Conductor .....	61
Connecting the Power Supply to the E-871 .....	62
Connecting the Stage .....	63
Connecting the PC .....	63
Connecting an HID Device .....	66
Connecting Digital In- and Outputs .....	68
Connecting Analog Signal Sources.....	70

### 6.1 General Notes on Installation

- Install the E-871 near the power source so that the power plug can be quickly and easily disconnected from the mains.
- Only use cables and connections that meet local safety regulations.

### 6.2 Installing the PC Software

Communication between the E-871 and a PC is required to configure the E-871 and to command motions using the commands of the GCS. Various PC software applications are available for this purpose.

#### 6.2.1 Performing the Initial Installation

##### Accessories

- PC with a Windows operating system (XP, Vista, 7) or Linux operating system
- Product CD (included in the scope of delivery)

##### Installing the PC software on Windows

1. Start the installation wizard by double-clicking the **PI** icon or the **PI\_Setup.exe** file in the installation directory (root directory of the CD).

The **InstallShield Wizard** window for the installation of programs and manuals for the E-871 opens.

2. Follow the instructions on the screen.

You can choose between default installation (*Complete*) and user-defined installation (*Custom*).

With default installation (recommended), all components are installed. These include among others:

- LabVIEW drivers
- Dynamic program library for GCS
- PIMikroMove
- PI Firmware Updater

With user-defined installation, you have the option of excluding individual components from the installation.

When the installation of the components has finished, the **InstallShield Wizard** window for the installation of the USB drivers required for the E-871 opens after a short time.

3. Follow the instructions on the screen for installing USB drivers.

### Installing the PC software on Linux

1. Unpack the tar archive from the /linux directory of the product CD to a directory on your PC.
2. Open a terminal and go to the directory to which you have unpacked the tar archive.
3. Log on as a superuser (root rights).
4. Enter `./INSTALL` to start the installation.  
Pay attention to lower and upper case when entering commands.
5. Follow the instructions on the screen.

You can select individual components for installation.

### 6.2.2 Installing Updates

PI is constantly improving the PC software.

- Always install the latest version of the PC software and the PISTages2.dat and PIMicosStages2.dat stage databases.

## Prerequisite

- ✓ Active connection to the Internet.
- ✓ If your PC uses a Windows operating system:
  - If the PI Update Finder program is not on your product CD:  
You have downloaded the PI Update Finder from our Update Portal (<http://www.update.pi-portal.ws>).
  - You have the A000T0028 Technical Note for the PI Update Finder on hand. You can find the document either on the product CD or in the zip file that you have downloaded for the PI Update Finder.
  - If the PC to be updated is **not** directly connected to the Internet:  
You have Technical Note A000T0032 for the PI Update Finder at hand. You can find the document either on the product CD or in the zip file that you have downloaded for the PI Update Finder.
- ✓ If your PC uses a Linux operating system:
  - You have the user name and password for the E-871 at hand. Both of these can be found in the file "E-871 Releasenews\_V\_x\_x\_x.pdf" (x\_x\_x: version number of the CD) in the \Manuals folder on the product CD.

## Updating PC-Software, PIStages2.dat and PIMicosStages2.dat in Windows

- Use the PI Update Finder:
  - When the PC to be updated is directly connected to the Internet: Follow the instructions in the A000T0028 Technical Note (TECHNICAL\_NOTE\_PI\_UPDATE\_FINDER\_xx.pdf).
  - When the PC to be updated is **not** directly connected to the Internet: Follow the instructions in the A000T0032 Technical Note.

## Updating the PC software on Linux

1. Open the PI website (<http://www.pi-portal.ws>).
2. Click **Downloads**.
3. In the **User login** area on the left margin, enter the user name and password from the "E-871\_Releasenews\_V\_x\_x\_x.pdf" file on the product CD.
4. Click **Login**.
5. Click the category **E Piezo Drivers & Nanopositioning Controllers**.

6. Click **E-871 > Software** (if you click **Documents**, the latest versions of the corresponding manuals are displayed).
7. Underneath the latest CD mirror, click the **Download** button (also contains the manuals).
8. Save the downloaded archive file on the PC.
9. Unpack the file to a separate installation directory.
10. In the directory with the unpacked files, go to the linux subdirectory.
11. Unpack the archive file in the linux directory by entering the command `tar -xvpf <name of the archive file>` on the console.
12. Read the accompanying information (readme file) on the software update.
13. Log onto the PC as a superuser (root rights).
14. Only install the update if it is a good idea for your application.

### Updating PISTages2.dat and PIMicosStages2.dat on Linux

1. Open the PI website (<http://www.pi-portal.ws>).
2. Click **Downloads**.
3. In the **User login** area on the left margin, enter the user name and password from the "E-871\_Releasenews\_V\_x\_x\_x.pdf" file on the product CD.
4. Click **Login**.
5. Click the **General Software** category.
6. Click on **PI Stages**.
7. Click the name of the stage database - **pistages2** or **pimicosstages2** - or the **Download** button below it.
8. Log onto the PC as a superuser (root rights).
9. Install the downloaded file - **pistages2.dat** or **pimicosstages2.dat** - on the PC. You can select between the following options:
  - Save the file in the `/usr/local/PI/pi_gcs_translator/` directory.
  - Save the file in the directory where you unpacked the Linux software from the product CD. The path is `<unpacking directory>/pi_stages2_dat` or `<unpacking directory>/pimicosstages2_dat`. In this subdirectory start the `INSTALL.pi_stages2_dat` or `INSTALL.pimicosstages2_dat` script.



### 6.2.3 Installing a Custom Stage Database

With a custom stage, you will receive, if necessary, a file from PI with a custom stage database. You have to install this file on your PC so that you can load the parameter values for the custom stage in the E-871.

#### Installing a custom stage database on Windows

1. Open the \PI\GCSTranslator directory on your PC:

If you are working with PIMikroMove:

- a) From the main window of PIMikroMove open via the **Connections > Search for controller software** menu item the **Version Information** window.
- b) In the **Version Information** window, click on the **Show GCS PATH...** button to open the \PI\GCSTranslator directory in Windows Explorer.

The path where the \PI directory is located was defined during the installation of the PC software, normally C:\Documents and Settings\All Users\Application Data (Windows XP) or C:\ProgramData (Windows Vista, Windows 7).

2. Copy the stage database file to the \PI\GCSTranslator directory on your PC.

#### INFORMATION

If the \PI\GCSTranslator directory is not present on your PC:

For an executable file (.exe) to be able to access a stage database, both files have to be in the same directory.

#### Installing a custom stage database on Linux

1. Log onto the PC as a superuser (root rights).
2. Copy the stage database file to the /usr/local/PI/pi\_gcs\_translator/ directory.

## 6.3 Ensuring Ventilation

High temperatures can overheat the E-871.

- Set up the E-871 with a distance of at least 10 cm to the top and rear sides and at least 5 cm to the sides. If this is not possible, make sure that the area is cooled sufficiently.
- Ensure sufficient ventilation at the place of installation.
- Keep the ambient temperature to a non-critical level (<50° C).

## 6.4 Mounting the E-871

The E-871 can be used as a bench-top device or mounted in any orientation on a surface.

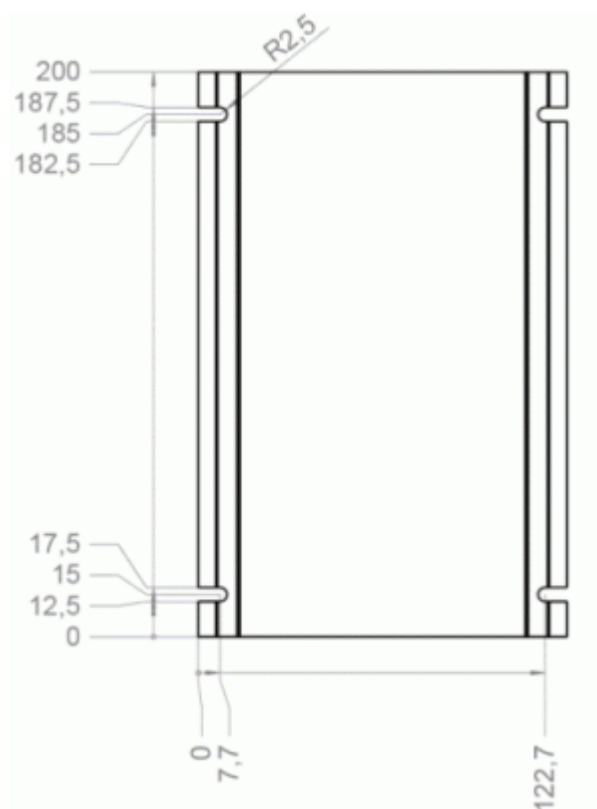


Figure 7: E-871 mounting rails with recesses

### Tools and accessories

- Suitable screws
- Suitable screwdriver

### Mounting the E-871

1. Make the necessary holes in the surface.

The arrangement of the recesses in the mounting rails of the E-871 can be found in the figure.

2. Mount the E-871 in the recesses in the mounting rails with two suitable screws on each side.

## 6.5 Connecting the E-871 to the Protective Earth Conductor

### INFORMATION

- Observe the applicable standards for mounting the protective earth conductor.


### Prerequisite

- ✓ You have read and understood the General Notes on Installation (p. 55).
- ✓ The E-871 is switched off, i. e. the power supply is **not** connected to the power socket via the power cord.

### Tools and accessories

- Suitable protective earth conductor:
  - Cable cross-section  $\geq 0.75 \text{ mm}^2$
  - Contact resistance  $< 0.1 \text{ ohm}$  at 25 A at all connection points relevant for mounting the protective earth conductor
- Fastening material for the protective earth conductor, sits on the protective earth connector (threaded bolt) in the following order upon delivery of the E-871, starting from the case:
  - Safety washer
  - Nut
  - Flat washer
  - Toothed washer
  - Nut
- Suitable wrench

### Connecting the E-871 to the protective earth conductor

1. If necessary, fasten a suitable cable lug to the protective earth conductor.
2. Remove the outer nut from the protective earth connector on the rear panel of the E-871 (threaded bolt marked with  (p. 10)).

3. Connect the protective earth conductor:
  - a) Push the cable lug of the protective earth conductor onto the threaded bolt.
  - b) Screw the nut onto the threaded bolt. In this way, the cable lug of the protective earth conductor is wedged between the toothed washer and the nut.
  - c) Tighten the nut with at least three rotations and a torque of 1.2 Nm to 1.5 Nm.

## 6.6 Connecting the Power Supply to the E-871


### Prerequisites

- ✓ The power cord is **not** connected to the power socket.

### Tools and accessories

- The included 24 V wide-range-input power supply (for line voltages between 100 and 240 volts alternating current voltage at 50 or 60 Hz)
- Alternatively: sufficiently dimensioned power supply that provides 24 volts direct current voltage with a maximum of 2.0 ampere.
- Supplied adapter for the power supply connector; barrel connector to M8 4-pin connector, A-coded
- Alternatively: sufficiently dimensioned adapter
- Supplied power cord
- Alternatively: sufficiently dimensioned power cord

### Connecting the power supply to the E-871

- Connect the M8 connector of the adapter to the **24 V**  **2.5 A** connection of the E-871.
- Connect the barrel connector of the adapter to the barrel connector socket of the power supply.
- Connect the power cord to the power supply.

## 6.7 Connecting the Stage

### Prerequisite

- ✓ The E-871 is switched off, i. e. the power supply is **not** connected to the power socket via the power cord.

### Connecting the stage

- Connect the motor connection of the stage to the **Motor** socket of the E-871.
- If present: connect the sensor connection of the stage to the **Sensor** panel plug of the E-871.

## 6.8 Connecting the PC

The communication between the E-871 and a PC is necessary to configure the E-871 and send motion commands using the commands of the GCS. The E-871 has the following interfaces for this purpose:

- RS-232 interface
- USB interface

In this section, you will learn how to make the proper cable connections between the E-871 and a PC as well as in a daisy chain. All other steps required for establishing communication between the E-871 and PC are described in the following sections:

- "Establishing Communication via RS-232" (p. 76)
- "Establishing Communication via USB" (p. 77)
- "Establishing Communication for Networked Controllers" (p. 79)

### INFORMATION

Using a daisy chain network, up to 16 controllers can be connected to the PC via a single RS-232 or USB connection.

## 6.8.1 Connecting to the RS-232 Interface

### NOTICE

**Incorrect wiring!**

Connecting the USB and RS-232 interfaces of the controller to the PC at the same time can damage the PC or the controller.

- Connect either the USB or the RS-232 interface to the PC.

### Prerequisites

- ✓ The PC has a free RS-232 interface (also called a "serial interface" or "COM port", e. g. COM1 or COM2).

### Tools and accessories

- RS-232 null-modem cable (C-815.34 included in the scope of delivery)

### Connecting the E-871 to the PC

- Connect the **RS-232 In** socket on the front panel of the E-871 and the RS-232 interface of the PC (a Sub-D 9(m) panel plug) to the null-modem cable.

## 6.8.2 Connecting to the USB Interface

### NOTICE

**Incorrect wiring!**

Connecting the USB and RS-232 interfaces of the controller to the PC at the same time can damage the PC or the controller.

- Connect either the USB or the RS-232 interface to the PC.

### Prerequisites

- ✓ The PC has a free USB interface.

### Tools and accessories

- USB cable (type A to Mini-B) for connecting to the PC (order number 000014651; included in the scope of delivery)

### Connecting the E-871 to the PC

- Connect the USB socket of the E-871 and the USB interface of the PC with the USB cable.

## 6.8.3 Setting Up a Daisy Chain Network

### INFORMATION

Interlinking in a daisy chain occurs in series. See also "Definition" (p. 3). Here the first controller is connected directly to the PC.

### INFORMATION

The DIP switches of the E-871 must be set accordingly:

- Set a unique address for each controller in a daisy chain network. In doing so, one of the controllers must have the address 1. This controller does not have to be the one directly connected to the PC. Details see "Controller Address" (p. 73).
- Set the same baud rate for every controller in a daisy chain network. Details see "Baud Rate" (p. 74).

### Tools and accessories

- A network cable for every controller to be connected to the network. Currently available:
  - C-862.CN, 30 cm, included in the scope of delivery
  - C-862.CN2, 180 cm, available as an optional accessory (p. 13)

### Interlinking the controllers

- Set up the controller chain. For this purpose, always connect the **RS-232 Out** connection of the previous controller via the network cable to the **RS-232 In** connection of the subsequent controller.
- Connect the first controller of the chain to the PC.
  - Use the RS-232 interface (p. 64).

or

  - Use the USB interface (p. 64).

**INFORMATION**

The E-871 can be operated in a common daisy chain with the following controllers:

- C-863.11 Mercury DC motor controller
- C-663.11 Mercury Step stepper motor controller
- PLine® piezomotor controller of the C-867 series
- E-861 NEXACT® controller

## 6.9 Connecting an HID Device

**INFORMATION**

A total of 4 axes of an HID device can be connected to the **Joystick** (p. 311) and **I/O** (p. 310) sockets of the E-871. The axes of the HID device are suitable for controlling the following motion parameters of the stage axis connected to the E-871:

- Axes 1 and 2: absolute target position
- Axes 3 and 4: relative target position

Connection options at the **Joystick** socket (p. 311):

- Axis 1: pin 4 (0 to 3.3 V)
- Axis 2: pin 2 (-10 to 10 V)

Connection options at the **I/O** socket (p. 310):

- Axis 3: pins 1 and 2 (TTL signals)
- Axis 4: pins 3 and 4 (TTL signals)

The two buttons of the HID device can be connected at the **Joystick** socket (p. 311).

Connection options:

- Button 1: pin 5 (0 or 3.3 V)
- Button 2: pin 6 (0 or 3.3 V)

Further information see "Control with an HID Device" (p. 118).



**INFORMATION**

The C-819.20 and C-819.30 joysticks, available as optional accessories, use pins 4, 5 and 6 of the **Joystick** socket. Pin 3 of this socket is used as power source of the joystick.

You can use a C-819.20Y Y cable to connect two E-871s to a C-819.20 joystick. In this case, the joystick is powered by the E-871, which is connected to the X branch of the cable.

**Tools and accessories**

If the relative target position of the axis of the E-871 is to be controlled with the HID device:

- Rotary encoder or pulse generator for manual operation, type of output signals: AB, maximum 500 Hz, TTL

If the absolute target position of the axis of the E-871 is to be controlled with the HID device:

- Joystick from PI for operation with 0 to 3.3 V, available as an optional accessory (p. 13):
  - C-819.20 analog joystick for 2 axes
  - If a C-819.20 joystick is to be connected to two controllers: C-819.20Y Y cable
- or
- C-819.30 analog joystick for 3 axes
- Alternative: analog signal source that supplies -10 to 10 V

**Connecting an HID device**

- If you want to use axis 3 and/or 4 of the HID device, connect a suitable rotary encoder or pulse generator to the following pins of the **I/O** socket of the E-871:
  - For axis 3 of the HID device: pins 1 and 2
  - For axis 4 of the HID device: pins 3 and 4
- If you want to use axis 1 of the HID device, connect the following to the **Joystick** socket of the E-871:
  - If you want to operate a C-819.20 joystick only with this controller, connect it directly to the controller.

- If you want to operate a C-819.20 joystick with two controllers (i.e. two axes), connect the joystick to the C-819.20Y Y cable and connect both controllers to the X and Y branches of the cable. The joystick is powered via the X branch. For this reason, the X branch has to be connected to a controller even if the HID control is not to be enabled for this controller.
- If you want to connect an axis of a C-819.30 joystick, connect the corresponding cable of the joystick to the controller.
- If you want to use axis 2 of the HID device: connect an analog signal source that supplies -10 to 10 V to pin 2 of the **Joystick** socket.

## 6.10 Connecting Digital In- and Outputs

The digital inputs and outputs on the **I/O** socket of the E-871 can be used as follows:

- Outputs: Triggering of external devices; see "Digital Output Signals" (p. 102).
- Inputs: Use in macros (p. 114) and/or as a source for the reference point switch and limit switch signals of the axis (p. 114) and/or for the HID control (p. 118)

### 6.10.1 Connecting the Outputs

#### **INFORMATION**

Digital output signals are available on pins 5, 6, 7 and 8 of the **I/O** socket.

#### **INFORMATION**

If the C-170.PB pushbutton box from PI is connected to the **I/O** socket, it displays via LEDs the state of the digital output lines.

#### **Tools and accessories**

- Suitable cable, e. g. C-170.IO IO cable with open end, available as an optional accessory (p. 13)
- Device to be triggered having digital input for TTL signals

#### **Connecting a device to be triggered**

- Connect an appropriate device to one of the pins 5, 6, 7 or 8 of the **I/O** socket of the E-871.

## 6.10.2 Connecting the Inputs

### INFORMATION

Digital input signals can be fed into the E-871 via pins 1, 2, 3 and 4 of the **I/O** socket.

### INFORMATION

The digital inputs (pins 1 to 4) on the **I/O** socket can also be used as analog inputs.

- Digital: TTL
- Analog: 0 to +5 V

### Tools and accessories

- Suitable signal source:
  - If the digital inputs are to be used in macros, the C-170.PB pushbutton box, for example, can be connected, available as an optional accessory (p. 13).
  - If the digital inputs are to be used as the source for the reference and limit switch signals of the axis, the signal level may only change once across the entire travel range.
- If necessary: Suitable cable, e. g. C-170.IO IO cable with open end, available as an optional accessory (p. 13).

### Connecting a digital signal source

- If you want to use the digital inputs in macros or as switch signals: connect a suitable signal source to one of the pins 1, 2, 3 or 4 of the **I/O** socket of the E-871.
- If you want to use the digital inputs for the HID control, follow the instructions in "Connecting an HID Device" (p. 66).

## 6.11 Connecting Analog Signal Sources

The analog inputs on the **I/O** socket of the E-871 can be used as follows:

- Use in macros (p. 118): Details and examples of macros are found in "Controller Macros" (p. 131).
- Scanning applications with PIMikroMove (see PIMikroMove manual)

### **INFORMATION**

Analog input signals can be fed into the E-871 via pins 1, 2, 3 and 4 of the **I/O** socket.

### **INFORMATION**

The analog inputs (pins 1 to 4) on the **I/O** socket can also be used as digital inputs.

- Analog: 0 to +5 V
- Digital: TTL

### **Tools and accessories**

- Suitable signal source
- If necessary: Suitable cable, e. g. C-170.IO IO cable with open end, available as an optional accessory (p. 13).

### **Connecting an analog signal source**

- Connect an appropriate signal source to one of the pins 1, 2, 3 or 4 of the **I/O** socket of the E-871.

## 7 Start-Up

### In this Chapter

General Notes on Start-Up .....	71
Adapting the DIP Switch Settings.....	72
Switching on the E-871.....	75
Establishing Communication .....	76
Starting Motions.....	84
Setting Notch Filters .....	89
Optimizing the Servo-Control Parameters.....	94

### 7.1 General Notes on Start-Up

#### CAUTION



#### **Risk of electric shock if the protective earth conductor is not connected!**

If a protective earth conductor is not or not properly connected, dangerous touch voltages can occur on the E-871 in the case of malfunction or failure of the system. If touch voltages exist, touching the E-871 can result in minor injuries due to electric shock.

- Connect the E-871 to a protective earth conductor before start-up (p. 61).
- Do **not** remove the protective earth conductor during operation.
- If the protective earth conductor has to be removed temporarily (e. g. in the case of modifications), reconnect the E-871 to the protective earth conductor before starting it up again.

## 7.2 Adapting the DIP Switch Settings

### 7.2.1 General Procedure

#### INFORMATION

Changed DIP switch settings become effective after the E-871 is switched on.

- If you have changed the DIP switch settings while the E-871 was switched on, switch the E-871 off and back on again to activate the new settings.

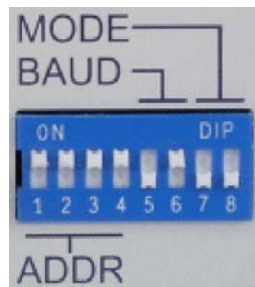


Figure 8: DIP switches: switch up = ON; switch down = OFF

Switches	Function
1 to 4	Controller address (p. 73); 16 possible combinations
5 and 6	Baud rate (p. 74)
7	without function
8	Update mode (p. 74)

#### Prerequisite

- ✓ The E-871 is switched off, i.e. the power supply is **not** connected to the power socket via the power cord.

#### Adapting the DIP switch settings

- Put the individual DIP switches in the correct position for your application. Details are given in the following tables.

## 7.2.2 Controller Address

Address*	S1	S2	S3	S4
1	<b>ON</b>	<b>ON</b>	<b>ON</b>	<b>ON</b>
2	ON	ON	ON	OFF
3	ON	ON	OFF	ON
4	ON	ON	OFF	OFF
5	ON	OFF	ON	ON
6	ON	OFF	ON	OFF
7	ON	OFF	OFF	ON
8	ON	OFF	OFF	OFF
9	OFF	ON	ON	ON
10	OFF	ON	ON	OFF
11	OFF	ON	OFF	ON
12	OFF	ON	OFF	OFF
13	OFF	OFF	ON	ON
14	OFF	OFF	ON	OFF
15	OFF	OFF	OFF	ON
16	OFF	OFF	OFF	OFF

\*Factory settings are shown in bold.

### INFORMATION

A unique address must be set for each controller in a daisy chain network. In doing so, one of the controllers must have the address 1. This controller does not have to be the one directly connected to the PC.

### INFORMATION

A non-networked controller must have the address 1, if

- it is to be used in PIMikroMove.
- it is to be used in LabVIEW.
- it is to be addressed with the PITerminal without specifying the target address; in this case, in the responses of the E-871 target and sender addresses (p. 152) are omitted as well.

### 7.2.3 Baud Rate

Baud rate*	S5	S6
9600	ON	ON
19200	ON	OFF
38400	OFF	ON
<b>115200</b>	<b>OFF</b>	<b>OFF</b>

\*Factory settings are shown in bold.

#### **INFORMATION**

The same baud rate must be set for every controller in a daisy chain network.

### 7.2.4 Update Mode

Update mode	S8
Firmware update	ON
<b>Normal operation</b>	<b>OFF</b>

\*Factory settings are shown in bold.

#### **INFORMATION**

If the E-871 is in the firmware update mode (DIP switch 8 in "ON" (upper) position), all LEDs remain deactivated after switching on the E-871.



## 7.3 Switching on the E-871

### INFORMATION

The E-871 is intended for closed-loop operation with incremental position sensors (servo mode On). After switching-on, open-loop operation is enabled by default (servo mode Off).

- Get the current operating mode with the `SVO?`, `#4` or `SRG?` commands.
- Enable closed-loop operation with the `SVO` command.
- If necessary, program a start-up macro that starts the E-871 via the `SVO` command in closed-loop operation; see "Setting up a start-up macro" (p. 141).

### INFORMATION

The ID chip is not read when you connect the stage while the E-871 is switched on.

- After connecting a stage, reboot the E-871 with the `RBT` command (p. 228) or with the corresponding PC software function in order to read the data from the ID chip.

### Prerequisites

- ✓ You have read and understood the General Notes on Start-Up (p. 71).
- ✓ The E-871 has been installed properly (p. 55).
- ✓ You have set the DIP switches of the E-871 in accordance with your application (p. 72).

### Switching on the E-871

- Connect the power cord of the power supply with the power socket.

The E-871 loads information to the volatile memory in the following order:

- a) Parameter values from the nonvolatile memory
- b) Parameter values from the ID chip of the stage

The **STA** LED on the front panel of the E-871 displays the state of the E-871:

- green: E-871 is ready for normal operation
- off: If DIP switch 8 is in the "ON" (upper) position, the E-871 is in firmware update mode. Otherwise the E-871 might be defective.
- If DIP switch 8 is in the "OFF" (lower) position and the **STA** LED does not light after switching-on, contact our customer service department (p. 301).

## 7.4 Establishing Communication

The procedure for PIMikroMove is described in the following.

### INFORMATION

Use the **USB Daisy Chain** and **RS-232 Daisy Chain** tabs in the PC software for establishing communication only if you have actually connected a daisy chain network to the PC.

### INFORMATION

A non-networked controller must have the address 1, if it is to be used in PIMikroMove. Details see "Controller Address" (p. 73).

### 7.4.1 Establishing Communication via RS-232

#### Prerequisites

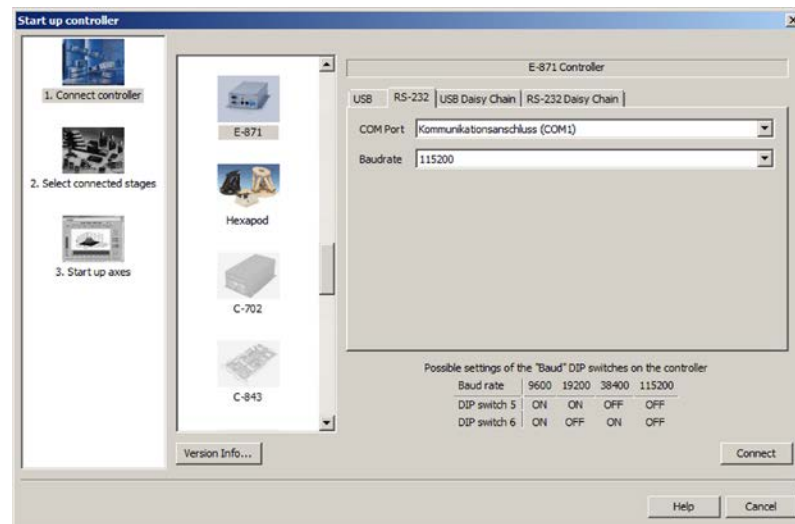
- ✓ You have read and understood the General Notes on Start-Up (p. 71).
- ✓ The E-871 is connected to the RS-232 interface of the PC (p. 64).
- ✓ You have made the following settings with the respective DIP switches prior to switching on the E-871 (p. 72):
  - controller address = 1
  - appropriate baud rate
- ✓ The E-871 is switched on (p. 75).
- ✓ The PC is switched on.
- ✓ The required software is installed on the PC (p. 55).
- ✓ You have read and understood the manual of the used PC software. The software manuals are found on the product CD.

#### Establishing communication

1. Start PIMikroMove.

The **Start up controller** window opens with the **Connect controller** step.

- If the **Start up controller** window does not automatically open, select the **Connections > New...** menu item in the main window.



2. Select **E-871** in the field for controller selection.
3. Select the **RS-232** tab on the right side of the window.
4. In the **COM Port** field, select the COM port of the PC to which you have connected the E-871.
5. In the **Baudrate** field, set the value that is set with DIP switches 5 and 6 of the E-871.

This adapts the baud rate of the PC to the baud rate of the E-871.

6. Click **Connect** to establish communication.

If communication has been successfully established, PIMikroMove guides you through the configuration of the E-871 for the connected stage, see "Starting Motions" (p. 84).

## 7.4.2 Establishing Communication via USB

### INFORMATION

If the controller is connected via the USB connection and switched on, the USB interface in the PC software is also shown as a COM port.

### Prerequisites

- ✓ You have read and understood the General Notes on Start-Up (p. 71).
- ✓ The E-871 is connected to the USB interface of the PC (p. 64).

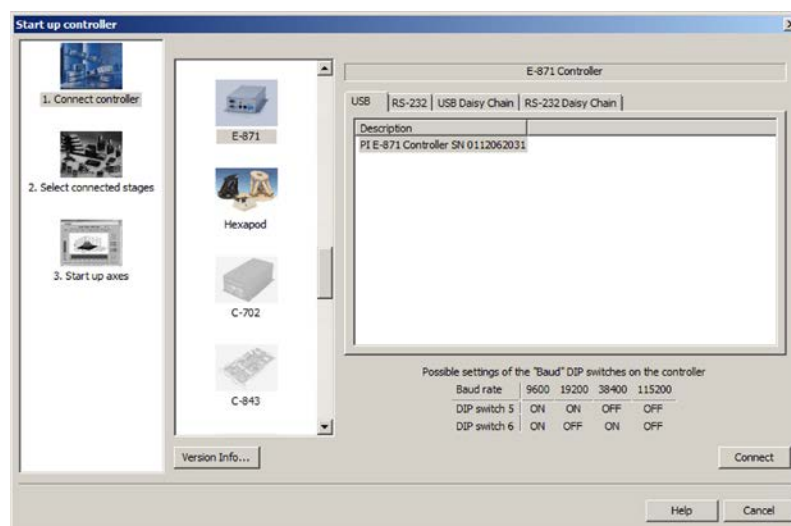
- ✓ Prior to switching on the E-871, you have set the DIP switches for the controller address to the address 1 (p. 72).
- ✓ The E-871 is switched on (p. 75).
- ✓ The PC is switched on.
- ✓ The required software and USB drivers are installed on the PC.
- ✓ You have read and understood the manual of the used PC software. The software manuals are found on the product CD.

## Establishing communication

1. Start PIMikroMove.

The **Start up controller** window opens with the **Connect controller** step.

- If the **Start up controller** window does not automatically open, select the **Connections > New...** menu item in the main window.



2. Select **E-871** in the field for controller selection.
3. Select the **USB** tab on the right side of the window.
4. On the **USB** tab, select the connected E-871.
5. Click **Connect** to establish communication.

If communication has been successfully established, PIMikroMove guides you through the configuration of the E-871 for the connected stage, see "Starting Motions" (p. 84).

- If communication could not be established, look for a solution to the problem in "Troubleshooting" (p. 297).

### 7.4.3 Establishing Communication for Networked Controllers

The procedure for PIMikroMove and for PITerminal is described in the following.

#### INFORMATION

If you are establishing communication with a networked controller via PITerminal, the address of the controller to be addressed is required in every command line. Details see "Target and Sender Address" (p. 152).

- Use PITerminal to test communication with networked controllers.

#### INFORMATION

The RS-232 output lines of some PCs are not adapted for the maximum number of 16 controllers in a network. If you have connected a daisy chain network to such a PC via the RS-232 interface, communication malfunctions may occur (e. g. timeout). In case of communication malfunctions:

1. Disconnect the null-modem cable from the **RS-232 In** socket of the controller which is connected to the PC.
2. Connect the daisy chain network to the PC via the USB interface of this controller.

#### Prerequisites

- ✓ You have read and understood the General Notes on Start-Up (p. 71).
- ✓ You have set up a daisy chain network (p. 65).
- ✓ Prior to switching-on you have properly set on each networked E-871 the DIP switches for the controller address (p. 73) and the baud rate (p. 74).
- ✓ Every controller in the daisy chain network is switched on (p. 75).
- ✓ The PC is switched on.
- ✓ The required software is installed on the PC (p. 55).
- ✓ If you have connected the first controller in the chain to the PC via the USB interface: The USB drivers are installed on the PC (p. 55).
- ✓ You have read and understood the manual of the used PC software. The software manuals are found on the product CD.

## Establishing communication with PIMikroMove

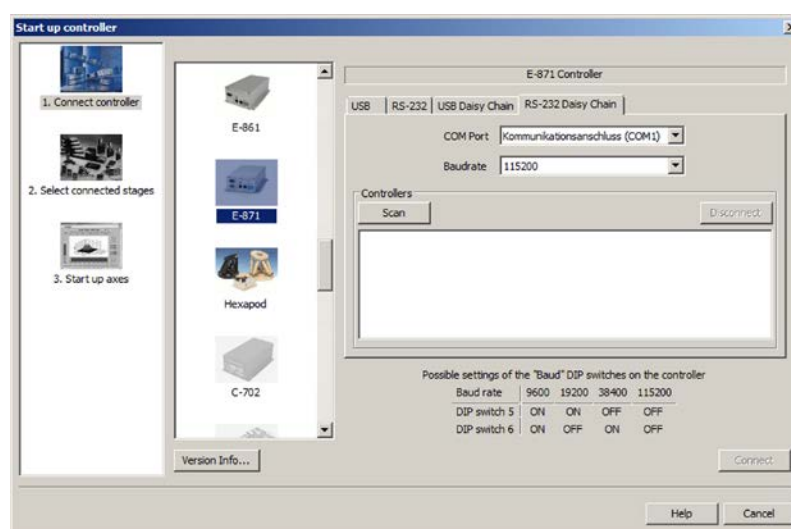
1. Start PIMikroMove.

The **Start up controller** window opens with the **Connect controller** step.

- If the **Start up controller** window does not automatically open, select the **Connections > New...** menu item in the main window.

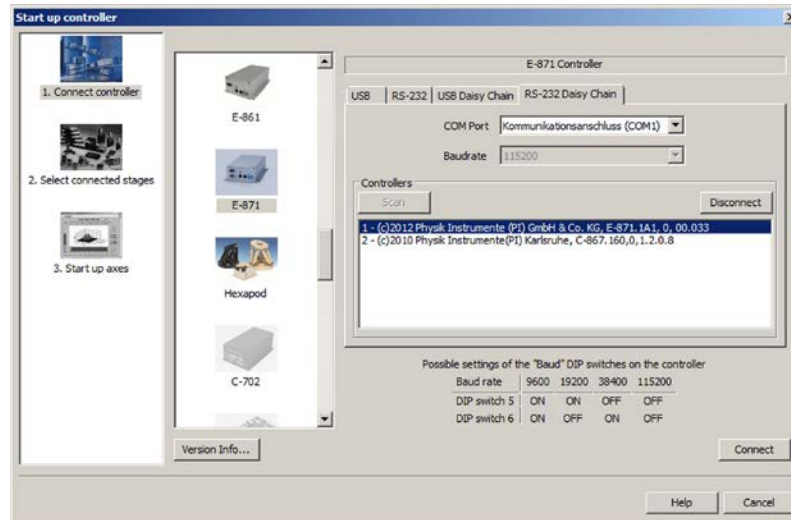
2. Select the appropriate controller type in the field for controller selection.

In the example in the following figures, there is a daisy chain consisting of one E-871 with the controller address 1 and one C-867.160 with the controller address 2. If you want to connect the E-871 first, select **E-871**.



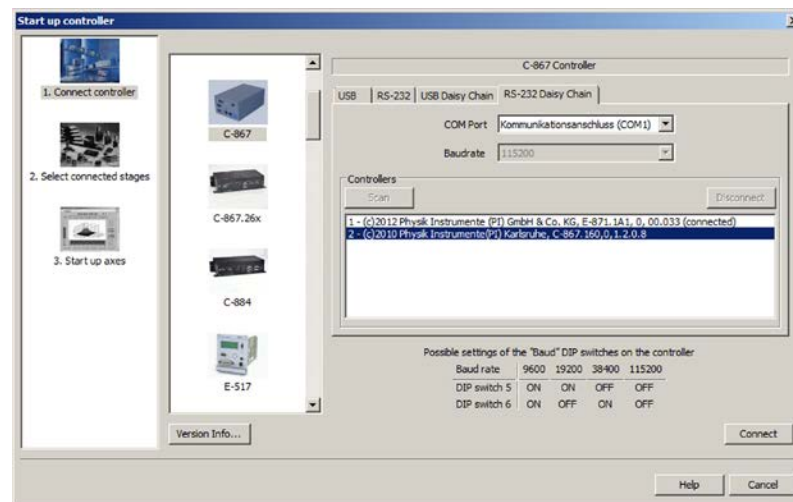
3. Select the appropriate tab on the right side of the window:
  - If you have connected the first controller in the chain to the PC via the RS-232 interface, select the **RS-232 Daisy Chain** tab.
  - If you have connected the first controller in the chain to the PC via the USB interfaces, select the **USB Daisy Chain** tab.
4. Make the settings for the interface in the selected tab:
  - **RS-232 Daisy Chain** tab:
    - In the **COM Port** field, select the PC COM port to which you have connected the E-871.
    - In the **Baudrate** field, set the value which is set with DIP switches 5 and 6 of the E-871.
  - **USB Daisy Chain** tab:
    - In the top section of the tab, select the connected E-871.

- In the bottom section of the tab, click on the **Scan** button to list every controller in the daisy chain network.



- Select a controller from the list. The selection must match the controller type that you selected in step 2.
- Click **Connect** to establish communication with the controller selected.  
If communication has been successfully established, PIMikroMove guides you through the configuration of the E-871 for the connected stage:
  - Proceed further as described in "Starting Motions" (p. 84).
- If you want to connect an additional controller of the daisy chain network, select the **Connections > New...** menu item in the main window.
- Execute again steps 2, 6 and 7 in the given order.

In the following figure, the **C-867** is to be connected as well.



10. Repeat steps 8, 2, 6 and 7 in the given order for every additional controller in the daisy chain that you want to connect.

If you want to terminate communication with one of the controllers of the daisy chain network:

- In the main window, select the **Connections > Close** menu item for the corresponding controller.

## Establishing communication with PITerminal

### INFORMATION

Via the **Mercury** button, PITerminal supports controllers with older firmware versions that are not compatible with GCS.

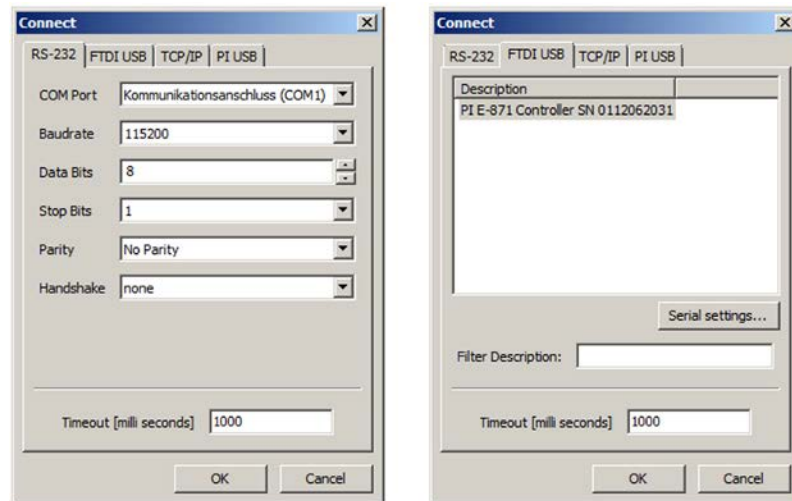
- Make sure that the **Mercury** button is **not** enabled in PITerminal.

1. Start PITerminal.
2. Click on **Connect....**

The **Connect** window opens.

3. In the **Connect** window, select the **RS-232** or **FTDI USB** tab, depending on via which interface you have connected the first controller in the chain to the PC.



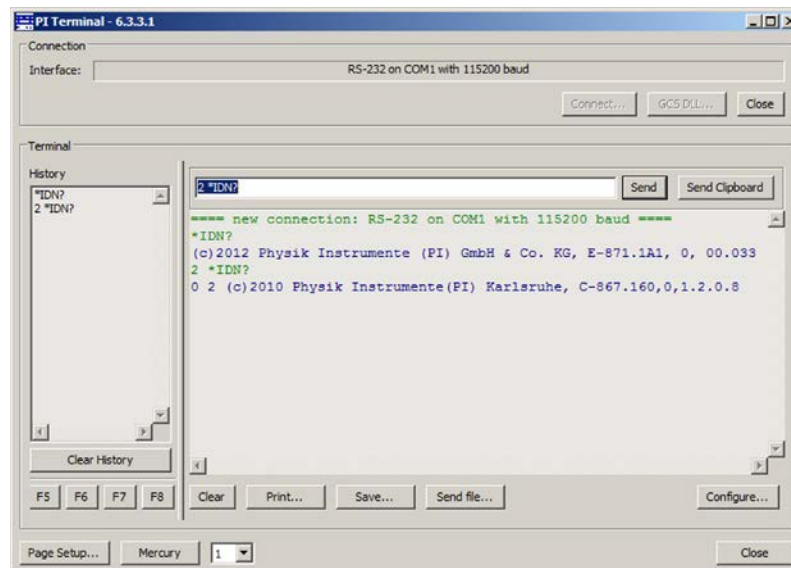


4. Make the settings for the interface in the selected tab:
  - **RS-232** tab:
    - In the **COM Port** field, select the PC COM port to which you have connected the E-871.
    - In the **Baudrate** field, set the value which is set with DIP switches 5 and 6 of the E-871.
  - **FTDI USB** tab:
    - Select the connected E-871.
5. Click **OK** to establish communication.
6. Send the `*IDN?` command for every controller in the daisy chain network to check the communication.

In the example in the following figure, the daisy chain comprises one E-871.1A1 with the controller address 1 and one C-867.160 with the controller address 2. Send:

- `*IDN?` to query the device identification string of the controller with the address 1; the controller address is not required (because = 1)
- `2 *IDN?` to query the device identification string of the controller with the address 2.

Further information see "Target and Sender Address" (p. 152).



## 7.5 Starting Motions

In the following, PIMikroMove is used to move the stage. Here the program guides you through the following steps so that you do not have to deal with the respective GCS commands:

- Configuration of the E-871 for the connected stage
- Switching on the servo mode (closed-loop operation)
- Executing a reference move; details see "Reference Point Definition" (p. 37).

### NOTICE



#### Damage to the stage and the load from oscillations

Unsuitable settings of the notch filter and the servo-control parameters of the E-871 can cause the stage to oscillate. Oscillations can damage the stage and/or the load affixed to it.

- If the stage is oscillating (unusual operating noise), immediately switch off the servo mode or disconnect the E-871 from the power source.
- Only switch on the servo mode after you have modified the settings of the notch filter and the servo-control parameters of the E-871; see „Setting the Notch Filter“ (p. 89) and "Optimizing Servo-Control Parameters" (p. 94).

**INFORMATION**

After communication has been established between the E-871 and the PC, PIMikroMove guides you through the configuration of the E-871 for the connected stage. The selection of the offered configuration steps by PIMikroMove is based on the evaluation of the following parameter values from the volatile memory of the E-871:

- **Stage Name** (ID 0x3C): The value is used by PIMikroMove as a criterion for finding a suitable parameter set in the stage databases.
- **Stage Type** (ID 0x0F000100): The value was loaded from the ID-Chip (p. 41) of the connected stage when the E-871 was switched on.

Possible configuration steps:

- When the values of the parameters 0x3C and 0x0F000100 are identical, PIMikroMove assumes that all parameters of the E-871 have already been adapted to the connected stage. The **Start up controller** window goes directly to the **Start up axes** step, where the reference move can be started.
- If the values of the parameters 0x3C and 0x0F000100 are not identical, the **Stage Type Configuration** window opens. The **Yes, configure for ...** button can be used to load a suitable parameter set from a stage database to the E-871. After the parameter set has been loaded, the **Start up controller** window goes to the **Start up axes step**. If no suitable parameter set is available in the stage databases, the **Stage Type Configuration** window will contain a corresponding notice.
- If the value of the parameter 0x0F000100 is empty because the stage does not have an ID chip, for example, the **Start up controller** window will go to the **Select connected stages** step.

**Prerequisites**

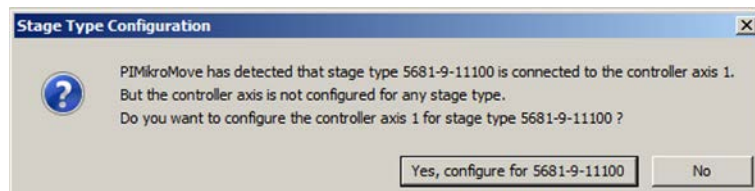
- ✓ You have read and understood the General Notes on Start-Up (p. 71).
- ✓ PIMikroMove is installed on the PC (p. 55).
- ✓ You have read and understood the PIMikroMove manual. The manual is found on the product CD.
- ✓ You have installed the latest versions of the PIMicosStages2.dat and PIStages2.dat stage databases on your PC (p. 57).
- ✓ If PI has provided you with a custom stage database for your stage, then you have installed this database on your PC (p. 59).

- ✓ You have installed the stage in the same manner as it will be used in your application (corresponding load, orientation and fastening).
- ✓ You have connected the stage to the E-871 (p. 63).
- ✓ You have established communication between the E-871 and the PC with PIMikroMove (p. 76).

### Starting motions with PIMikroMove

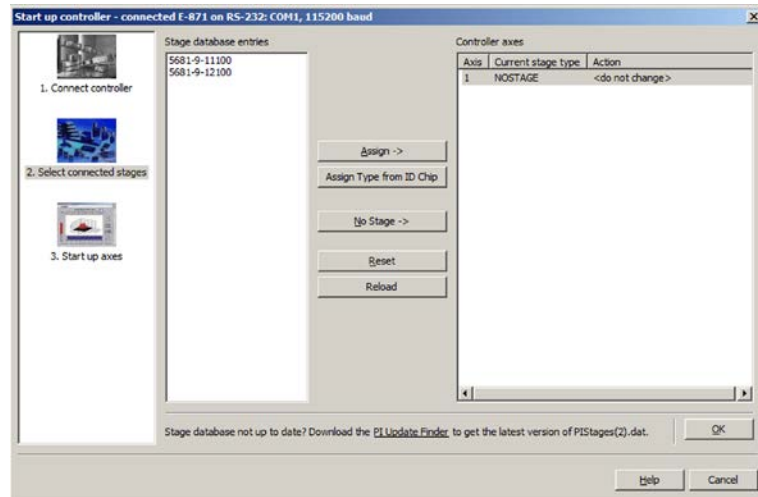
1. If one of the two following points applies, configure the E-871 for the connected stage:
  - The **Stage Type Configuration** window has opened.
  - In the **Start up controller** window, the **Select connected stages** step is displayed.

If the **Stage Type Configuration** window has opened:

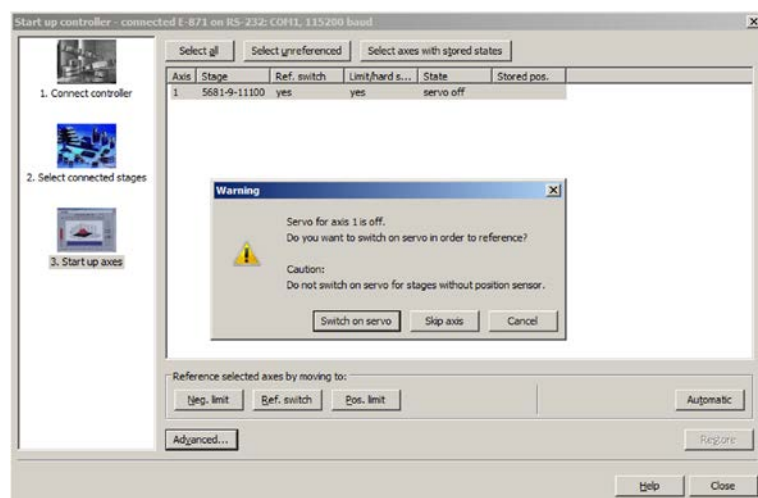


- Click the **Yes, configure for ...** button to load the appropriate parameter set from a stage database to the volatile memory of the E-871. The **Stage Type Configuration** window closes, and the **Start up controller** window goes to the **Start up axes** step.

If the **Select connected stages** step is displayed in the **Start up controller** window:



- a) Select the appropriate stage type: Click **Assign Type from ID Chip** or mark the appropriate stage type in the **Stage database entries** list.
  - b) If you have marked the appropriate stage type in the **Stage database entries** list in step a, click **Assign**.
  - c) Confirm the selection with **OK** to load the parameter settings for the selected stage type from the stage database to the volatile memory of the E-871. The **Start up controller** window goes to the **Start up axes** step.
2. In the **Start up axes** step, execute the reference move for the axis so that the controller knows the absolute axis position:



- If you want to start the reference move to the reference point switch, click on **Ref. switch**.

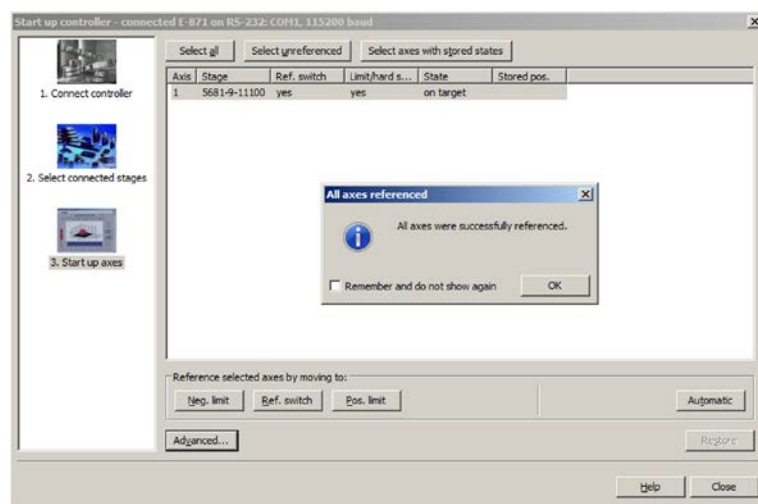
- If you want to start the reference move to the negative physical limit of the travel range, click **Neg. limit**.
- If you want to start the reference move to the positive physical limit of the travel range, click **Pos. limit**.

If a warning message appears indicating that the servo mode is switched off:

- Switch on the servo mode by clicking on the **Switch on servo** button (closed-loop operation).

The axis executes the reference move.

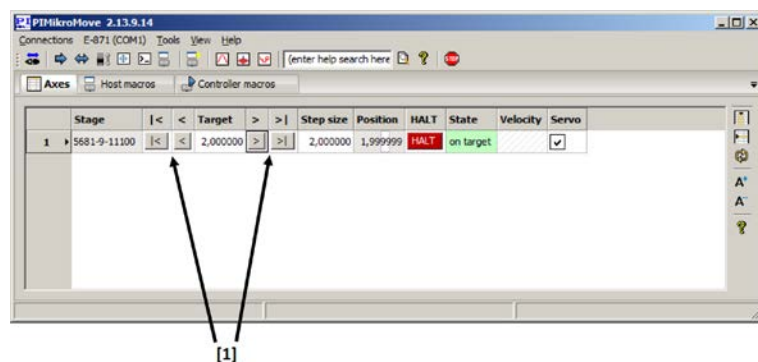
3. After a successful reference move, click **OK > Close**.



The main window of PIMikroMove opens.

4. Start a few test motions of the axis.

In the main window of PIMikroMove, you can execute, for example, motions of a particular distance (specification in **Step size** column) or to the limits of the travel range by clicking the corresponding arrow keys [1] for the axis.



## 7.6 Setting the Notch Filter

The notch filter corrects the control value in linear mode. The corrections by the notch filter take place in closed-loop and open-loop operation. The appropriate frequency component is reduced in the control value to compensate for undesired resonances in the mechanical system. Adjusting the notch filter frequency can be useful, particularly in the case of very high loads.

### INFORMATION

The settling behavior of the axis in closed-loop operation is influenced by the notch filter settings.

- Set the notch filter before you optimize the servo-control parameters (p. 94).

To set the notch filter, a step response is recorded in open-loop operation. The procedure for PIMikroMove is described in the following. Details on using the data recorder and on the configuration of the graphic display, see PIMikroMove user manual (SM148E).

### Prerequisite

- ✓ You have installed the stage in the same manner as it will be used in your application (corresponding load, orientation and fastening).
- ✓ You have started initial motions with PIMikroMove (p. 84).
- ✓ All devices are still ready for operation.

### Setting the notch filter

1. In the main window of PIMikroMove, open the **Data Recorder** window via the **E-871 > Show/Hide data recorder** menu item.
2. Switch off the servo mode with the **Servo** check box (uncheck).
3. Configure the data recorder.
  - a) Set the value 1 for the amplitude of the step to be performed (= 1 step in step mode).
  - b) Set the value 5 for the record table rate in the **Record Rate** field.
  - c) Set the value 1024 for the number of data points to be read for the graphic display.

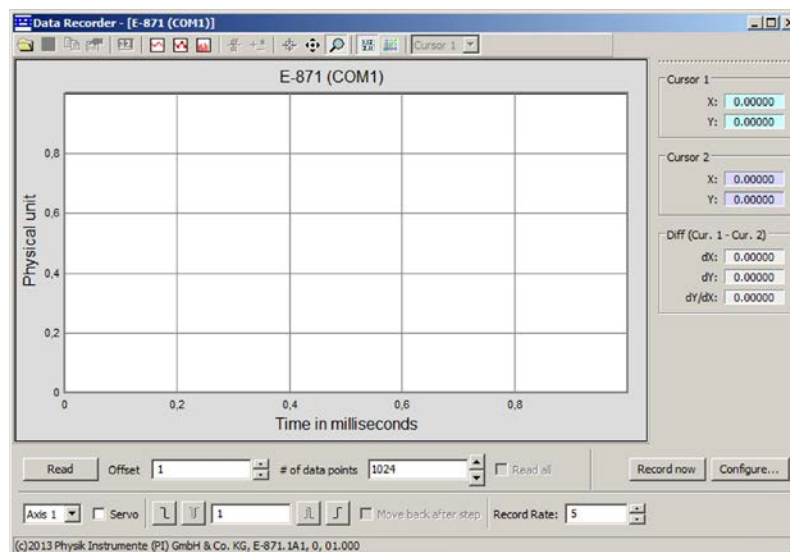


Figure 9: PIMikroMove: Data recorder

- d) Click the **Configure...** button and make sure that "Actual Position of Axis" is selected in the **Configure Data Recorder** window as the quantity to be recorded.

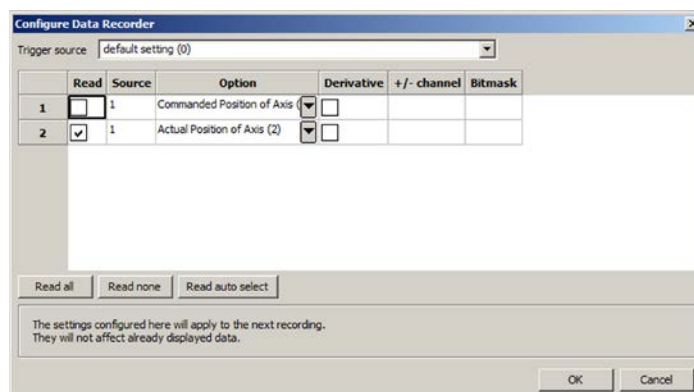



Figure 10: PIMikroMove: selection of the quantity to be recorded

4. In the **Data Recorder** window, start the step in the positive direction as well as the recording by clicking the  button.



The axis executes the step and the step response is recorded and displayed graphically.

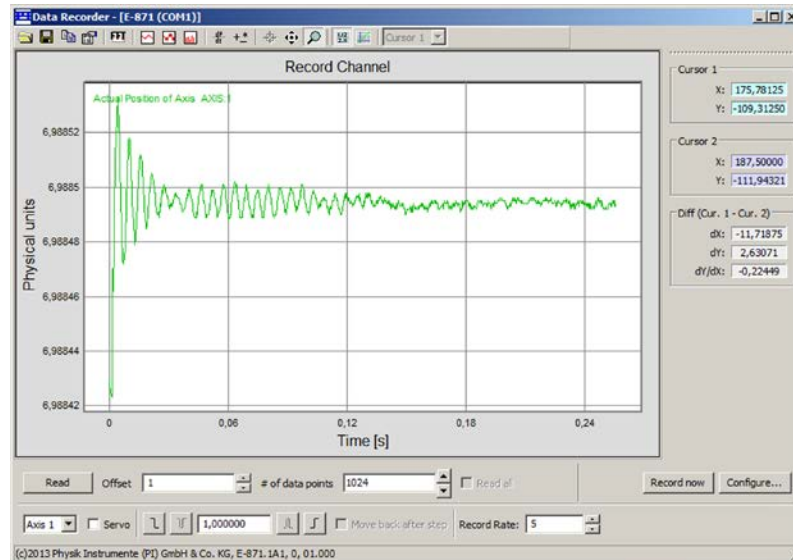


Figure 11: PIMikroMove: graphic display of the step response

5. Determine the resonant frequency of the axis from the graphic display of the step response:
  - a) Calculate the FFT (Fast Fourier Transformation) of the step response by clicking the **FFT** button. The FFT is graphically displayed.

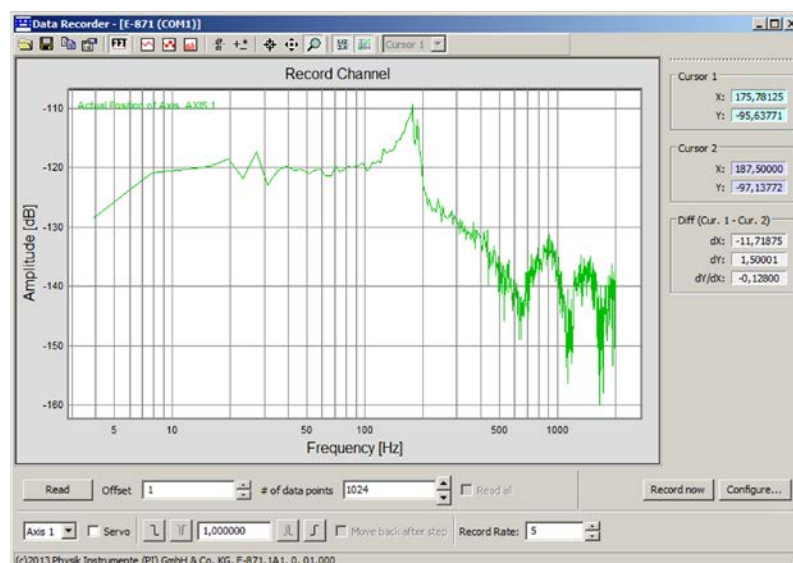





Figure 12: PIMikroMove: FFT of the step response

- b) If necessary, enlarge the view by clicking the  button and dragging the mouse pointer, which has turned into a magnifying glass, over a section of the graphic display with the left mouse button pressed (clicking the right mouse button in the graphics field reduces the view back to the original size).
- c) Display the cursors in the graphic display by clicking the  button.
- d) Activate the cursor movement with the mouse by clicking the  button
- e) Select cursor 1 in the selection field in the icon bar above the graphic display.
- f) Place cursor 1 on the resonant frequency by clicking it and dragging it with the left mouse button pressed. The resonant frequency can be identified in the FFT graph by the distinct maximum.  
In the example shown - at a load of 1.05 kg - there is a first resonant frequency at 175 Hz (can be read in field **X:** in the **Cursor 1** area to the right of the graphic display).
- g) If a second resonant frequency is visible: Select cursor 2 in the selection field in the icon bar above the graphic display.
- h) Place cursor 2 on the second resonant frequency by clicking it and dragging it with the left mouse button pressed. In the example shown, the second resonant frequency is at 187 Hz (can be read in field **X:** in the **Cursor 2** area to the right of the graphic display).

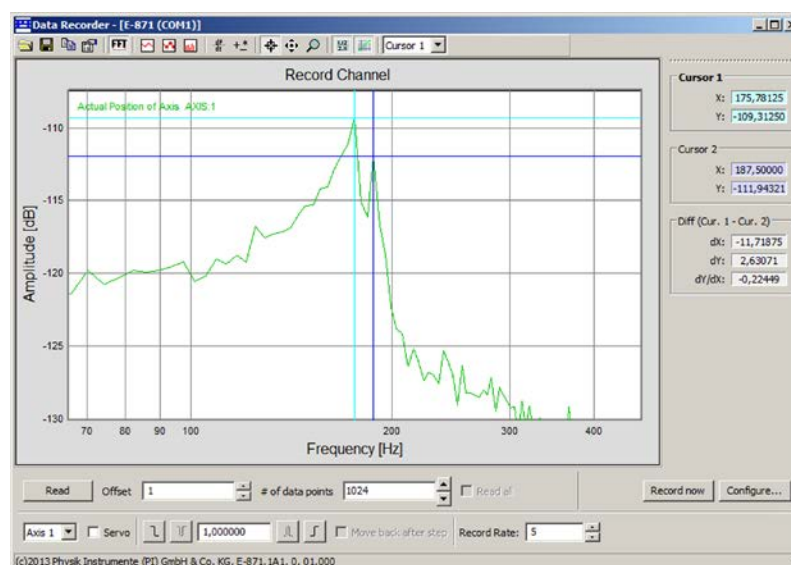


Figure 13: PIMikroMove: resonant frequencies in FFT, marked with cursors, enlarged view

6. In the main window of PIMikroMove, open the single axis window for the connected stage by selecting the stage in the **View > Single Axis Window** menu.

7. Expand the view of the single axis window by clicking on the > button at the right edge of the window.

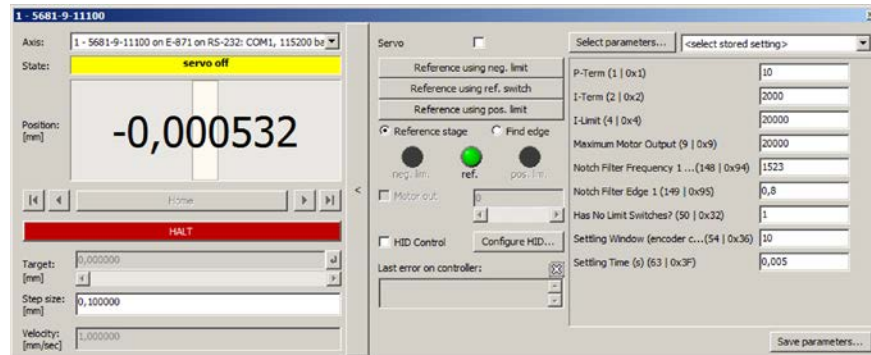


Figure 14: PIMikroMove: Expanded single axis window

8. Set the notch filter:

The value of the **Notch Filter Frequency 1** parameter (ID 0x94) must be set to the resonant frequency determined in step 5.

When two resonant frequencies located close to each other were determined, as in the present example (175 Hz and 187 Hz), the **Notch Filter Frequency 1** parameter (ID 0x94) should be set to a value that is roughly in the middle between the two resonant frequencies - here 181 Hz. In this case, the value of the **Notch Filter Edge 1** parameter (ID 0x95) has to be reduced to a value in the range of 0.3 to 0.2 in order to increase the notch filter bandwidth.

Proceed as follows to enter the values:

- a) If the **Notch Filter Frequency 1** (ID 0x94) and **Notch Filter Edge 1** (ID 0x95) parameters are not included in the list on the right side of the window, click **Select parameters...** and add them to the list.
  - b) Type the new parameter value into the appropriate input field in the list.
  - c) Press the **Enter** key of the PC to transfer the parameter value to the volatile memory of the controller. When this is done, the entry changes its color from blue to black.
9. Save the new settings. You have the following options:
    - Save a parameter set in the PI\_UserStages2.dat stage database on the PC. See "Creating or Modifying a Stage Type" (p. 277).
    - Transfer the current values of **all** parameters from the volatile to the nonvolatile memory of the E-871 (p. 275).

## 7.7 Optimizing the Servo-Control Parameters

Adjusting the P-I-D controller optimizes the dynamic properties of the system (overshoot and settling time). The optimum P-I-D controller setting depends on your application and your requirements.

### INFORMATION

The settling behavior of the axis in closed-loop operation is primarily influenced by the notch filter settings.

- Set the notch filter before you optimize the servo-control parameters.

The optimization of the P-I-D controller is typically done empirically and comprises the following parameters; details see "Control Algorithm and Other Control Value Corrections" (p. 26):

- **P-Term** (0x1)
- **I-Term** (0x2)

The behavior of the stage is monitored under various values in closed-loop operation.

In the following, PIMikroMove is used for optimizing the P-I-D servo-control parameters.

### Prerequisite

- ✓ You have installed the stage in the same manner as it will be used in your application (corresponding load, orientation and fastening).
- ✓ You have set the notch filter with PIMikroMove (p. 89).
- ✓ All devices are still ready for operation.

### Checking the servo-control parameters

1. In the main window of PIMikroMove, open the **Data Recorder** window via the **E-871 > Show/Hide data recorder** menu item.
2. Switch on the servo mode with the **Servo** check box (check).
3. Configure the data recorder.
  - a) Set the value 0.1 (= 0.1 mm) for the amplitude of the step to be performed.
  - b) Set the value 10 for the record table rate in the **Record Rate** field.
  - c) Set the value 512 for the number of data points to be read for the graphic display.

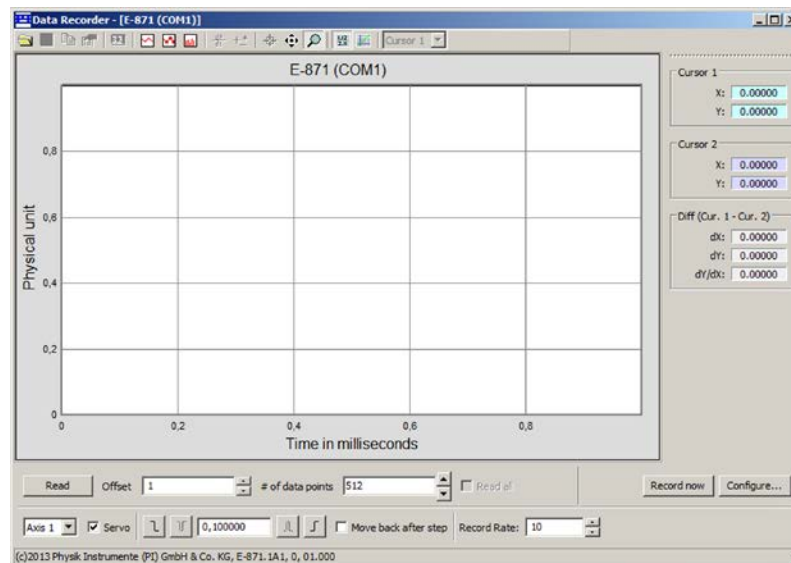


Figure 15: PIMikroMove: Data recorder

- d) Click the **Configure...** button and make sure that "Commanded Position of Axis" and "Actual Position of Axis" are selected in the **Configure Data Recorder** window as the quantities to be recorded.

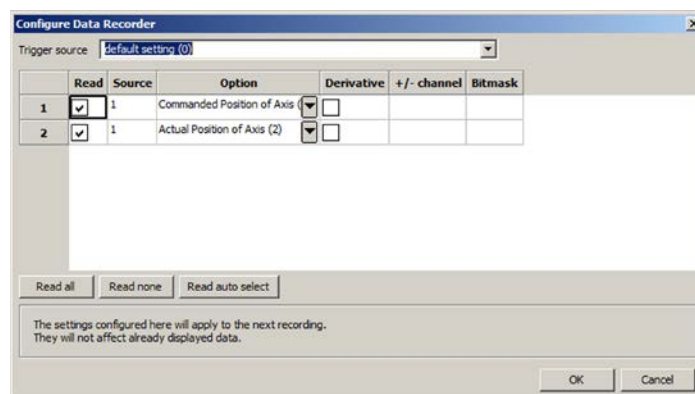



Figure 16: PIMikroMove: selection of the quantities to be recorded

4. In the **Data Recorder** window, start the step in the positive direction as well as the recording by clicking the  button.

The axis executes the step and the step response is recorded and displayed graphically.

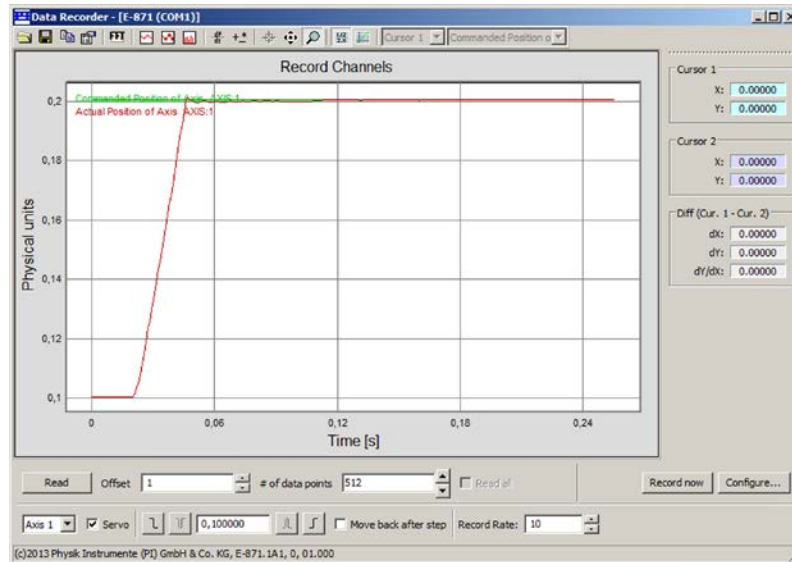



Figure 17: PIMikroMove: graphic display of the step response

5. Check the displayed step response.
  - If necessary, enlarge the view by clicking the  button and dragging the mouse pointer, which has turned into a magnifying glass, over a section of the graphic display with the left mouse button pressed (clicking the right mouse button in the graphics field reduces the view back to the original size).

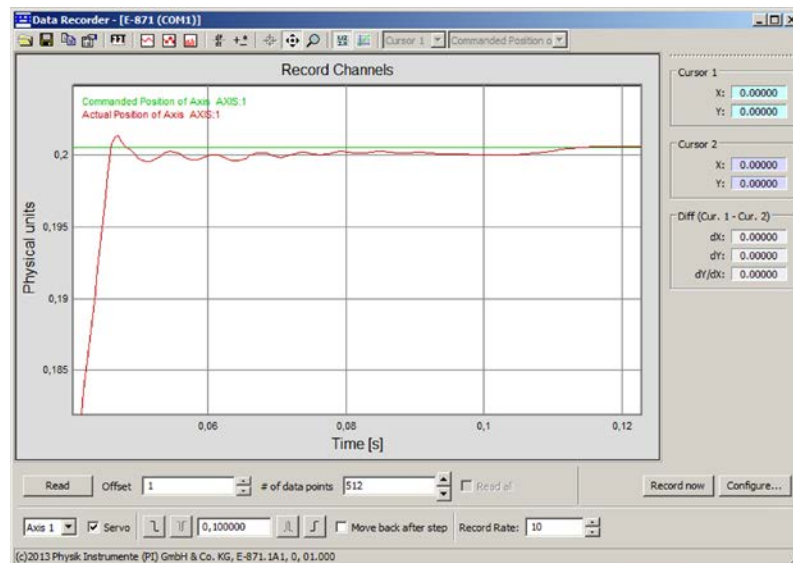


Figure 18: PIMikroMove: step-and-settle of the axis; enlarged view of the step response

If the result is satisfactory (i.e. minimum overshoot, settling time not too long):

- You already have optimum parameter settings and do not have to do anything further.

If the result is not satisfactory:

- Optimize the servo-control parameters, see below.

## Optimizing the servo-control parameters

1. In the main window of PIMikroMove, open the single axis window for the connected stage by selecting the stage in the **View > Single Axis Window** menu.
2. Expand the view of the single axis window by clicking on the > button at the right edge of the window.

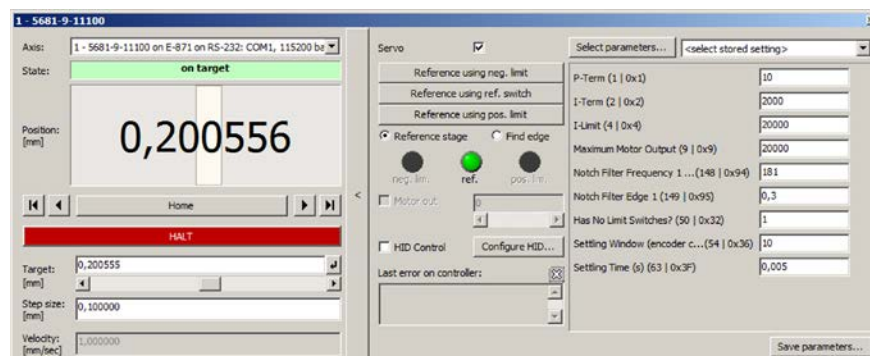


Figure 19: PIMikroMove: Expanded single axis window

3. Enter new values for the servo-control parameters:
  - a) If the **P-Term (ID 0x1)** and **I-Term (ID 0x2)** parameters are not included in the list on the right side of the window, click **Select parameters...** and add them to the list.
  - b) Type the new parameter value into the appropriate input field in the list.
  - c) Press the **Enter** key of the PC to transfer the parameter value to the volatile memory of the controller. When this is done, the entry changes its color from blue to black.
4. In the **Data Recorder** window, record the step response of the stage again.

If the result is not satisfactory:

  - Enter different values for the servo-control parameters and record the step response again.

If you are satisfied with the result and want to keep the new servo-control parameter settings, save the new settings. You have the following options:

  - Save a parameter set in the PI\_UserStages2.dat stage database on the PC. See "Creating or Modifying a Stage Type" (p. 277).
  - Transfer the current values of **all** parameters from the volatile to the nonvolatile memory of the E-871 (p. 275).



## 8 Operation

### In this Chapter

Motion Errors .....	99
Data Recorder .....	100
Digital Output Signals .....	102
Digital Input Signals .....	111
Analog Input Signals .....	116
Control with an HID Device .....	118
Controller Macros .....	131

## 8.1 Motion Errors

### 8.1.1 Protective Functions of the E-871

Motion errors can have the following causes, for example:

- Malfunctions of the drive or the position sensor of the stage
- Switch-off of the output for the piezo voltage on the E-871 due to overheating of the integrated PIShift drive electronics (internal temperature  $\geq 75\text{ °C}$ ).

A motion error is present if the current position of the axis does not change when the control value changes.

A motion error can occur in closed-loop or open-loop operation.

If a motion error occurs, the E-871 reacts as follows to protect the system from damage:

- Servo mode is switched off for the axis.
- The motion is stopped.
- Error code -1024 is set.

### 8.1.2 Re-establishing Readiness for Operation

#### Re-establishing readiness for operation

1. Send the `ERR?` command to read out the error code.

If a motion error occurred, error code -1024 is output. `ERR?` resets the error code to zero during the query.

2. Check your system and make sure that the following points are fulfilled:
  - The axis can be moved without danger.
  - The E-871 is **not** overheated (internal temperature is maximally 65 °C).
3. When the motion error has occurred in closed-loop operation:
  - Switch on the servo mode for the axis with the `SVO` command (p. 245).

When the servo mode is switched on, the target position is set to the current axis position.

#### INFORMATION

With the `CTO` (p. 169) and `TRO` (p. 250) commands, you can program the digital output lines of the E-871 so that they are activated in case of motion errors. The programmed output lines remain active until the error code is reset to 0. Details see "Setting Up "Motion Error" Trigger Mode" (p. 107).

## 8.2 Data Recorder

### 8.2.1 Data Recorder Properties

The E-871 contains a real-time data recorder. The data recorder can record different values for the axis (e.g. current position) and analog input signals.

The recorded data is temporarily stored in 2 data recorder tables with 1024 points each. Each data recorder table contains the data of one data source.

You can configure the data recorder, e.g., by defining the data type to be recorded and the data sources, and by specifying how the recording is to be started.

### 8.2.2 Setting up the Data Recorder

#### INFORMATION

The settings for setting up the data recorder can only be changed in the volatile memory of the E-871. After the E-871 has been switched on or rebooted, factory settings will be active unless a configuration takes place with a start-up macro.

## Reading out general information about the data recorder

- Send the `HDR?` command (p. 193).

The available record options and trigger options as well as information on additional parameters and commands for data recording are displayed.

## Configuring the data recorder

You can assign the data sources and record options to the data recorder tables.

- Send the `DRC?` command (p. 179) to read out the current configuration. Data recorder tables with the record option 0 are deactivated, i.e., nothing is recorded. In the default setting, the two data recorder tables of the E-871 record the following:
  - Data recorder table 1: commanded position of the axis
  - Data recorder table 2: current position of the axis
- Configure the data recorder with the `DRC` command (p. 178).

You can specify how the recording is to be triggered.

- Get the current trigger option with `DRT?` (p. 184).
- Change the trigger option with the `DRT` command (p. 183). The trigger option applies to all data recorder tables whose record option is not set to 0.

## Setting sampling interval

- Send the `RTR?` command (p. 232) to read out the record table rate of the data recorder.

The record table rate indicates after how many servo cycles a data point each is recorded. The default value is 10 servo cycles. The cycle time of the E-871 is 50 µs.

- Change the record table rate with the `RTR` command. (p. 231)

As the record table rate increases, you increase the maximum duration of the data recording.

## 8.2.3 Starting the Recording

- Start the recording with the trigger option set with `DRT`.

Regardless of the trigger option set, the data recording is always triggered when a step response measurement is started with `STE` (p. 244).

The data recording always takes place for all data recorder tables whose record option is not set to 0. It ends when the data recorder tables are full.

## 8.2.4 Reading Out Recorded Data

### INFORMATION

Reading out the recorded data can take some time, depending on the number of data points.

The data can also be read out while data is being recorded.

- Read out the last recorded data with the `DRR?` command.

The data is output in the GCS array format (see the SM146E user manual on the product CD).

## 8.3 Digital Output Signals

The digital outputs of the E-871 are available at the **I/O** socket (p. 310).

- Get the number of the output lines available on the E-871 with the `TIO?` command (p. 248).

External devices can be triggered via the digital outputs of the E-871. Potential applications:

- Linking the trigger output to the motion of the axis. Details and examples can be found in this section.
- Direct switching of output lines, e. g. in macros. Details and examples of macros can be found in "Controller Macros" (p. 131).

### 8.3.1 Commands for Digital Outputs

The following commands are available for the use of digital outputs:

Command	Syntax	Function
<code>CTO</code>	<code>CTO {&lt;TrigOutID&gt; &lt;CTOPam&gt; &lt;Value&gt;}</code>	Configures the conditions for the trigger output. Couples the trigger output to the axis motion.

Command	Syntax	Function
DIO	DIO {<DIOID> <OutputOn>}	Switches digital output lines directly to the low or high state, either separately or all lines at once. Should not be used for output lines on which the trigger output is enabled with TRO.
TRO	TRO {<TrigOutID> <TrigMode>}	Enables or disables the trigger output conditions set with CTO. Default: trigger output disabled.

One configuration setting can be made per CTO command:

CTO <TrigOutID> <CTOPam> <Value>

- <TrigOutID> is one digital output line of the controller.
- <CTOPam> is the CTO parameter ID in decimal format.
- <Value> is the value to which the CTO parameter is set.

The following trigger modes (<Value>) can be set for <CTOPam> = 3:

<Value>	Trigger mode	Short description
0 (default)	Position Distance	Once the axis has moved a specified distance, a trigger pulse is output (p. 104). Optionally, start and stop values can be defined to limit triggering to one position range and one particular direction of motion (negative or positive).
2	On Target	The on-target state of the axis selected is output at the selected trigger output (p. 106).
5	Motion Error	The selected digital output line becomes active when a motion error occurs (p. 107). The line stays active until the error code is reset to 0 (by a query with ERR?).
6	In Motion	The selected digital output line is active as long as the selected axis is in motion (p. 108).

<Value>	Trigger mode	Short description
7	Position+Offset	The first trigger pulse is output when the axis has reached a specified position. The next trigger pulses are each output when the axis position equals the sum of the last valid trigger position and a specified distance. The trigger output is stopped when a stop value is reached. The polarity sign of the distance value determines the direction of motion in which trigger pulses are to be output. Details see "Setting Up "Position + Offset" Trigger Mode" (p. 108).
8	Single Position	The selected digital output line is active when the axis position has reached or exceeded a given position (p. 110).

In addition, the polarity (active high / active low) of the signal can be set at the digital output (p. 111).

#### INFORMATION

The settings for the configuration of the digital output lines can only be modified in the volatile memory of the E-871. After the E-871 has been switched on or rebooted, factory default settings are enabled, provided that a configuration has not been carried out already with a start-up macro.

### 8.3.2 Setting Up "Position Distance" Trigger Mode

The *Position Distance* trigger mode lends itself to scanning applications. Once the axis has moved along the distance that was set with CTO parameter ID = 1 (TriggerStep), a trigger pulse is output. The pulse width is one servo cycle (50 µs).

The unit of the distance (TriggerStep) is subject to the settings of parameters 0xE and 0xF. Default is mm.

1. Configure the digital output line (<TrigOutID>) that is to be used as the trigger output:
  - Send CTO <TrigOutID> 2 A, where A indicates the axis to be moved.
  - Send CTO <TrigOutID> 3 0, where 0 specifies the *Position Distance* trigger mode.
  - Send CTO <TrigOutID> 1 S, where S indicates the distance.

2. If you want to enable the conditions for trigger output, send TRO  
<TrigOutID> 1.

**Example:**

A pulse on digital output line 1 is output every time the axis 1 of the stage has covered a distance of 0.1  $\mu\text{m}$ .

## ➤ Send:

CTO 1 2 1

CTO 1 3 0

CTO 1 1 0.0001

TRO 1 1

**"Position Distance" trigger mode with start and stop values for positive motion direction of the axis**

Optionally, you can define start and stop values for limiting the range and for specifying the motion direction of the axis (positive or negative).

**INFORMATION**

If start and stop values have the same value, they are ignored.

If the direction of motion is reversed before the axis position has reached the stop value, trigger pulses continue to be output.

1. Configure the digital output line (<TrigOutID>) that is to be used as the trigger output:
  - Send CTO <TrigOutID> 2 A, where A indicates the axis to be moved.
  - Send CTO <TrigOutID> 3 0, where 0 specifies the *Position Distance* trigger mode.
  - Send CTO <TrigOutID> 1 S, where S indicates the distance.
  - Send CTO <TrigOutID> 8 Start, where Start indicates the start value.
  - Send CTO <TrigOutID> 9 Stop, where Stop indicates the stop value.
2. If you want to enable the conditions for trigger output, send TRO  
<TrigOutID> 1.

**Example**

A pulse on digital output line 1 is output every time the axis 1 of the stage has covered a distance of 0.1  $\mu\text{m}$ , as long as axis 1 is moving in positive direction of motion within the range of 0.2  $\mu\text{m}$  to 0.55  $\mu\text{m}$  (start value < stop value).

➤ Send:

```
CTO 1 2 1
```

```
CTO 1 3 0
```

```
CTO 1 1 0.0001
```

```
CTO 1 8 0.0002
```

```
CTO 1 9 0.00055
```

```
TRO 1 1
```

**"Position Distance" trigger mode with start and stop values for negative motion direction of the axis**

The above example is presented with interchanged start and stop values in the following. Triggering occurs in negative motion direction of the axis (stop value < start value) in the range between 0.55  $\mu\text{m}$  and 0.2  $\mu\text{m}$ .

**Example:**

➤ Send:

```
CTO 1 2 1
```

```
CTO 1 3 0
```

```
CTO 1 1 0.0001
```

```
CTO 1 8 0.00055
```

```
CTO 1 9 0.0002
```

```
TRO 1 1
```

### 8.3.3 Setting Up "On Target" Trigger Mode

In the *On Target* trigger mode the on-target state of the axis selected (p. 29) is output at the selected trigger output.

1. Configure the digital output line (<TrigOutID>) that is to be used as the trigger output:



- Send `CTO <TrigOutID> 2 A`, where *A* indicates the axis to be moved.
  - Send `CTO <TrigOutID> 3 2`, where 2 specifies the *On Target* trigger mode.
2. If you want to enable the conditions for trigger output, send `TRO <TrigOutID> 1`.

**Example:**

The on-target state of axis 1 is to be output on the digital output line 1.

➤ Send:

```
CTO 1 2 1
```

```
CTO 1 3 2
```

```
TRO 1 1
```

### 8.3.4 Setting Up "Motion Error" Trigger Mode

The *Motion Error* trigger mode lends itself to monitoring motions. The selected digital output line becomes active when a motion error occurs on the connected axis. The line stays active until the error code is reset to 0 (by a query with `ERR?`).

#### INFORMATION

A motion error is present if the current position of the axis does not change when the control value changes.

Further information see "Motion Error" (p. 99).

1. Configure the digital output line (<TrigOutID>) that is to be used as the trigger output:
  - Send `CTO <TrigOutID> 3 5`, where 5 specifies the *Motion Error* trigger mode.
2. If you want to enable the conditions for trigger output, send `TRO <TrigOutID> 1`.

### 8.3.5 Setting Up "In Motion" Trigger Mode

In the *In Motion* trigger mode, the motion state of the selected axis is output at the selected trigger output. The line is active, as long as the selected axis is in motion.

The motion state can also be read with the #5 (p. 160), #4 (p. 159) and SRG? (p. 241) commands.

#### INFORMATION

If the axis is in motion, then bit 14 of the state register 1 of the axis is set.

1. Configure the digital output line (<TrigOutID>) that is to be used as the trigger output:
  - Send CTO <TrigOutID> 2 A, where A indicates the axis to be moved.
  - Send CTO <TrigOutID> 3 6, where 6 specifies the *In Motion* trigger mode.
2. If you want to enable the conditions for trigger output, send TRO <TrigOutID> 1.

#### Example:

Digital output line 1 is to be active if axis 1 of the stage is in motion.

➤ Send:

```
CTO 1 2 1
```

```
CTO 1 3 6
```

```
TRO 1 1
```

### 8.3.6 Setting Up "Position + Offset" Trigger Mode

The *Position+Offset* trigger mode lends itself to scanning applications. The first trigger pulse is output when the axis has reached a specified position (TriggerPosition). The next trigger pulses each are output when the axis position equals the sum of the last valid trigger position and a specified distance (TriggerStep). The trigger output is stopped when a stop value is reached. The polarity sign of the distance value determines for which direction of motion trigger pulses are to be output.

The pulse width is one servo cycle (50 µs).

The unit for TriggerPosition, TriggerStep and stop value is subject to the settings of parameters 0xE and 0xF. Default is mm.

1. Configure the digital output line (<TrigOutID>) that is to be used as the trigger output:
  - Send CTO <TrigOutID> 2 A, where A indicates the axis to be moved.
  - Send CTO <TrigOutID> 3 7, where 7 specifies the *Position+Offset* trigger mode.
  - Send CTO <TrigOutID> 1 S, where S indicates the distance.
  - Send CTO <TrigOutID> 10 TriPos, where TriPos indicates the position for the first trigger pulse output.
  - Send CTO <TrigOutID> 9 Stop, where Stop indicates the stop value.
2. If you want to enable the conditions for trigger output, send TRO <TrigOutID> 1.

#### Example 1:

The first trigger pulse is to be output on digital output line 1 if the absolute position of axis 1 is 1.5 mm. Then a pulse should be output on this line every time axis 1 has covered a distance of 0.1  $\mu\text{m}$  in positive direction. The last trigger pulse is to be output if the absolute axis position is 2.5 mm.

➤ Send:

```
CTO 1 2 1
CTO 1 3 7
CTO 1 1 0.0001
CTO 1 10 1.5
CTO 1 9 2.5
TRO 1 1
```

#### Example 2:

The first trigger pulse is to be output on digital output line 2 if the absolute position of axis B is 0.4 mm. Then a pulse should be output on this line every time axis B has covered a distance of 1  $\mu\text{m}$  in negative direction. The last trigger pulse is to be output if the absolute axis position is 0.1 mm.

➤ Send:

```
CTO 2 2 B
```

```
CTO 2 3 7
```

```
CTO 2 1 -0.001
```

```
CTO 2 10 0.4
```

```
CTO 2 9 0.1
```

### 8.3.7 Setting Up "Single Position" Trigger Mode

In the *Single Position* trigger mode, the selected digital output line is active when the axis position has reached or exceeded a given position (TriggerPosition).

The unit for TriggerPosition depends on the settings of the parameters 0xE and 0xF. Default is mm.

1. Configure the digital output line (<TrigOutID>) that is to be used as the trigger output:
  - Send CTO <TrigOutID> 2 A, where A indicates the axis to be moved.
  - Send CTO <TrigOutID> 3 8, where 8 specifies the *Single Position* trigger mode.
  - Send CTO <TrigOutID> 10 TriPos, where TriPos indicates the position at which the output line is to become active.
2. If you want to enable the conditions for trigger output, send TRO <TrigOutID> 1.

#### Example:

Digital output line 1 is to become active when the absolute position of axis 1 is at least 1.5 mm.

➤ Send:

```
CTO 1 2 1
```

```
CTO 1 3 8
```

```
CTO 1 10 1.5
```

### 8.3.8 Setting Signal Polarity

The polarity of the signal at the digital output which is used for triggering can be selected with the *Polarity* CTO parameter. The polarity can have the following values:

- active high = 1 (default setting)
- active low = 0
- Configure the digital output line (<TrigOutID>) that is to be used as the trigger output:

Send `CTO <TrigOutID> 7 P`, where *P* indicates the polarity.

Example:

The signal polarity for digital output line 1 is to be set to active low.

- Send:

`CTO 1 7 0`

## 8.4 Digital Input Signals

The digital inputs of the E-871 are available on the **I/O** socket (p. 310).

- Get the number of the input lines available on the E-871 with the `TIO?` command (p. 248).
- Get the state of the input lines with the `DIO?` command (p. 177).

Potential applications:

- Use in macros (p. 114). Details and examples of macros can be found in "Controller Macros" (p. 131).
- Use as switch signals (p. 114)
- Use for the HID control. Details can be found in "Control with an HID Device" (p. 118).

#### INFORMATION

The digital inputs (pins 1 to 4) on the **I/O** socket can also be used as analog inputs.

- Digital: TTL
- Analog: 0 to +5 V

## 8.4.1 Commands and Parameters for Digital Inputs

### Commands

The following commands are available for the use of digital inputs:

Command	Syntax	Function
CPY	CPY <Variable> <CMD?>	Copies, in combination with the DIO? query command, the state of a digital input line to a variable. Use in macros to set local variables (p. 153).
DIO?	DIO? [{<DIOID>}]	Gets the state of the digital input lines.
FED	FED {<AxisID> <EdgeID> <Param>}	Starts a move to a signal edge. The signal source can be a digital input line.
FNL	FNL [{<AxisID>}]	Starts a reference move to the negative physical limit of the travel range. A digital input line can be used as the source of the negative limit switch signal instead of the negative limit switch.
FPL	FPL [{<AxisID>}]	Starts a reference move to the positive physical limit of the travel range. A digital input line can be used as the source of the positive limit switch signal instead of the positive limit switch.
FRF	FRF [{<AxisID>}]	Starts a reference move to the reference point switch. A digital input line can be used as the source of the reference switch signal instead of the reference point switch.
JRC	JRC <Jump> <CMD?> <OP> <Value>	Can only be used in macros. Triggers a relative jump of the macro execution pointer depending on the state of a digital input line when used in combination with the DIO? query command.
MEX	MEX <CMD?> <OP> <Value>	Can only be used in macros. Stops the macro execution depending on the state of a digital input line when used in combination with the DIO? query command.
WAC	WAC <CMD?> <OP> <Value>	Can only be used in macros. Waits until a digital input line reaches a certain state when used in combination with the DIO? query command.

For the commands for HID control, see "Commands and Parameters for HID Devices" (p. 119).

## Parameters

The following parameters are available for the configuration of digital inputs:

Parameter	Description and possible values
<b>Source Of Reference Signal</b> 0x5C	Specifies the source of the reference signal for the <b>FRF</b> and <b>FED</b> commands: 0 = reference point switch 1 = digital input 1 2 = digital input 2 3 = digital input 3 4 = digital input 4
<b>Source Of Negative Limit Signal</b> 0x5D	Specifies the source(s) of the negative limit switch signal for the <b>FNL</b> and <b>FED</b> commands via a bitmask: 0 = Negative limit switch (default setting) 1 = Digital input 1 (bit 0) 2 = Digital input 2 (bit 1) 4 = Digital input 3 (bit 2) 8 = Digital input 4 (bit 3)
<b>Source Of Positive Limit Signal</b> 0x5E	Specifies the source(s) of the positive limit switch signal for the <b>FPL</b> and <b>FED</b> commands via a bitmask: 0 = Positive limit switch (default setting) 1 = Digital input 1 (bit 0) 2 = Digital input 2 (bit 1) 4 = Digital input 3 (bit 2) 8 = Digital input 4 (bit 3)
<b>Invert Digital Input Used For Negative Limit</b> 0x5F	Inverts the polarity of the digital inputs that are used for the source of the negative limit switch signal via a bitmask: 0 = No digital input inverted (default setting). 1 = Digital input 1 inverted (bit 0) 2 = Digital input 2 inverted (bit 1) 4 = Digital input 3 inverted (bit 2) 8 = Digital input 4 inverted (bit 3)

Parameter	Description and possible values
<b><i>Invert Digital Input Used For Positive Limit</i></b> 0x60	Inverts the polarity of the digital inputs that are used for the source of the positive limit switch signal via a bitmask: 0 = No digital input inverted (default setting). 1 = Digital input 1 inverted (bit 0) 2 = Digital input 2 inverted (bit 1) 4 = Digital input 3 inverted (bit 2) 8 = Digital input 4 inverted (bit 3)

### 8.4.2 Using Digital Input Signals in Macros

The digital inputs on the **I/O** socket can be used in macros as follows:

- Conditional execution of the macro
- Conditional stopping of the macro execution
- Conditional jump of the macro execution pointer
- Copying the input state to a variable

Further information and examples can be found in "Controller Macros" (p. 131).

#### **INFORMATION**

You can connect the C-170.PB pushbutton box from PI to the **I/O** socket (p. 310) to generate the digital input signals for use in macros. It also displays via LEDs the state of the digital output lines.

### 8.4.3 Using Digital Input Signals as Switch Signals

The digital inputs on the **I/O** socket can be used as the source of reference point and limit switch signals (e. g. for reference moves (p. 37)) for an axis.

#### **Using digital input as reference signal**

#### **INFORMATION**

The level of the digital input signal which you use instead of the reference point switch may only change once across the entire travel range.

- Use a suitable signal source.
- If necessary, invert the signal logic of the digital input line by setting the ***Invert Reference?*** parameter (ID 0x31) accordingly.



**INFORMATION**

The **Has Reference?** parameter (ID 0x14) has no bearing on the use of a digital input line as the source of the reference signal.

- Select the source of the reference signal for the axis by changing the **Source Of Reference Signal** parameter (ID 0x5C).

Detailed information about changing parameters can be found in "Adapting Settings" (p. 271).

**Using digital inputs as source of the limit switch signals****INFORMATION**

Several digital inputs can be selected as the source for a limit switch signal.

If a limit switch signal is used for reference moves, only one digital input line may be selected as the source of the limit switch signal.

**INFORMATION**

The level of the digital input signal which you use instead of an internal limit switch may only change once across the entire travel range.

- Use suitable signal sources.
- If necessary, invert the signal logic of the digital input lines by setting parameters **Invert Digital Input Used For Negative Limit** (ID 0x5F) and **Invert Digital Input Used For Positive Limit** (ID 0x60) accordingly.

**INFORMATION**

The **Has No Limit Switches?** parameter (ID 0x32) determines whether the E-871 evaluates the signals from the internal limit switches of the stage. This parameter has no bearing on the use of digital input lines as the source of the limit switch signal.

- Select the source(s) of the negative limit switch signal for the axis by changing the **Source Of Negative Limit Signal** parameter (ID 0x5D).
- Select the source(s) of the positive limit switch signal for the axis by changing the **Source Of Positive Limit Signal** parameter (ID 0x5E).

Detailed information about changing parameters can be found in "Adapting Settings" (p. 271).

**Example:**

Digital input lines 1, 3 and 4 are to be used for axis 1 as the sources of the positive limit switch signal. In addition, the signal polarity of lines 1 and 3 is to be inverted for axis 1. All adaptations are made only in the volatile memory of the E-871.

➤ Send:

`SPA 1 0x5E 13`, to select lines 1, 3 and 4.

`SPA 1 0x60 5`, to invert the signal polarity of lines 1 and 3.

## 8.4.4 Using Digital Input Signals for the HID Control

The digital inputs on the **I/O** socket can be used as axes of the HID device for the HID control:

- Axis 3: pins 1 and 2 (TTL signals)
- Axis 4: pins 3 and 4 (TTL signals)

These HID axes can be used to control the relative target position of the stage axis connected to the E-871.

Further information and examples can be found in "Connecting an HID Device" (p. 66) and "Control with an HID Device" (p. 118).

## 8.5 Analog Input Signals

The analog inputs of the E-871 are available on the **I/O** socket (p. 310).

- Get the number of analog input lines available on the E-871 with the `TAC?` command (p. 247).
- Query the voltage on the analog inputs with the `TAV?` command (p. 247).
- Use the data recorder (p. 100) to record the analog input signals.

Potential applications:

- Use in macros (p. 118): Details and examples of macros are found in "Controller Macros" (p. 131).
- Scanning applications with PIMikroMove (see PIMikroMove manual)

**INFORMATION**

The analog inputs (pins 1 to 4) on the **I/O** socket can also be used as digital inputs.

- Analog: 0 to +5 V
- Digital: TTL

### 8.5.1 Commands for Analog Inputs

The following commands are available for the use of analog inputs:

Command	Syntax	Function
CPY	CPY <Variable> <CMD?>	Copies, in combination with the TAV? query command, the voltage value of an analog input line to a variable. Use in macros to set local variables (p. 153).
DRC	DRC {<RecTableID> <Source> <RecOption>}	Configures the data recorder. Analog input values can be recorded using record option 81.
JRC	JRC <Jump> <CMD?> <OP> <Value>	Can only be used in macros. Triggers a relative jump of the macro execution pointer depending on the voltage at an analog input line when used in combination with the TAV? query command.
MEX	MEX <CMD?> <OP> <Value>	Can only be used in macros. Stops macro execution depending on the voltage at an analog input line when used in combination with the TAV? query command.
TAC?	TAC?	Get the number of installed analog lines.
TAV?	TAV? [<AnalogInputID>]	Get voltage at analog input.
WAC	WAC <CMD?> <OP> <Value>	Can only be used in macros. Waits until an analog input line reaches a certain voltage when used in combination with the TAV? query command.

## 8.5.2 Using Analog Input Signals in Macros

The analog inputs on the **I/O** socket can be used in macros as follows:

- Conditional execution of the macro
- Conditional stopping of the macro execution
- Conditional jump of the macro execution pointer
- Copying the input state to a variable

Further information and examples can be found in "Controller Macros" (p. 131).

## 8.6 Control with an HID Device

### 8.6.1 Functionality of the HID Control

The axis of an HID device can control the following motion parameters of the stage axis connected to the E-871:

- **Absolute target position:** The relationship between the displacement of the axis of the HID device and the target position of the stage axis is created by the E-871 using a lookup table. For details on lookup tables, see the descriptions of the HDT (p. 195) and HIT (p. 206) commands.
- **Relative target position:** Intended for use with AB rotary encoders or pulse generators (p. 66). Each received pulse (if present: each mechanical detent) triggers a relative motion of the stage axis over the distance that is set with the SST command (p. 243).

For further details, see the description of the HIA command (p. 197).

When the HID control is disabled, the target position is set to the current position of the controlled axis of the E-871.

#### **INFORMATION**

Motion commands are not allowed when the HID control is enabled for the axis.  
HID control is not possible in open-loop operation (servo mode OFF).

#### **INFORMATION**

It is recommended to use PIMikroMove for setting up and enabling the HID control and for testing and calibrating the HID device.

The HID control and the use of the buttons of the HID device can be programmed e. g. with controller macros (p. 131). In this manual, you will find an example macro for the HID control alternating with relative motions (p. 145).

## 8.6.2 Commands and Parameters for HID Devices

### Commands

The following commands are available for the use of HID devices:

Command	Syntax	Function
HDT	HDT {<HIDDeviceID> <HIDDeviceAxis> <HIDTableID>}	Assigns a lookup table to the axis of an HID device.  The assignment can be saved in the nonvolatile memory with WPA.
HDT?	HDT? [{<HIDDeviceID> <HIDDeviceAxis>}]	Gets the current assignment of lookup tables to the axes of HID devices.
HIA	HIA {<AxisID> <MotionParam> <HIDDeviceID> <HIDDeviceAxis>}	Configures the control of the axis of the E-871 by axes of HID devices ("HID control").  The configuration can be saved in the nonvolatile memory with WPA.
HIA?	HIA? [{<AxisID> <MotionParam>}]	Gets the current configuration of the HID control.
HIB?	HIB? [{<HIDDeviceID> <HIDDeviceButton>}]	Gets the current state of the buttons of HID devices.
HIE?	HIE? [{<HIDDeviceID> <HIDDeviceAxis>}]	Gets the current displacement of the axes of HID devices.
HIN	HIN {<AxisID> <HIDControlState>}	Enables or disables the HID control for the axis of the E-871.
HIN?	HIN? [{<AxisID>}]	Gets the activation state of the HID control.
HIS?	HIS? [{<HIDDeviceID> <HIDItemID> <HIDPropID>}]	Gets the properties of the operating elements of HID devices.
HIT	HIT {<HIDTableID> <HIDTableAddr> <HIDTableValue>}	Fills lookup tables with values.  The table contents can be saved in the nonvolatile memory with WPA.

Command	Syntax	Function
HIT?	HIT? [<StartPoint> [<NumberOfPoints> [<HIDTableID>]]]	Gets the values of the points in the lookup tables.
SST	SST {<AxisID> <StepSize>}	Is only used when the relative target position is set as the motion parameter to be controlled.  Sets the distance to be covered per impulse received.
SST?	SST? [{<AxisID>}]	Gets the distance set with SST.

### Parameter

The following parameter is available for the HID control:

Parameters	Description and possible values
<b><i>Invert Direction Of Motion For Joystick-Controlled Axis?</i></b> 0x61	Specifies the direction of motion for the axis of the E-871 during HID control.  0 = direction of motion not inverted (default setting) 1 = direction of motion inverted

### 8.6.3 Testing an HID Device

After an HID device has been connected to the E-871, it is recommended to test the operating elements of the HID device in PIMikroMove.

#### INFORMATION

No stage has to be connected to the E-871 to test the operating elements of HID devices in PIMikroMove.

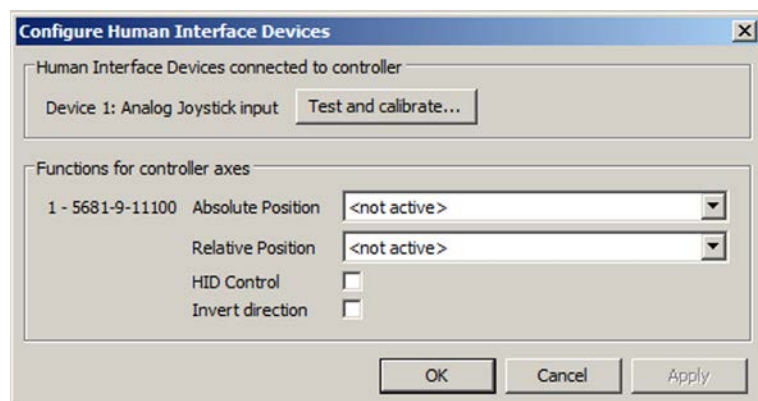
### Prerequisites

- ✓ You have read and understood the General Notes on Start-Up (p. 71).
- ✓ PIMikroMove is installed on the PC (p. 55).
- ✓ You have read and understood the PIMikroMove manual. The manual is found on the product CD.

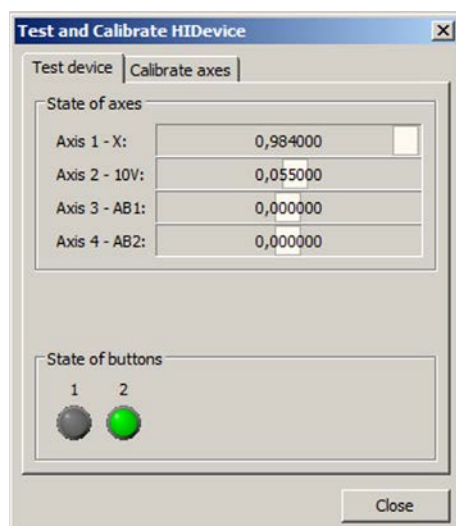
- ✓ You have established communication between the E-871 and the PC with PIMikroMove (p. 76).
- ✓ You have connected the HID device to the E-871 (p. 66).

### Testing the HID device in PIMikroMove

1. In the main window of PIMikroMove, open the **Configure Human Interface Devices** window via the **E-871 > Configure controller HIDevice(s)...** menu item.



2. Open the **Test and Calibrate HIDevice** window by clicking the **Test and calibrate...** button.
3. In the **Test and Calibrate HIDevice** window, select the **Test device** tab.
4. Test the operating elements of the HID device:
  - Move the axes of the HID device and observe the status displays in the **State of axes** area.
  - Press the buttons of the HID device and observe the status displays in the **State of buttons** area.



In this example, a C-819.20 joystick is connected to the **Joystick** socket of the E-871. The E-871 supports one axis of this joystick. The identifier of the axis is 1, the name is X. The two buttons of the C-819.20 joystick are available on the E-871 via the identifiers 1 and 2. Current status in the figure: The axis of the joystick is displaced in the positive direction, and button 2 is pressed.

### 8.6.4 Setting Up and Enabling the HID Control

It is recommended to use PIMikroMove for setting up and enabling the HID control. Before the HID control is enabled, it is recommended to test the connected HID device (p. 120).

#### **INFORMATION**

A total of 4 axes of an HID device can be connected to the **Joystick** (p. 311) and **I/O** (p. 310) sockets of the E-871. The axes of the HID device are suitable for controlling the following motion parameters of the stage axis connected to the E-871:

- Axes 1 and 2: absolute target position
- Axes 3 and 4: relative target position

Information on the connection options and suitable devices, see "Connecting an HID Device" (p. 66).



**INFORMATION**

It is not possible to simultaneously control the absolute and relative target position of the E-871 via the axes of the HID device.

- Configure the axis of the E-871 **either** for the HID control of the absolute target position **or** the relative target position.

**INFORMATION**

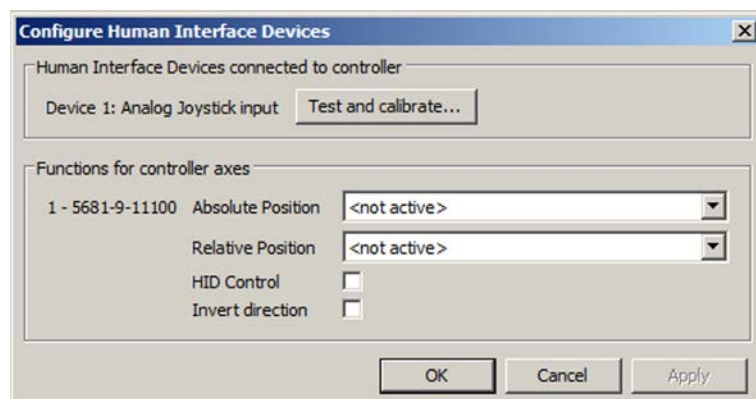
In PIMikroMove, the settings in the **Configure Human Interface Devices** window go into effect when the **Apply** button is clicked or when the window is closed by clicking the **OK** button.

**Prerequisite**

- ✓ You have carried out a successful reference move for each axis of the E-871 with PIMikroMove; see "Starting Motions" (p. 84).
- ✓ You have connected the HID device to the E-871 (p. 66).
- ✓ All devices are still ready for operation.

**Setting up and enabling the HID control in PIMikroMove**

1. In the main window of PIMikroMove, open the **Configure Human Interface Devices** window via the **E-871 > Configure controller HIDevice(s)...** menu item.



2. In the **Functions for controller axes** area of the **Configure Human Interface Devices** window, set up the HID control for the axis of the E-871:
  - a) In the **Absolute Position** field or the **Relative Position** field, select the axis of the HID device to be used for the motion parameter to be controlled.

- b) Enable the HID control by marking the **HID Control** check box.
  - c) If the direction of motion is to be inverted during the HID control, mark the **Invert direction** check box.
3. Send the settings for setting up the HID control to the E-871 by clicking the **OK** button.

The **Configure Human Interface Devices** window closes.

4. In PIMikroMove, make sure that the servo mode is switched on for the axis of the E-871 (e. g. by marking the **Servo** check box on the **Axes** tab in the main window of PIMikroMove).

The axis of the E-871 can now be controlled by the axis of the HID device in accordance with the settings made in step 2.

If the HID control of the absolute target position with axis 1 or 2 of the HID device does not work satisfactorily:

- Follow the instructions in "Calibrating the Axes of HID Devices" (p. 124).

If you want to save the assignment of the axis of the HID device to the controlled motion parameter in the nonvolatile memory of the E-871:

- Follow the instructions in "Permanently Saving the Configuration of the HID Control" (p. 127).

## 8.6.5 Calibrating Axes of HID Devices

Axes 1 and 2 of the HID device are connected to the **Joystick** socket and are intended for controlling the absolute target position of the axis of the E-871.

Calibration is required for axes 1 and 2 of the HID device in the following cases:

- Once the HID control is enabled, the stage moves even though you are not operating the axis of the HID device.
- You have connected the Z axis of the C-819.30 joystick to the **Joystick** socket of the E-871.

Calibration involves the following steps:

- If corresponding operating elements are present on the HID device: mechanical adjustment of the axis.
- Calibration of the axis of the HID device in PIMikroMove

**INFORMATION**

No mechanical adjustment is possible for the Z axis of the C-819.30 joystick.

- Calibrate the Z axis of the joystick after connecting it to the E-871 with PIMikroMove.

**Prerequisites**

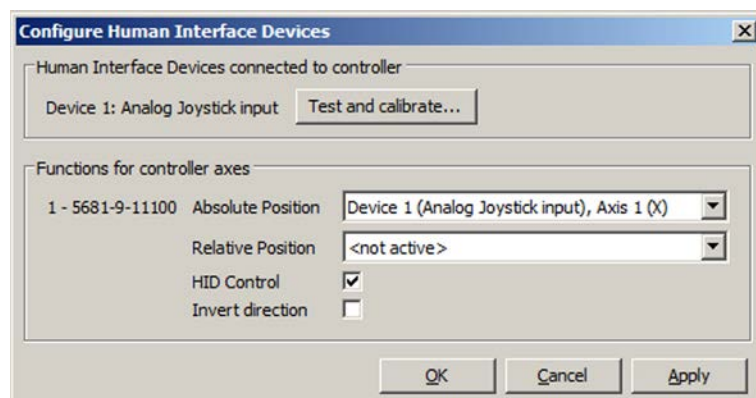
- ✓ You have set up and enabled the HID control in PIMikroMove (p. 122).
- ✓ All devices are still ready for operation.

**Mechanically adjusting an axis of the HID device**

- Check whether the axis of the HID device is locked mechanically and disengage the lock if necessary.
- Keep the affected axis of the HID device in center position and adjust it with the appropriate operating elements until the stage is no longer moving. With the C-819.20 and C-819.30 joysticks, turn the corresponding rotary knob for adjustment.

**Calibrating the axis of an HID device in PIMikroMove**

1. In the main window of PIMikroMove, open the **Configure Human Interface Devices** window via the **E-871 > Configure controller HIDevice(s)...** menu item.



The figure shows an example in which a C-819.20 joystick is connected to the E-871. One axis of this joystick controls the absolute target position of the axis of the E-871. The identifier of the joystick axis is 1, the name is X.

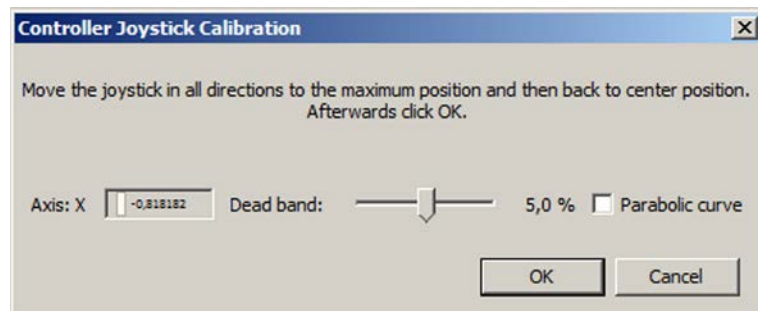
2. Open the **Test and Calibrate HIDevice** window by clicking the **Test and calibrate...** button.

3. In the **Test and Calibrate HIDevice** window, select the **Calibrate axes** tab.
4. Assign a lookup table with the name **User Table** to the axis of the HID device to be calibrated in the corresponding selection field on the **Calibrate axes** tab.



In the example in the figure, a user-defined lookup table was assigned to the axis of the connected C-819.20 joystick. The predefined parabolic lookup table is assigned to each of the axes 2 to 4.

5. Calibrate the axis of the HID device by filling the assigned user-defined lookup table with values:
  - a) Click the corresponding **Calibrate...** button to open the **Controller Joystick Calibration** window.
  - b) Move the axis of the HID device to all extreme positions. In this way, custom lookup table values are determined.
  - c) Let go of the axis.
  - d) If you want to change the neutral area of the axis (i. e. the area around the middle position of the axis in which no change in the controlled motion parameter is triggered), set the **Dead band** slider in the **Controller Joystick Calibration** window correspondingly.
  - e) If the values in the user-defined lookup table are to describe a parabolic curve shape, mark the **Parabolic curve** check box in the **Controller Joystick Calibration** window.
  - f) Click **OK** in the **Controller Joystick Calibration** window to write the lookup table values to the volatile memory of the E-871. You can observe the writing process in a separate window.



The window for the writing process and the **Controller Joystick Calibration** window automatically close after the writing process has ended.

6. If you want to save the assignment of the lookup tables to the axes of the HID device and the contents of user-defined lookup tables in the nonvolatile memory of the E-871:
  - a) Close the **Test and Calibrate HIDevice** window.
  - b) If necessary, adapt the settings in the **Configure Human Interface Devices** window to your application; see "Setting up and Enabling the HID Control" (p. 122).
  - c) If necessary, click the **Apply** button to enable the settings in the **Configure Human Interface Devices** window.
  - d) Close the **Configure Human Interface Devices** window.
  - e) Follow the instructions in "Permanently Saving the Configuration of the HID Control" (p. 127).

### 8.6.6 Permanently Saving the Configuration of the HID Control

The following settings for the configuration of the HID control can be saved in the nonvolatile memory of the E-871:

- Assignment of lookup tables to the axes of the HID device; see "Calibrating Axes of HID Devices" (p. 124)
- Contents of user-defined lookup tables, see "Calibrating Axes of HID Devices" (p. 124)
- Assignment of axes of the HID device to the motion parameters to be controlled for the axis of the E-871, see "Setting up and Enabling the HID Control" (p. 122)

These settings can only be saved together – a specific selection is **not** possible during saving.

**INFORMATION**

The values in the nonvolatile memory are loaded to the volatile memory as default values when the E-871 is switched on or rebooted and take effect immediately.

**Prerequisites**

- ✓ You have read and understood the General Notes on Start-Up (p. 71).
- ✓ PIMikroMove is installed on the PC (p. 55).
- ✓ You have read and understood the PIMikroMove manual. The manual is found on the product CD.
- ✓ You have established communication between the E-871 and the PC with PIMikroMove (p. 76).

**Permanently saving the configuration of the HID control in PIMikroMove**

If you want to write the current settings for the configuration of the HID control to the nonvolatile memory of the E-871:

1. In the main window of PIMikroMove, select the **E-871 > Save parameters to non-volatile memory** menu item. The **Save Parameters to Non-Volatile Memory** dialog opens.
2. In the selection field of the **Save Parameters to Non-Volatile Memory** dialog, enter either the password *HID* or select the entry *Settings of HDT, HIA, HIT (HID)*.
3. Click **OK** to save and close the dialog.

**INFORMATION**

The settings for the configuration of the HID control are also written to the nonvolatile memory of the E-871 when you select the *All Parameters*, *Settings of HDT, HIA, HIT (100)* entry or enter the password *100*. The entry or the password 100 also saves the current values of all parameters of the E-871, however; see the description of the WPA command (p. 256) and "Adapting Settings" (p. 271).

### 8.6.7 Available HID devices

PI provides the HID devices described in the following as optional accessories (p. 13).

#### Analog C-819.20 joystick, 2 axes

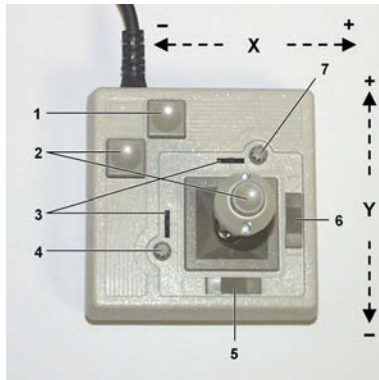


Figure 20: C-819.20 joystick

- 1 Pushbutton for the X axis
- 2 Pushbutton for the Y axis
- 3 Adjustment indicator
- 4 Rotary knob for adjustment of the Y axis (calibration)
- 5 X axis lock
- 6 Y axis lock
- 7 Rotary knob for adjustment of the X axis (calibration)

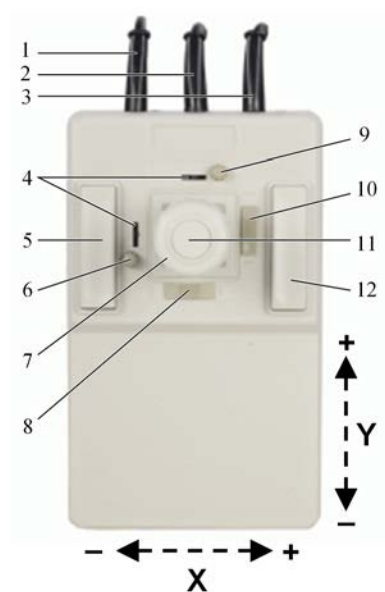
**C-819.30 analog joystick, 3 axes**

Figure 21: C-819.30 joystick

- 1 Cable for the Z axis
- 2 Cable for the Y axis
- 3 Cable for the X axis
- 4 Adjustment indicator
- 5 Pushbutton for the Y axis
- 6 Rotary knob for adjustment of the Y axis (calibration)
- 7 XY control lever with rotary knob for Z axis
- 8 X axis lock
- 9 Rotary knob for adjustment of the X axis (calibration)
- 10 Y axis lock
- 11 Pushbutton for the Z axis
- 12 Pushbutton for the X axis





Figure 22: C-819.30 joystick, rotary knob for the Z axis

## 8.7 Controller Macros

### 8.7.1 Overview: Macro Functionalities and Example Macros

The E-871 can save and process command sequences as macros.

The following functionalities make macros an important tool in many application areas:

- Several macros can be saved at the same time.
- Any macro can be defined as the start-up macro. The start-up macro is executed each time that the E-871 is switched on or rebooted.
- Processing a macro and stopping macro execution can be linked to conditions. In this way, loops can be realized as well.
- Macros can call up themselves or other macros in several nesting levels.
- Variables (p. 153) can be set for the macro and in the macro itself and used in different operations.
- Input signals can be evaluated for conditions and variables.

In this manual, you will find example macros for the following tasks:

- Moving an axis back and forth (p. 136)
- Recording a macro for a controller whose address is different from 1 (p. 137)
- Moving an axis with a variable travel back and forth (p. 139)
- Implementing multiple calls of a macro via a loop (p. 140)
- Preparing an axis via start-up macro for closed-loop operation
- Stopping motion by pushbutton (p. 143)
- HID control alternating with relative motions (p. 145)

## 8.7.2 Commands and Parameters for Macros

### Commands

The following commands are specially available for handling macros or for use in macros:

Com-mand	Syntax	Function
ADD (p. 163)	ADD <Variable> <FLOAT1> <FLOAT2>	Adds two values and saves the result to a variable (p. 153). Can only be used for local variables in macros.
CPY (p. 167)	CPY <Variable> <CMD?>	Copies a command response to a variable (p. 153). Can only be used for local variables in macros.
DEL (p. 173)	DEL <uint>	Can only be used in macros. Delays <uint> milliseconds.
JRC (p. 213)	JRC <Jump> <CMD?> <OP> <Value>	Can only be used in macros. Triggers a relative jump of the macro execution pointer depending on a condition.
MAC (p. 216)	MAC BEG <macroname>	Starts the recording of a macro with the name <i>macroname</i> on the controller. <i>macroname</i> can consist of up to 8 characters.
	MAC DEF <macroname>	Defines the given macro as the start-up macro.
	MAC DEF?	Gets the start-up macro.
	MAC DEL <macroname>	Deletes the given macro.
	MAC END	Ends the macro recording.
	MAC ERR?	Reports the last error which occurred during macro execution.
	MAC NSTART <macroname> <uint> [<String1> [<String2>]]	Starts the given macro n times in succession (n = number of executions). The values of local variables can be set for the macro with <String1> and <String2>.
	MAC START <macroname> [<String1> [<String2>]]	Starts one execution of the specified macro. The values of local variables can be set for the macro with <String1> and <String2>.

Com- mand	Syntax	Function
MAC? (p. 219)	MAC? [<macroname>]	Lists all macros or the content of a given macro.
MEX (p. 221)	MEX <CMD?> <OP> <Value>	Can only be used in macros. Stops the macro execution depending on a condition.
RMC? (p. 228)	RMC?	Lists macros which are currently running.
VAR (p. 252)	VAR <Variable> <String>	Sets a variable (p. 153) to a certain value or deletes it. Can only be used for local variables in macros.
VAR? (p. 254)	VAR ? [{<Variable>}]	Gets variable values.
WAC (p. 255)	WAC <CMD?> <OP> <Value>	Can only be used in macros. Waits until a condition is met.
#8 (p. 161)	-	Tests if a macro is running on the controller.

### Parameter

The following parameter is available for working with macros:

Parameter	Description and possible values
<b>Ignore Macro Error?</b> 0x72	Determines whether the controller macro is stopped when an error occurs while it is being executed. <ul style="list-style-type: none"> <li>0 = Stop macro when error occurs (default)</li> <li>1 = Ignore error</li> </ul>

## 8.7.3 Working with Macros

Work with macros comprises the following:

- Recording of macros (p. 134)
- Starting macro execution (p. 138)
- Stopping macro execution (p. 141)
- Setting up a start-up macro (p. 141)
- Deleting of macros (p. 143)

**INFORMATION**

For working with controller macros, it is recommended to use the **Controller macros** tab in PIMikroMove. There you can conveniently record, start and manage controller macros. Details are found in the PIMikroMove manual.

**Recording a macro****INFORMATION**

The E-871 can save up to 32 macros simultaneously. A maximum of 5 nesting levels are possible in macros.

**INFORMATION**

Basically all GCS commands (p. 149) can be included in a macro. Exceptions:

- `RBT` for rebooting the E-871
- `MAC BEG` and `MAC END` for macro recording
- `MAC DEL` for deleting a macro

Query commands can be used in macros in combination with the `CPY`, `JRC`, `MEX` and `WAC` commands. Otherwise they have no effect, since macros do not send any responses to interfaces.

**INFORMATION**

If you record a macro on a E-871 whose controller address differs from 1, note the following when entering the commands that are to be an element of the macro:

- If you are working with PITerminal and have established communication with the **Connect...** button, the target address has to be typed in in every command line.
- If you are working with PIMikroMove or have established communication with PITerminal using the **GCS DLL...** button, the target address is automatically sent and may not be typed in.

**INFORMATION**

To make the use of macros more flexible, you can use local and global variables in macros. See "Variables" (p. 153) for more information.

**INFORMATION**

The number of write cycles in the nonvolatile memory is restricted by the limited lifetime of the memory chip.

- Only record macros when it is necessary.
- Use variables (p. 153) in macros to make macros more flexible, and give the corresponding variable values when starting macro execution.
- Contact our customer service department (p. 301) if the E-871 shows unexpected behavior.

**INFORMATION**

A macro is overwritten if a macro with the same name is re-recorded.

1. Start the macro recording.
  - If you are working with PITerminal or in the **Command entry** window of PIMikroMove: Send the `MAC BEG macroname` command, where *macroname* indicates the name of the macro.
  - If you are working in PIMikroMove on the **Controller macros** tab: Click the **Create new empty macro** icon to create a tab for entering a new macro. Do **not** enter the `MAC BEG macroname` command.
2. Enter the commands to be included in the *macroname* macro line by line, using the normal command syntax.
 

Macros can call up themselves or other macros in several nesting levels.
3. End the macro recording.
  - If you are working with PITerminal or in the **Command entry** window of PIMikroMove: Send the `MAC END` command.
  - If you are working in PIMikroMove on the **Controller macros** tab: Do **not** enter the `MAC END` command. Click the **Send macro to controller** icon and enter the name of the macro in a separate dialog window.

The macro has been stored in the nonvolatile memory of the E-871.
4. If you want to check whether the macro has been correctly recorded:
 

If you are working with PITerminal or in the **Command entry** window of PIMikroMove:

  - Get which macros are saved in the E-871 by sending the `MAC?` command.

- Get the contents of the *macroname* macro by sending the `MAC? macroname` command.

If you are working in PIMikroMove on the **Controller macros** tab:

- Click the **Read list of macros from controller** icon.
- Mark the macro to be checked in the list on the left side and click the **Load selected macro from controller** icon.

#### Example: Moving an axis back and forth

##### INFORMATION

When macros are recorded on the **Controller macros** tab in PIMikroMove, the `MAC BEG` and `MAC END` commands must be omitted.

The axis 1 is to move back and forth. For this purpose, 3 macros are recorded. Macro 1 starts the motion in the positive direction and waits until the axis has reached the target position. Macro 2 performs this task for the negative direction of motion. Macro 3 calls up macro 1 and 2.

- Record the macros by sending:

```
MAC BEG macro1
MVR 1 12.5
WAC ONT? 1 = 1
MAC END
MAC BEG macro2
MVR 1 -12.5
WAC ONT? 1 = 1
MAC END
MAC BEG macro3
MAC START macro1
MAC START macro2
MAC END
```

**Example: Recording macro for controller whose address is different from 1****INFORMATION**

When macros are recorded on the **Controller macros** tab in PIMikroMove, the `MAC` `BEG` and `MAC` `END` commands must be omitted.

The controller address is set to 2 via the DIP switches. In this example, macro recording is done using PITerminal, whereby communication was established with the **Connect...** button (as a result, the target address has to be typed in in every command line).

The servo mode is to be switched on for axis 1 via the ref macro, and a reference move to the reference point switch is to be started.

1. Record the macro by sending:

```
2 MAC BEG ref
```

```
2 SVO 1 1
```

```
2 DEL 1000
```

```
2 FRF 1
```

```
2 MAC END
```

2. Check the content of the ref macro by sending:

```
2 MAC? ref
```

The response reads:

```
0 2 SVO 1 1
```

```
DEL 1000
```

```
FRF 1
```

The first line of the response contains the target and sender address corresponding to the GCS syntax for multiline responses. However, the target address is not included in the macro.

## Starting a macro execution

### INFORMATION

Any commands can be sent from the command line when a macro is running on the controller. The macro content and move commands received from the command line can overwrite each other.

### INFORMATION

Simultaneous execution of multiple macros is not possible. Only one macro can be executed at a time.

### INFORMATION

You can link the macro execution to conditions with the `JRC` and `WAC` commands. The commands must be included in the macro.

In the following, PITerminal or the **Command entry** window of PIMikroMove is used to enter commands. Details on working with the **Controller macros** tab in PIMikroMove are found in the PIMikroMove manual.

1. If the macro execution is to be continued despite the occurrence of an error:
  - Set the **Ignore Macro Error?** parameter (ID 0x72) correspondingly: Send the `SPA 1 0x72 Status` command, whereby *Status* can have the value 0 or 1 (0 = Stop macro when error occurs (default); 1 = Ignore macro error).

Further information on changing parameters can be found in "Adapting Settings" (p. 271).

2. Start the macro execution:
  - If the macro is to be executed once, send the `MAC START macroname string` command, whereby *macroname* indicates the name of the macro.
  - If the macro is to be executed *n* times, send the `MAC NSTART macroname n string` command, whereby *macroname* indicates the name of the macro and *n* indicates the number of executions.



*string* stands for the values of local variables. The values only have to be given when the macro contains corresponding local variables. The sequence of the values in the input must correspond to the numbering of the appropriate local variables, starting with the value of the local variable 1. The individual values must be separated from each other with spaces.

3. If you want to check the macro execution:
  - Get whether a macro is being executed on the controller by sending the `#8` command.
  - Get the name of the macro that is currently being executed on the controller by sending the `RMC?` command.

#### Example: Moving an axis with a variable travel distance back and forth

##### INFORMATION

When macros are recorded on the **Controller macros** tab in PIMikroMove, the `MAC` `BEG` and `MAC` `END` commands must be omitted.

The axis 1 is to move back and forth. The travel to the left and to the right is to be variably adjustable without having to change the used macros. Local and global variables are therefore used.

1. Create the global variables LEFT and RIGHT by sending:

```
VAR LEFT 5
```

```
VAR RIGHT 15
```

LEFT thus has the value 5, and RIGHT has the value 15. These values can be changed at any time, e.g., by sending the `VAR` command again.

- Create the global variables again each time that the E-871 is switched on or rebooted, since they are only written to the volatile memory of the E-871.
2. Record the MOVLR macro by sending:

```
MAC BEG movlr
```

```
MAC START movwai ${LEFT}
```

```
MAC START movwai ${RIGHT}
```

```
MAC END
```

MOVLr successively starts the MOVWAI macro (which is still to be recorded) for both directions of motion. The values of the global variables LEFT and RIGHT are used when MOVWAI is started, to set the value of the local variable 1 contained in MOVWAI (dollar signs and braces are necessary for the local variable 1 in the macro to actually be replaced by the *value* of the global variable and not by its *name*).

3. Record the MOVWAI macro by sending:

```
MAC BEG movwai
MOV 1 $1
WAC ONT? 1 = 1
MAC END
```

MOVWAI moves axis 1 to the target position which is given by the value of the local variable 1 and waits until the axis has reached the target position.

4. Start the execution of the MOVLr macro by sending:

```
MAC NSTART movlr 5
```

The MOVLr macro is executed five times in succession, i.e., axis 1 alternately moves to the positions 5 and 15 five times. You can also select any other value for the number of executions.

### Example: Implementing multiple calls of a macro via a loop

#### INFORMATION

When macros are recorded on the **Controller macros** tab in PIMikroMove, the `MAC BEG` and `MAC END` commands must be omitted.

The TESTDION macro checks the status of the digital input lines on the **I/O** socket. It uses a local variable to identify the digital input line (1 to 4). So that the TESTDION macro does not have to be called separately for each input line, another macro with a loop is recorded.

- Record the LOOPDION macro by sending:

```
MAC BEG loopdion
VAR COUNTER 1
MAC START TESTDION ${COUNTER}
```

```
ADD COUNTER ${COUNTER} 1
JRC -2 VAR? COUNTER < 5
MAC END
```

The COUNTER variable is created with the value 1. After this, the TESTDION macro is started for the input line whose identifier is specified via the COUNTER variable. Then the value of the COUNTER is increased by 1. As long as the value of the COUNTER is less than 5, the macro execution pointer subsequently jumps two lines back, so that the TESTDION is now started for the next digital input line.

## Stopping a macro execution

### INFORMATION

You can link the stopping of the macro execution to a condition with the `MEX` command. The command must be included in the macro.

In the following, PITerminal or the **Command entry** window of PIMikroMove is used to enter commands. Details on working with the **Controller macros** tab in PIMikroMove are found in the PIMikroMove manual.

- Stop the macro execution with the `#24` or `STP` commands.
- If you want to check whether an error has occurred during the macro execution, send the `MAC ERR?` command. The response shows the last error that has occurred.

## Setting up a start-up macro

Any macro can be defined as the start-up macro. The start-up macro is executed each time that the E-871 is switched on or rebooted.

### INFORMATION

Deleting a macro does not delete its selection as the start-up macro.

In the following, PITerminal or the **Command entry** window of PIMikroMove is used to enter commands. Details on working with the **Controller macros** tab in PIMikroMove are found in the PIMikroMove manual.

- Define a macro as the start-up macro with the `MAC DEF macroname` command, whereby *macroname* indicates the name of the macro.
- If you want to cancel the selection of the start-up macro and do not want to define another macro as the start-up macro, only send `MAC DEF`.
- Get the name of the currently defined start-up macro by sending the `MAC DEF?` command.

#### Example: Preparing an axis via a start-up macro for closed-loop operation

##### INFORMATION

When macros are recorded on the **Controller macros** tab in PIMikroMove, the `MAC BEG` and `MAC END` commands must be omitted.

The STARTCL macro switches the HID control off and the servo mode on for axis 1 and starts a reference move to the negative physical limit of the travel range. As STARTCL is defined as the start-up macro, axis 1 is ready for closed-loop operation immediately after switching-on.

- Send:

```
MAC BEG startcl
HIN 1 0
SVO 1 1
DEL 1000
FNL 1
MAC END
MAC DEF startcl
```

##### INFORMATION

When this macro is used, the parameter settings of the E-871 should be adapted to the connected stage in the nonvolatile memory. Alternatively, the parameter settings can also be configured in the volatile memory via the start-up macro. Further information see "Adapting Settings" (p. 271).

## Deleting a macro

### INFORMATION

A running macro may not be deleted.

In the following, PITerminal or the **Command entry** window of PIMikroMove is used to enter commands. Details on working with the **Controller macros** tab in PIMikroMove are found in the PIMikroMove manual.

- Delete a macro with the `MAC DEL macroname` command, whereby *macroname* indicates the name of the macro.

## 8.7.4 Macro Example: Stopping Motion by Pushbutton

### INFORMATION

You can connect the C-170.PB pushbutton box from PI to the **I/O** socket to generate the digital input signals for use in macros. It also displays via LEDs the state of the digital output lines.

### INFORMATION

When macros are recorded on the **Controller macros** tab in PIMikroMove, the `MAC BEG` and `MAC END` commands must be omitted.

Action	Command	Result
Connect digital input line 1 on the <b>I/O</b> socket to an appropriate signal source.	- Pin assignment see "I/O" (p. 310).	The digital input signal can be used for a conditional jump of the macro execution pointer, for example.

Action	Command	Result
Record the HALT macro on the controller.	<pre>MAC BEG halt MVR 1 5 JRC 2 DIO? 1 = 1 JRC -1 ONT? 1 = 0 HLT 1 MAC END</pre>	<p>The macro has the following tasks:</p> <ul style="list-style-type: none"> <li>Start relative motion of axis 1</li> <li>Set condition: If digital input line 1 has the high state (when using the pushbutton box: button 1 is pressed), the macro execution pointer jumps two lines forward. The axis is stopped as a result. Otherwise macro execution is continued with the next line.</li> <li>Set condition: As long as axis 1 has not yet reached the target position, the macro execution pointer jumps back one line. A loop is established as a result.</li> </ul>
Start the HALT macro on the controller.	<pre>MAC START halt</pre>	Axis 1 starts to move. It is stopped by switching digital input line 1 to the high state (e.g. by pushbutton). Regardless of whether the axis has reached the target position or was halted previously, the error code is set to 10 via the <code>HLT</code> command.
If error code 10 interferes: Record alternative HALTVAR macro which uses a variable. Details see "Variables" (p. 153).	<pre>MAC BEG haltvar MVR 1 5 JRC 2 DIO? 1 = 1 JRC -1 ONT? 1 = 0 CPY TARGET POS? 1 MOV 1 \${TARGET} VAR TARGET MAC END</pre>	The macro has the same tasks as the HALT macro. However with pushbutton, axis 1 is not stopped via the <code>HLT</code> command, instead the result of the <code>POS? 1</code> query is copied to the TARGET variable. Then this variable is used as the target position for the <code>MOV</code> command. As a result, the axis stays right where it was. To clean up, TARGET is defined as empty with the <code>VAR</code> command which deletes the variable.
Start the HALTVAR macro on the controller.	<pre>MAC START haltvar</pre>	Axis 1 starts to move. It is stopped by switching digital input line 1 to the high state (e.g. by pushbutton). Error code 10 is not set because no halt or stop command is used.

### 8.7.5 Macro example: HID control alternating with relative motions

#### Task:

The identifier of axis 1 was changed in X with the SAI command (p. 232). The absolute target position of axis X is to be controlled with axis 1 of the HID device (e.g. an axis of the C-819.20 (p. 129) or C-819.30 (p. 130) joystick). The buttons of a connected C-170.PB (p. 13) pushbutton box are to be used for the following tasks:

- Button 1: Starting a relative motion in the positive direction when the HID control is disabled
- Button 2: Starting a relative motion in the negative direction when the HID control is disabled
- Button 3: Disabling of the HID control
- Button 4: Enabling of the HID control

#### Approach:

The macros STARTUP, LOOP, PBLOOP, BUTTON1, BUTTON2, BUTTON3 and BUTTON4 are recorded on the controller.

#### INFORMATION

When macros are recorded on the **Controller macros** tab in PIMikroMove, the **MAC** **BEG** and **MAC** **END** commands must be omitted.

Action	Command	Result
Connect C-170.PB pushbutton box from PI to the <b>I/O</b> socket.	-	Digital input lines 1 to 4 are switched to high state as long as the respective button is pressed.
Connect C-819.20 or C-819.30 joystick to the <b>Joystick</b> socket.	-	For commands, the connected joystick axis is accessible as axis 1 of HID device 1.

Action	Command	Result
Record the STARTUP macro on the controller.	<pre>MAC BEG startup HIN X 0 SVO X 1 FRF X WAC ONT? X = 1 HIA X 0 0 0 HIA X 1 1 1 HIN X 1 MAC START LOOP MAC END</pre>	<p>The macro has the following tasks:</p> <ul style="list-style-type: none"> <li>Switch on the servo mode for axis X.</li> <li>Start a reference move for axis X.</li> <li>Configure the HID control for axis X: The absolute target position is to be controlled with axis 1 of HID device 1.</li> <li>Enable the HID control for axis X</li> <li>Start the LOOP macro for the main loop</li> </ul>
Record the LOOP macro on the controller.	<pre>MAC BEG loop MAC START button3 MAC START loop MAC END</pre>	<p>The macro has the following tasks:</p> <ul style="list-style-type: none"> <li>Start BUTTON3 macro</li> <li>Call itself to set up the main loop</li> </ul>
Record BUTTON3 macro on the controller.	<pre>MAC BEG button3 MEX DIO? 3 = 0 HIN X 0 MAC START pbloop MAC END</pre>	<p>The macro has the following tasks:</p> <ul style="list-style-type: none"> <li>If button 3 is not pressed: end the execution of BUTTON3</li> <li>If button 3 is pressed: disable the HID control and start the loop for checking buttons 1, 2 and 4</li> </ul>
Record the PBLOOP macro on the controller.	<pre>MAC BEG pbloop MAC START button1 MAC START button2 MAC START button4 MAC START pbloop MAC END</pre>	<p>The macro has the following tasks:</p> <ul style="list-style-type: none"> <li>Start the BUTTON1, BUTTON2 and BUTTON4 macros in succession</li> <li>Call itself up to create the loop to check buttons 1, 2 and 4</li> </ul>
Record the BUTTON1 macro on the controller.	<pre>MAC BEG button1 MEX DIO? 1 = 0 MVR X 1 WAC ONT? X = 1 MAC END</pre>	<p>The macro has the following tasks:</p> <ul style="list-style-type: none"> <li>If button 1 is not pressed: end the execution of BUTTON1</li> <li>If button 1 is pressed: start a motion of axis X over distance 1 in the positive direction and pause the macro execution until axis X is at the target position</li> </ul>



Action	Command	Result
Record the BUTTON2 macro on the controller.	<pre>MAC BEG button2 MEX DIO? 2 = 0 MVR X -1 WAC ONT? X = 1 MAC END</pre>	<p>The macro has the following tasks:</p> <ul style="list-style-type: none"> <li>▪ If button 2 is not pressed: end the execution of BUTTON2</li> <li>▪ If button 2 is pressed: start a motion of axis X over distance 1 in the negative direction and pause the macro execution until axis X is at the target position</li> </ul>
Record the BUTTON4 macro on the controller.	<pre>MAC BEG button4 MEX DIO? 4 = 0 HIN X 1 MAC START LOOP MAC END</pre>	<p>The macro has the following tasks:</p> <ul style="list-style-type: none"> <li>▪ If button 4 is not pressed: end the execution of BUTTON4</li> <li>▪ If button 4 is pressed: enable the HID control and start the main loop</li> </ul>
Start the STARTUP macro on the controller.	<pre>MAC START startup</pre>	<p>Axis X starts a reference move to the reference point switch. The HID control is then enabled for axis X, so that the absolute target position can be controlled with the joystick. As long as the HID control is enabled, buttons 1 and 2 have no effect. The HID control is disabled by pressing button 3. Relative motions of axis X can then be started with buttons 1 and 2, and the HID control can be enabled again with button 4.</p>



## 9 GCS Commands

### In this Chapter

Notation .....	149
GCS Syntax for Syntax Version 2.0 .....	150
Target and Sender Address .....	152
Variables.....	153
Command Overview .....	155
Command Descriptions for GCS 2.0 .....	159
Error Codes .....	258

### 9.1 Notation

The following notation is used to define the GCS syntax and to describe the commands:

<...>	Angle brackets indicate an argument of a command, can be an item identifier or a command-specific parameter
[...]	Square brackets indicate an optional entry
{...}	Curly brackets indicate a repetition of entries, i.e. that it is possible to access more than one item (e.g. several axes) in one command line.
<span style="border: 1px solid black; padding: 0 2px;">LF</span>	LineFeed (ASCII char #10), is the default termination character (character at the end of a command line)
<span style="border: 1px solid black; padding: 0 2px;">SP</span>	Space (ASCII char #32), indicates a space character
"..."	Quotation marks indicate that the characters enclosed are returned or to be entered.

## 9.2 GCS Syntax for Syntax Version 2.0

A GCS command consists of 3 characters, e.g. CMD. The corresponding query command has a question mark added to the end, e. g. CMD?.

Command mnemonic:

CMD ::= character1 character2 character3 [?]

Exceptions:

- Single-character commands, e. g. fast query commands, consist only of one ASCII character. The ASCII character is written as combination of # and the character code in decimal format, e. g. as #24.
- \*IDN? (for GPIB compatibility).

The command mnemonic is not case-sensitive. The command mnemonic and all arguments (e. g. axis identifiers, channel identifiers, parameters, etc.) must be separated from each other by a space (SP). The command line ends with the termination character (LF).

CMD[{{SP}}<Argument>]]LF

CMD?[{{SP}}<Argument>]]LF

Exception:

- Single-character commands are not followed by a termination character. The response to a single-character commands is followed by a termination character, however.

The argument <AxisID> is used for the logical axes of the controller. Depending on the controller, an axis identifier can consist of up to 16 characters. All alphanumeric characters and the underscore are allowed. See "Commandable Items" (p. 15) for the identifiers supported by the E-871.

Example 1:

Axis 1 is to be moved to position 10.0. The unit depends on the controller (e. g. µm or mm).

Send: MOV SP1 SP10.0 LF

More than one command mnemonic per line is not allowed. Several groups of arguments following a command mnemonic are allowed.

Example 2:

Two axes which are connected to the same controller are to be moved:

Send: `MOV SP1 SP17.3 SP2 SP2.05 LF`

When a part of a command line cannot be executed, the line is not executed at all.

When all arguments are optional and are omitted, the command is executed for all possible argument values.

Example 3:

All parameters in the volatile memory are to be reset.

Send: `RPA LF`

Example 4:

The position of all axes is to be queried.

Send: `POS? LF`

The response syntax is as follows:

`[<Argument>[{SP}<Argument>]]"="<Value> LF`

With multi-line replies, the space preceding the termination character is omitted in the last line:

`{[<Argument>[{SP}<Argument>]]"="<Value> SP LF}`

`[<Argument>[{SP}<Argument>]]"="<Value> LF` for the last line!

In the response, the arguments are listed in the same order as in the query command.

Query command:

`CMD? SP<Arg3> SP<Arg1> SP<Arg2> LF`

Response to this command:

`<Arg3>"="<Val3> SP LF`

`<Arg1>"="<Val1> SP LF`

`<Arg2>"="<Val2> LF`

Example 5:

Send: `TSP? SP2 SP1 LF`

Receive: `2=-1158.4405SP LF`  
`1=+0000.0000LF`

### INFORMATION

With the E-871 only a single element per command line can be addressed (e.g. axis, channel or parameter).

Example:

By sending command line

```
SEP 100 1 0x32 0
```

a new value of parameter 0x32 is saved in nonvolatile memory for axis 1,  
 sending command line

```
SEP 100 1 0x32 0 1 0x14 1
```

is not possible, however, because two parameters are to be changed.

If the command supports this, all elements can be addressed by omitting the element identifier.

Example:

By sending command line

```
SEP?
```

all parameters from the nonvolatile memory are queried.

## 9.3 Target and Sender Address

In principle, the addresses of the target controller and the sender are required in every command line. This applies even to single-character commands like #4 or to macro recording. But because only the PC may send command lines to the controllers, its address (0) can be omitted. Both target and sender address, however, are part of any controller reply.

Example:

In a terminal, e.g. PITerminal, the identification string of a controller with address 2 (here: a C-863.11) is queried using the \*IDN? command.

Send: `2 0 *IDN?`

or

Send: `2 *IDN?`

The reply in either case is:

```
0 2 (c)2011 Physik Instrumente(PI) Karlsruhe, C-  
863.11,0,1.2.0.0
```

Exception:

The target address can be omitted if the target controller has the address 1, even if this controller is part of a daisy chain. If the target address is omitted when addressing a controller, the target and sender addresses will also be omitted in the reply of the controller.

Example:

Send: `*IDN?`

The controller with address 1 (here: a C-863.11) replies with:

```
(c)2011 Physik Instrumente(PI) Karlsruhe, C-863.11,0,1.2.0.0
```

Send: `1 *IDN?`

The same controller replies with:

```
0 1 (c)2011 Physik Instrumente(PI) Karlsruhe, C-  
863.11,0,1.2.0.0
```

See "Adapting the DIP Switch Settings" (p. 72) for how to set the controller address. The controller address can be in the range of 1 to 16, address 1 is default. The PC always has the address 0. With the broadcast address 255, all controllers in a daisy chain can be addressed at the same time, but no replies are sent to the PC.

## 9.4 Variables

For more flexible programming, the E-871 supports variables. While global variables are always available, local variables are only valid for a given macro. Typically, variables are used when working with macros.

Variables are present in volatile memory (RAM) only. The variable values are of the STRING data type.

The following conventions apply to variable names:

- Variable names must not contain special characters (especially no "\$").
- The maximum number of characters is 8.
- Names of global variables can consist of characters A to Z and 0 to 9. They must start with an alphabetic character.

- Names of local variables must not contain alphabetic characters. Possible characters are 0 to 9.
- The variable name can also be given via the value of another variable.

If the value of a variable is to be used, the notation must be as follows:

- The variable name must be preceded by the dollar sign (\$).
- Variable names consisting of multiple characters must be put in curly brackets.

If the variable name consists of a single character, the curly brackets can be omitted.

Note that if the curly brackets are omitted with variable names consisting of multiple characters, the first character after the "\$" is interpreted as the variable name.

#### Local variables:

- Local variables can only be used in macros.
- At present, the controller firmware supports three local variables: 0, 1 and 2.
- The values of the local variables 1 and 2 are given as arguments of the `MAC` `START` or `MAC NSTART` command when starting the macro.

The command formats are:

```
MAC START <macroname> [<String1> [<String2>]]
```

```
MAC NSTART <macroname> <uint> [<String1> [<String2>]]
```

<STRING1> and <STRING2> indicate the values for the local variables 1 and 2 used in the macro. <STRING1> and <STRING2> can be given directly or via the values of variables. <uint> defines the number of times the macro is to be run. See the `MAC` command (p. 216) description for more information.

- The local variable 0 is read-only. Its value gives the number of arguments (i. e. values of local variables) set when starting the macro.
- Inside a macro, the values of local variables can be modified using `ADD` (p. 163), `CPY` (p. 167) or `VAR` (p. 252), and can be deleted with `VAR` (except for the local variable 0).
- As long as the macro is running, the values of the local variables can be queried with

```
VAR? 0
```

```
VAR? 1
```

```
VAR? 2
```

The queries can be sent inside or outside of the macro.



**Global variables:**

- Global variables can be used inside and outside of macros.
- The maximum number of global variables is 10.
- Global variables are created and modified using **ADD**, **CPY** or **VAR**. They can be deleted with **VAR**.
- The variable values can be queried with **VAR?**.

## 9.5 Command Overview

Command	Format	Description
#4 (p. 159)	#4	Request Status Register
#5 (p. 160)	#5	Request Motion Status
#7 (p. 161)	#7	Request Controller Ready Status
#8 (p. 161)	#8	Query If Macro Is Running
#24 (p. 162)	#24	Stop All Axes
*IDN? (p. 162)	*IDN?	Get Device Identification
ADD (p. 163)	ADD <Variable> <FLOAT1> <FLOAT2>	Add and Save To Variable
CCL (p. 166)	CCL <Level> [<PSWD>]	Set Command Level
CCL? (p. 167)	CCL?	Get Command Level
CPY (p. 167)	CPY <Variable> <CMD?>	Copy Into Variable
CST? (p. 168)	CST? [{<AxisID>}]	Get Assignment Of Stages To Axes
CSV? (p. 169)	CSV?	Get Current Syntax Version
CTO (p. 169)	CTO {<TrigOutID> <CTOPam> <Value>}	Set Configuration Of Trigger Output
CTO? (p. 173)	CTO? [{<TrigOutID> <CTOPam>}]	Get Configuration Of Trigger Output
DEL (p. 173)	DEL <uint>	Delay The Command Interpreter
DFH (p. 174)	DFH [{<AxisID>}]	Define Home Position
DFH? (p. 176)	DFH? [{<AxisID>}]	Get Home Position Definition
DIO (p. 176)	DIO {<DIOID> <OutputOn>}	Set Digital Output Lines
DIO? (p. 177)	DIO? [{<DIOID>}]	Get Digital Input Lines

Command	Format	Description
DRC (p. 178)	DRC {<RecTableID> <Source> <RecOption>}	Set Data Recorder Configuration
DRC? (p. 179)	DRC? [{<RecTableID>}]	Get Data Recorder Configuration
DRL? (p. 180)	DRL? [{<RecTableID>}]	Get Number Of Recorded Points
DRR? (p. 181)	DRR? [<StartPoint> <NumberOfPoints> [{<RecTableID>}]]	Get Recorded Data Values
DRT (p. 183)	DRT {<RecTableID> <TriggerSource> <Value>}	Set Data Recorder Trigger Source
DRT? (p. 184)	DRT? [{<RecTableID>}]	Get Data Recorder Trigger Source
ERR? (p. 185)	ERR?	Get Error Number
FED (p. 185)	FED {<AxisID> <EdgeID> <Param>}	Find Edge
FNL (p. 187)	FNL [{<AxisID>}]	Fast Reference Move To Negative Limit
FPL (p. 189)	FPL [{<AxisID>}]	Fast Reference Move To Positive Limit
FRF (p. 190)	FRF [{<AxisID>}]	Fast Reference Move To Reference Switch
FRF? (p. 191)	FRF? [{<AxisID>}]	Get Referencing Result
GOH (p. 192)	GOH [{<AxisID>}]	Go To Home Position
HAR? (p. 192)	HAR? [{<AxisID>}]	Indicate Hard Stops
HDR? (p. 193)	HDR?	Get All Data Recorder Options
HDT (p. 195)	HDT {<HIDDeviceID> <HIDDeviceAxis> <HIDTableID>}	Set HID Default Lookup Table
HDT? (p. 196)	HDT? [{<HIDDeviceID> <HIDDeviceAxis>}]	Get HID Default Lookup Table
HIA (p. 197)	HIA {<AxisID> <MotionParam> <HIDDeviceID> <HIDDeviceAxis>}	Configure Control Done By HID Axis
HIA? (p. 199)	HIA? [{<AxisID> <MotionParam>}]	Get Configuration Of Control Done By HID Axis
HIB? (p. 199)	HIB? [{<HIDDeviceID> <HIDDeviceButton>}]	Get State Of HID Button
HIE? (p. 200)	HIE? [{<HIDDeviceID> <HIDDeviceAxis>}]	Get Deflection Of HID Axis
HIN (p. 202)	HIN {<AxisID> <HIDControlState>}	Set Activation State For HID Control

Command	Format	Description
HIN? (p. 203)	HIN? [{<AxisID>}]	Get Activation State Of HID Control
HIS? (p. 203)	HIS? [{<HIDeviceID> <HIDItemID> <HIDPropID>}]	Get Configuration Of HI Device
HIT (p. 206)	HIT {<HIDTableID> <HIDTableAddr> <HIDTableValue>}	Fill HID Lookup Table
HIT? (p. 207)	HIT? [<StartPoint> [<NumberOfPoints> [{<HIDTableID>}]]]	Get HID Lookup Table Values
HLP? (p. 210)	HLP?	Get List of Available Commands
HLT (p. 211)	HLT [{<AxisID>}]	Halt Motion Smoothly
HPA? (p. 211)	HPA?	Get List Of Available Parameters
JRC (p. 213)	JRC <Jump> <CMD?> <OP> <Value>	Jump Relatively Depending On Condition
LIM? (p. 214)	LIM? [{<AxisID>}]	Indicate Limit Switches
MAC (p. 216)	MAC <keyword> {<parameter>} Especially: MAC BEG <macroname> MAC DEF <macroname> MAC DEF? MAC DEL <macroname> MAC END MAC ERR? MAC NSTART <macroname> <uint> [<String1> [<String2>]] MAC START <macroname> [<String1> [<String2>]]	Call Macro Function
MAC? (p. 219)	MAC? [<macroname>]	List Macros
MAN? (p. 220)	MAN? <CMD>	Get Help String For Command
MEX (p. 221)	MEX <CMD?> <OP> <Value>	Stop Macro Execution Due To Condition
MOV (p. 223)	MOV {<AxisID> <Position>}	Set Target Position
MOV? (p. 224)	MOV? [{<AxisID>}]	Get Target Position

Command	Format	Description
MVR (p. 224)	MVR {<AxisID> <Distance>}	Set Target Relative To Current Position
ONT? (p. 226)	ONT? [{<AxisID>}]	Get On-Target State
POS (p. 227)	POS {<AxisID> <Position>}	Set Real Position
POS? (p. 227)	POS? [{<AxisID>}]	Get Real Position
RBT (p. 228)	RBT	Reboot System
RMC? (p. 228)	RMC?	List Running Macros
RON (p. 229)	RON {<AxisID> <ReferenceOn>}	Set Reference Mode
RON? (p. 229)	RON? [{<AxisID>}]	Get Reference Mode
RPA (p. 230)	RPA [{<ItemID> <PamID>}]	Reset Volatile Memory Parameters
RTR (p. 231)	RTR <RecordTableRate>	Set Record Table Rate
RTR? (p. 232)	RTR?	Get Record Table Rate
SAI (p. 232)	SAI {<AxisID> <NewIdentifier>}	Set Current Axis Identifiers
SAI? (p. 233)	SAI? [ALL]	Get List Of Current Axis Identifiers
SEP (p. 234)	SEP <Pswd> {<ItemID> <PamID> <PamValue>}	Set Nonvolatile Memory Parameters
SEP? (p. 235)	SEP? [{<ItemID> <PamID>}]	Get Nonvolatile Memory Parameters
SMO (p. 236)	SMO {<AxisID> <ControlValue>}	Set Open-Loop Control Value
SMO? (p. 238)	SMO? [{<AxisID>}]	Get Control Value
SPA (p. 238)	SPA {<ItemID> <PamID> <PamValue>}	Set Volatile Memory Parameters
SPA? (p. 241)	SPA? [{<ItemID> <PamID>}]	Get Volatile Memory Parameters
SRG? (p. 241)	SRG? {<AxisID> <RegisterID>}	Query Status Register Value
SST (p. 243)	SST {<AxisID> <StepSize>}	Set Step Size
SST? (p. 243)	SST? [{<AxisID>}]	Get Step Size
STE (p. 244)	STE <AxisID> <Amplitude>	Start Step And Response Measurement

Command	Format	Description
STP (p. 245)	STP	Stop All Axes
SVO (p. 245)	SVO {<AxisID> <ServoState>}	Set Servo Mode
SVO? (p. 246)	SVO? [{<AxisID>}]	Get Servo Mode
TAC? (p. 247)	TAC?	Tell Analog Channels
TAV? (p. 247)	TAV? [{<AnalogInputID>}]	Get Analog Input Voltage
TIO? (p. 248)	TIO?	Tell Digital I/O Lines
TMN? (p. 248)	TMN? [{<AxisID>}]	Get Minimum Commandable Position
TMX? (p. 249)	TMX? [{<AxisID>}]	Get Maximum Commandable Position
TNR? (p. 249)	TNR?	Get Number Of Record Tables
TRO (p. 250)	TRO {<TrigOutID> <TrigMode>}	Set Trigger Output State
TRO? (p. 250)	TRO? [{<TrigOutID>}]	Get Trigger Output State
TRS? (p. 251)	TRS? [{<AxisID>}]	Indicate Reference Switch
TVI? (p. 252)	TVI?	Tell Valid Character Set For Axis Identifiers
VAR (p. 252)	VAR <Variable> <String>	Set Variable Value
VAR? (p. 254)	VAR? [{<Variable>}]	Get Variable Value
VER? (p. 254)	VER?	Get Versions Of Firmware And Drivers
WAC (p. 255)	WAC <CMD?> <OP> <Value>	Wait For Condition
WPA (p. 256)	WPA <Pswd> [{<ItemID> <PamID>}]	Save Parameters To Nonvolatile Memory

## 9.6 Command Descriptions for GCS 2.0

### #4 (Request Status Register)

Description: Requests system status information.

Format: #4

Arguments: None

**Response:** The answer is bit-mapped. See below for the individual codes.

**Notes:** This command is identical in function to SRG? (p. 241), but only one character is sent via the interface. Therefore #4 can also be used while the controller is performing time-consuming tasks.

For the E-871, the response is the sum of the following codes, in hexadecimal format:

Bit	15	14	13	12	11	10	9	8
Descript- ion	On-target state	Determines the reference value	In motion	Servo mode on	-	-	-	Error flag
Bit	7	6	5	4	3	2	1	0
Descript- ion	Digital input line 4	Digital input line 3	Digital input line 2	Digital input line 1	-	Positive limit switch	Ref- erence point switch	Nega- tive limit switch

**Example:** Send: #4

Receive: 0x9005

Note: The response is given in hexadecimal format. It means that the axis is on target (on-target state = true), the servo mode is on, no error has occurred, the states of the digital input lines 1 to 4 are low, and the stage is on the positive side of the reference point switch (limit switches are not active; note that the logic of the signals is inverted in this example).

### #5 (Request Motion Status)

**Description:** Requests motion state of the axes.

**Format:** #5

**Arguments:** None

Response: The answer <uint> is bit-mapped and returned as the hexadecimal sum of the following codes:

1 = First axis is moving  
 2 = Second axis is moving  
 4 = Third axis is moving  
 ...

Examples: 0 indicates motion of all axes complete  
 3 indicates that the first and the second axis are moving

### #7 (Request Controller Ready Status)

Description: Asks controller for ready status (tests if controller is ready to perform a new command).

Note: Use #5 (p. 160) instead of #7 to verify if motion has finished.

Format: #7

Arguments: None

Response: B1h (ASCII character 177 = "±" in Windows) if controller is ready

B0h (ASCII character 176 = "°" in Windows) if controller is not ready  
 (e. g. performing a reference move)

Troubleshooting: The response characters may appear differently in non-Western character sets or other operating systems.

### #8 (Query if Macro Is Running)

Description: Tests if a macro is running on the controller.

Format: #8

Arguments: None

Response:           <uint>=0 no macro is running  
                   <uint>=1 a macro is currently running

#### #24 (Stop All Axes)

Description:       Stops all axes abruptly. For details see the notes below.

Sets error code to 10.

This command is identical in function to STP (p. 245), but only one character is sent via the interface.

Format:            #24

Arguments:       None

Response:       None

Notes:            #24 stops all motion caused by motion commands (e.g. MOV (p. 223), MVR (p. 224), GOH (p. 192), STE (p. 244), SMO (p. 236)), commands for reference point definition (FNL (p. 187), FPL (p. 189), FRF (p. 190)) and macros (MAC (p. 216)). Also stops macro execution.

After the axes are stopped, their target positions are set to their current positions.

#### \*IDN? (Get Device Identification)

Description:       Reports the device identity number.

Format:            \*IDN?

Arguments:       None

Response:       Single-line text terminated with a termination character (line feed) with controller name, serial number, and firmware version

Notes:            For the E-871, \*IDN? responds something like:

```
©2012 Physik Instrumente (PI) GmbH & Co. KG,
E-871.1A1, 112062031, 1.0.0.0
```



**ADD (Add and Save to Variable)**

Description: Adds two values and saves the result to a variable (p. 153).

The variable is present in volatile memory (RAM) only.

Format: ADD <Variable> <FLOAT1> <FLOAT2>

Arguments: <Variable> is the name of the variable to which the result is to be saved.

<FLOAT1> is the first summand.

<FLOAT2> is the second summand.

For the summands, floating point numbers are expected. They can be given directly or via the value of a variable.

Response: None

Notes: Local variables can be set using ADD in macros only.

Example 1: Value \$B is added to value \$A, and the result is saved to variable C:

```
ADD C $A $B
```

Example 2: The name of the variable to which the result is to be copied is given via the value of another variable:

Send: VAR?

Receive:

A=468

B=123

3Z=WORKS

Send: ADD A\${3Z} \$A \$B

Send: VAR?

Receive:

A=468

B=123

AWORKS=591

3Z=WORKS

Send: ADD \${3Z} \$A \$B

Send: VAR?

Receive:

A=468

B=123

AWORKS=591

WORKS=591

3Z=WORKS

Example 3: The macros below can be used to create a "flashing light" with LEDs that are connected to the digital output lines of the controller. \$1 and \$2 are values of local variables and must be given as arguments of the MAC START or MAC NSTART command when starting the macros (see below).

DIO 0 <bitmask>: Sets the output channels according to <bitmask>. For example, "DIO 0 5" activates channels 1 and 3 and deactivates all other channels (5 is 0000 0101 in binary notation).

To implement the "flashing light", perform the following steps:

1. Write the "STEPS" macro:

```
MAC BEG STEPS
DIO 0 $1
ADD 1 $1 1
DEL $2
JRC -3 VAR? 1 <= 15
ADD 1 $1 -1
DIO 0 $1
DEL $2
JRC -3 VAR? 1 > 0
MAC END
```

2. Write the "TEST" macro:

```
MAC BEG TEST
MAC START STEPS 0 $1
ADD 1 $1 10
JRC -2 VAR? 1 < 110
VAR 1 10
ADD 2 $2 -1
JRC -5 VAR? 2 > 0
MAC END
```

3. Start the TEST macro with arguments that define the variable values \$1 and \$2:

```
MAC START Test 10 50
```

Meaning of the variables here:

\$1: Delay in ms between each step in the STEPS macro. The value is incremented by 10 by the TEST macro until it reaches 110.

\$2: Number of repetitions of the whole "flashing light" procedure.

#### CCL (Set Command Level)

Description: Changes the active "command level" and thus determines the availability of commands and of write access to system parameters.

Format: CCL <Level> [<PSWD>]

Arguments: <Level> is a command level of the controller

<PSWD> is the password required for changing to the appropriate command level

The following command levels and passwords are valid:

Level = 0 is the default setting, all commands provided for "normal" users are available, read access to all parameters, no password required.

Level = 1 adds additional commands and write access to level-1 parameters (commands and parameters from level 0 are included). The required password is "advanced".

Level > 1 is provided for PI service personnel only. Users cannot change to a level > 1. Contact the customer service department if there seem to be problems with level 2 or higher parameters.

Response: None

Troubleshooting: Invalid password

Notes: With the E-871, the command levels only determine the write permission for the parameters. The availability of the commands of the E-871 is independent of the enabled command level.

HPA? (p. 211) lists the parameters including the information about which command level allows write access to them. For more information about using parameters, see "Adapting Settings" (p. 271).

After controller switching-on or reboot, the active command level is always level 0.

#### **CCL? (Get Command Level)**

Description: Get the active "command level".

Format: CCL?

Arguments: none

Response: <Level> is the currently active command level; uint.

Notes: <Level> should be 0 or 1.

<Level> = 0 is the default setting, write access is given for level 0 parameters, read access is given for all parameters

<Level> = 1 allows write access for level 1 parameters (parameters from level 0 are included).

#### **CPY (Copy Into Variable)**

Description: Copies a command response to a variable (p. 153).

The variable is present in volatile memory (RAM) only.

Format:	CPY <Variable> <CMD?>
Arguments:	<p>&lt;Variable&gt; is the name of the variable to which the command response is to be copied.</p> <p>&lt;CMD?&gt; is one query command in its usual notation. The response has to be a single value and not more.</p>
Response:	None
Notes:	Local variables can be set using CPY in macros only.
Example 1:	<p>Using the following macro, it is possible to connect through the digital input and output lines of the controller. 1 is a local variable whose value must be given as argument of the MAC START or MAC NSTART command when starting the macro.</p> <p>Write the "connect" macro:</p> <pre>MAC BEG connect CPY 1 DIO? 0 DIO 0 \$1 MAC START CONNECT MAC END</pre>
Example 2:	<p>It is possible to copy the value of one variable (e. g. SOURCE) to another variable (e. g. TARGET):</p> <pre>CPY TARGET VAR? SOURCE</pre>

**CST? (Get Assignment Of Stages To Axes)**

Description:	Gets the name of the stage type that is connected to the given axis.
Format:	CST? [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller

Response: {<AxisID>="<string> LF}

where

<string> is the name of the stage type that is assigned to the axis.

Notes: The stage name is read from the **Stage Name** parameter (ID 0x3C). When the parameter has the value NOSTAGE, the axis is "deactivated". A deactivated axis is not accessible for axis-related commands (e.g. motion commands or position queries). The identifier of a deactivated axis can only be queried with `SAI? ALL`.

You can set the value of the 0x3C parameter specifically to the name of your stage using SPA (p. 238) or SEP (p. 234). Since the PC software from PI uses the parameter value to configure the E-871 for the connected stage (p. 84), however; manually changing with SPA or SEP is not recommended.

### CSV? (Get Current Syntax Version)

Description: Gets the GCS syntax version used in the firmware.

Format: CSV?

Arguments: None

Response: The current GCS syntax version

Notes: 1.0 (for GCS 1.0) or 2.0 (for GCS 2.0) are possible responses.

### CTO (Set Configuration Of Trigger Output)

Description: Configures the trigger output conditions for the given digital output line.

Format: CTO {<TrigOutID> <CTOPam> <Value>}

Arguments: <TrigOutID> is one digital output line of the controller, see below for details.

<CTOPam> is the CTO parameter ID in decimal format, see below for the available IDs.

<Value> is the value to which the CTO parameter is set, see below.

Response: None

Notes: The trigger output conditions will become active when enabled with TRO (p. 250). Do not use DIO (p. 176) on digital output lines for which the trigger output is enabled with TRO.

The CTO settings are lost when you power down or reboot the E-871. An easy way to keep them is to save them to a macro.

Output lines and trigger conditions available: <TrigOutID> corresponds to the digital output lines 1 to 4, IDs = 1 to 4; see "I/O" (p. 310).

<CTOPam> parameter IDs, available for E-871:

1 = TriggerStep  
2 = Axis  
3 = TriggerMode  
7 = Polarity  
8 = StartThreshold  
9 = StopThreshold  
10 = TriggerPosition

<Value> available for the appropriate <CTOPam> ID:

for TriggerStep: distance

for Axis: the identifier of the axis to be connected to the digital output line. Irrelevant for the MotionError trigger mode.



for TriggerMode (default value is 0):

- 0 = PositionDistance;  
a trigger pulse is written whenever the axis has covered the TriggerStep distance (<CTOPam> ID 1). Optionally, values for StartThreshold and StopThreshold (<CTOPam> IDs 8 and 9) can be defined to enable the trigger output for a limited position range and a certain direction of motion only (negative or positive; Note: In case the motion direction is reversed before the axis position has reached the stop threshold, trigger pulses will continue to be generated). When StartThreshold and StopThreshold are set to the same value, they will not be used.
- 2 = OnTarget;  
the on-target state of the selected axis is transferred to the selected digital output line (this state can also be read with the ONT? command).
- 5 = MotionError;  
the selected digital output line becomes active when a motion error occurs. The line will stay active until the error code is reset to 0 (by a query).
- 6 = InMotion;  
the selected digital output line is active as long as the selected axis is in motion (the in-motion state can also be read with #4, #5 or the SRG? command).
- 7 = Position+Offset;  
the first trigger pulse is written when the axis has reached the position given by TriggerPosition (<CTOPam> ID 10). The next trigger pulses are written each time that the axis position equals the sum of the last valid trigger position and the distance given by TriggerStep (<CTOPam> ID 1). Trigger output ends when the axis position exceeds the value given by StopThreshold (<CTOPam> ID 9). The sign of the TriggerStep value determines for which direction of motion trigger pulses are to be output. Trigger processing is done by the DSP of the E-871.
- 8 = SinglePosition;  
the selected digital output line is active when the axis position has reached or exceeded the position given by TriggerPosition (<CTOPam> ID 10).

for Polarity (default value is 1): sets the signal polarity for the digital output line

0 = Active Low

1 = Active High

for StartThreshold/StopThreshold: position value;  
if used for the PositionDistance trigger mode, both thresholds must be set in order to determine the position range and the direction of motion for trigger output;  
StopThreshold is used as the stop condition for Position+Offset trigger mode

for TriggerPosition: position value;  
if used in the Position+Offset trigger mode, the first trigger pulse is output at this position;  
if used in the SinglePosition trigger mode, the output line becomes active when this position is reached or exceeded

Application examples and further details see "Digital Output Signals" (p. 102) and the lines below.

Example 1:

A pulse is to be generated on digital output line 1 (ID 1) whenever axis 1 has covered a distance of 0.05  $\mu\text{m}$ . The following parameters must be set:

TrigOutID = 1

Axis = 1

TriggerMode = 0

TriggerStep = 0.05

Send: CTO 1 2 1

Send: CTO 1 3 0

Send: CTO 1 1 0.00005

Example 2: In this example, the digital output line 1 shall be set from low to high when axis A starts to move. The following parameters must be set:

TrigOutID = 1

Axis = A (axis identifier was changed with `SAI`)

TriggerMode = 6

Polarity = Active High

So you have to send:

```
CTO 1 2 A
```

```
CTO 1 3 6
```

```
CTO 1 7 1
```

### CTO? (Get Configuration Of Trigger Output)

Description: Gets the values set for specified trigger output lines and parameters.

Format: CTO? [{<TrigOutID> <CTOPam>}]

Arguments: <TrigOutID>: is a digital output line of the controller; see CTO.

<CTOPam>: parameter ID; see CTO.

If all arguments are omitted, the response contains the values for all parameters and all output lines.

Response: {<TrigOutID> <CTOPam>="<Value> LF}

For <Value> see CTO.

### DEL (Delay the Command Interpreter)

Description: Delays <uint> milliseconds.

Format: DEL <uint>

Arguments: <uint> is the delay value in milliseconds.

Response: None

Notes: DEL can only be used in macros. Do not mistake MAC DEL (deletes macros) for DEL (delays).  
See the MAC command (p. 216) and "Controller Macros" (p. 131) for more information.

### DFH (Define Home Position)

Description: Redefines the zero position of the given axis by setting the position value to zero at the current position.

If all arguments are omitted, DFH defines the zero position of all axes.

Format: DFH [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller.

Response: none

Troubleshooting: Illegal axis identifier

Notes: DFH sets the current position of the axis to zero and saves the position value which was valid when the command was called as offset in the volatile memory. By adding this offset to the response, the output values of the following commands are adapted to the new zero position:

- POS? (p. 227) (query of the current position)
- TMN? (p. 248) (query of the minimum commandable position)
- TMX? (p. 249) (query of the maximum commandable position)

DFH does **not** change the values of the parameters for the definition of travel range and soft limits (p. 32).

The offset is reset to zero in the following cases:

- When switching on and rebooting the E-871:  
For all axes

- During reference point definition:  
For the axis in question

Example:

Send: MOV 1 9.87

Send: POS? 1

Receive: 1=9.8700005

Send: DFH? 1

Receive: 1=0.0000000

Send: TMN? 1

Receive: 1=0.0000000

Send: TMX? 1

Receive: 1=14.9999982

Note: Axis 1 is moved to absolute position 9.87 mm. Then the current axis position (with POS?), the current offset value (with DFH?) and the minimum and maximum commandable position (with TMN? and TMX?) are queried.

Send: DFH 1

Send: POS? 1

Receive: 1=0.0000000

Send: DFH? 1

Receive: 1=9.8700005

Send: TMN? 1

Receive: 1=-9.8700005

Send: TMX? 1

Receive: 1=5.1299978

Note: The axis has not moved. The current axis position was defined as the new zero position using DFH. As a result, the offset value for axis 1 is now 9.87 mm. The values for the minimum and maximum commandable position were adapted to the new zero position by adding the offset.

**DFH? (Get Home Position Definition)**

**Description:** Gets the sensor position at which the given axis has its zero position.

If all arguments are omitted, the position value of all axes is queried.

**Format:** DFH? [{<AxisID>}]

**Arguments:** <AxisID> is one axis of the controller.

**Response:** {<AxisID>="<SensorPosition> LF}

where

<SensorPosition> is the sensor position which was valid at the time the last DFH command was processed. This sensor position value is used internally as offset for the calculation of the current axis position.

**Troubleshooting:** Illegal axis identifier

**Notes:** The sensor position which was valid when the last DFH command was processed is available in the volatile memory as an offset. The offset is reset to zero in the following cases:

- When switching on and rebooting the E-871: for all axes
- During reference point definition: for the axis in question

See DFH for an example.

**DIO (Set Digital Output Line)**

**Description:** Switches the specified digital output line(s) to specified state(s).

Use TIO? (p. 248) to get the number of installed digital I/O lines.

**Format:** DIO {<DIOID> <OutputOn>}

Arguments:	<p>&lt;DIOID&gt; is one digital output line of the controller, see below for details.</p> <p>&lt;OutputOn&gt; is the state of the digital output line, see below for details.</p>
Response:	none
Notes:	<p>Using the DIO command, you can activate/deactivate the Output 1 to Output 4 lines which are located on the I/O socket (p. 310). With the E-871, you can either set a single line per DIO command, or all lines at once.</p> <p>The &lt;DIOID&gt; identifiers to use for the lines are 1 to 4. With the identifier 0, all lines are set according to a bit pattern given by &lt;OutputOn&gt;.</p> <p>If &lt;OutputOn&gt;=1 the line is set to HIGH/ON, if &lt;OutputOn&gt;=0 it is set to LOW/OFF.</p> <p>Do not use DIO on output lines for which the trigger output is enabled with TRO (p. 250).</p>

### DIO? (Get Digital Input Lines)

Description:	<p>Gets the states of the specified digital input lines.</p> <p>Use TIO? (p. 248) to get the number of available digital I/O lines.</p>
Format:	DIO? [{<DIOID>}]
Arguments:	<DIOID> is the identifier of the digital input line, see below for details.
Response:	{<DIOID>="<InputOn> LF}
	<p>where</p> <p>&lt;InputOn&gt; gives the state of the digital input line, see below for details.</p>

Notes: You can use the DIO? command to directly read the digital input lines 1 to 4 that are located on the **I/O** socket (p. 310).

The <DIOID> identifiers to use for the lines are 1 to 4. If the identifier is omitted or 0, all lines are queried.

If <InputOn>=0, the digital input is LOW/OFF; if <InputOn>=1, the digital input is HIGH/ON. If <DIOID> is 0, <InputOn> is a bit pattern which gives the states of all lines in hexadecimal format.

#### **DRC (Set Data Recorder Configuration)**

Description: Determines the data source to be used and the kind of data to be recorded (record option) for the data recorder table given.

Format: DRC {<RecTableID> <Source> <RecOption>}

Arguments: <RecTableID>: is one data recorder table of the controller, see below.

<Source>: is the data source, for example, an axis or a channel of the controller. The required source depends on the selected record option.

<RecOption>: is the kind of data to be recorded (record option).

See below for a list of the available record options and the corresponding data sources.

Response: None



Notes: The E-871 has two data recorder tables with 1024 points per table.

With HDR? (p. 193) you will obtain a list of all available record and trigger options and additional information about the data recording. The number of available data recorder tables can be read with TNR? (p. 249).

For more information see "Data Recorder" (p. 100).

Recording options available with the corresponding data sources:	0=Nothing is recorded 1=Commanded position of axis 2=Actual position of axis 3=Position error of axis 73=Motor output of axis 74=Kp of axis 75=Ki of axis 76=Kd of axis 80=Signal status register of axis 81=Analog input (channel = 1 - 9)
--	--

Note: The input channels for the record option 81 can be the Input 1 to Input 4 lines of the **I/O** socket (p. 310). Use the identifiers 1 to 4 for these data sources.

The data source identifiers 5 to 8 refer to the inputs for the axes and buttons of the HID device.

5 = Axis 1 of the HID device  
 6 = Button 1 of the HID device  
 7 = Axis 2 of the HID device  
 8 = Button 2 of the HID device

Further source identifiers are reserved for additional analog input channels.

### **DRC? (Get Data Recorder Configuration)**

Description: Gets the settings made with DRC (p. 178).

Format: DRC? [{<RecTableID>}]

Arguments: <RecTableID>: is a data recorder table of the controller; if this information is omitted, the response will contain the settings for all tables.

Response: The current DRC settings:

```
{<RecTableID>=" "<Source> <RecOption> LF}
```

where

<Source>: is the data source, for example an axis or a channel of the controller. The source type depends on the record option.

<RecOption>: is the kind of data to be recorded (record option).

See DRC for a list of the available record options and the corresponding data sources.

#### **DRL? (Get Number of Recorded Points)**

Description: Reads the number of points comprised by the last recording.

Format: DRL? [{<RecTableID>}]

Arguments: <RecTableID> is one data recorder table of the controller

Response: {<RecTableID>=" "<uint> LF}

where

<uint> gives the number of points recorded with the last recording

Notes: The number of points is reset to zero for a data recorder table when changing its configuration with DRC (p. 178).

**DRR? (Get Recorded Data Values)**

Description: Gets the last recorded data.

Reading can take some time depending on the number of points to be read!

It is possible to read the data while recording is still in progress.

Format: DRR? [<StartPoint> <NumberOfPoints> [{<RecTableID>}]]

Arguments: <StartPoint> is the first point to be read from the data recorder table; it starts with index 1.

<NumberOfPoints> is the number of points to be read per table.

<RecTableID> is one data recorder table of the controller.

Response: For the recorded data in GCS array format, see the separate manual for the GCS array, SM146E, and the example below.

Notes: If <RecTableID> is omitted, the data from all tables with a nonzero record option is read.

With HDR? (p. 193) you will obtain a list of available record and trigger options and additional information about data recording.

Further information, see the description of the DRC (p. 178) command as well as "Data Recorder" (p. 100).

Example:

```
rtr?  
10  
drr? 1 20  
# REM E-871  
#  
# VERSION = 1  
# TYPE = 1  
# SEPARATOR = 32  
# DIM = 2  
# SAMPLE_TIME = 0.000500  
# NDATA = 20  
#  
# NAME0 = Actual Position of Axis  AXIS:1  
# NAME1 = Motor Output of Axis  AXIS:1  
#  
# END_HEADER  
0.2000000 2247  
0.1998270 7313  
0.1997500 2705  
0.1996760 982  
0.1996840 358  
0.1996810 129  
0.1996760 46  
0.1996720 16  
0.1996660 5  
0.1996570 0  
0.1996650 0  
0.1996590 0  
0.1996590 0  
0.1996590 0  
0.1996590 0  
0.1996630 0  
0.1996590 0  
0.1996630 0  
0.1996620 0  
0.1996660 0  
0.1996610 0
```

**DRT (Set Data Recorder Trigger Source)**

Description:	Defines a trigger source for the given data recorder table.
Format:	DRT <RecTableID> <TriggerSource> <Value>
Arguments:	<p>&lt;RecTableID&gt; is one data recorder table of the controller. See below for details.</p> <p>&lt;TriggerSource&gt; ID of the trigger source, see below for a list of available options.</p> <p>&lt;Value&gt; depends on the trigger source, can be a dummy, see below.</p>
Response:	none
Notes:	<p>At present, only 0 is valid for &lt;RecTableID&gt;; this means that the given trigger source is set for all data recorder tables.</p> <p>Regardless of the trigger option set, the data recording is always triggered when a step response measurement is started with STE (p. 244).</p> <p>A trigger option set with DRT will become valid for all data recorder tables with non-zero record option.</p> <p>With HDR? (p. 193) you will obtain a list of available record options and trigger options and additional information about data recording.</p> <p>For further information see the description of the DRC command (p. 178) and "Data Recorder" (p. 100).</p>
Available trigger options:	<p>0 = default setting; data recording is triggered with STE; &lt;Value&gt; must be a dummy</p> <p>1 = any command changing target position (e.g. MVR (p. 224), MOV (p. 223)); &lt;Value&gt; must be a dummy</p> <p>2 = next command, resets trigger after execution; &lt;Value&gt; must be a dummy</p>

6 = any command changing target position (e.g. MVR, MOV), resets trigger after execution; <Value> must be a dummy

7 = SMO command, resets trigger after execution; <Value> must be a dummy

#### **DRT? (Get Data Recorder Trigger Source)**

Description: Gets the trigger source for the data recorder tables.

Format: DRT? [{<RecTableID>}]

Arguments: <RecTableID> is one data recorder table of the controller.

Response: {<RecTableID>="<TriggerSource> <Value> LF}

where

<TriggerSource> is the identifier of the trigger source.

<Value> depends on the trigger source; 0 is a dummy.

Further information is found in the description of the DRT command (p. 183).

Notes: Since all data recorder tables of the E-871 have the same trigger source, the DRT? response is given as a single line of the form

0=<TriggerSource> <Value>

**ERR? (Get Error Number)**

**Description:** Gets error code <int> of the last occurred error and resets the error to 0.

Only the last error is buffered. You should therefore call ERR? after each command.

The error codes and their descriptions are listed in "Error Codes" (p. 258).

**Format:** ERR?

**Arguments:** None

**Response:** The error code of the last error that occurred (integer).

**Troubleshooting:** Communication breakdown

**FED (Find Edge)**

**Description:** Moves given axis to a given signal edge.

FED does not set a certain position value at the selected edge (in contrast to the FNL (p. 187), FPL (p. 189) and FRF (p. 190) commands for reference point definition), i.e. the axis is not "referenced" after using FED.

If multiple axes are given in the command, they are moved synchronously.

**Format:** FED {<AxisID> <EdgeID> <Param>}

**Arguments:** <AxisID> is one axis of the controller.

<EdgeID> is the type of edge the axis has to move to. See below for available edge types.

<Param> depends on the selected edge and qualifies it. See below for details.

**Response:** none

- Troubleshooting: Illegal axis identifier; limit switches and/or reference point switch are disabled (see below); servo mode is off
- Notes: Servo mode must be switched on with SVO (p. 245) for the commanded axis prior to using this command (closed-loop operation).

The firmware of the E-871 determines the following based on parameters:

- Is a reference point switch present (parameter 0x14)?
- Are limit switches present (parameter 0x32)?
- Should the hard stops be used for reference moves (parameter 0x7A)?
- If the reference point switch is represented by an index pulse: How should the move to the index pulse take place (parameters 0x70, 0x78, 0x79)?

According to the values of those parameters, the E-871 enables or disables FED motions to the appropriate signal edges. Adapt the parameter values to your hardware using SPA (p. 238) or SEP (p. 234). See "Adapting Settings" (p. 271) for more information.

You can use the digital input lines instead of the switches as source of the switch signals for FED. Further information see "Digital Input Signals" (p. 111).

FED can be used to measure the physical travel range of a new mechanical system and thus determine the values for the corresponding parameters:

- Distance from the negative to the positive limit switch
- Distance between the negative limit switch and the reference point switch (parameter ID 0x17)
- Distance between the reference point switch and the positive limit switch (parameter ID 0x2F).

Further information, see "Travel Range and Soft Limits" (p. 32).

The motion can be stopped by #24 (p. 162), STP (p. 245) and HLT (p. 211).



Motion commands such as FED are not allowed when the HID control is enabled for the axis. Further information see "Control with an HID Device" (p. 118).

Available edge types and parameters:

The following edge types with their parameter settings are available:

- 1 = negative limit switch, <Param> must be 0
- 2 = positive limit switch, <Param> must be 0
- 3 = reference point switch, <Param> must be 0

### **FNL (Fast Reference Move To Negative Limit)**

Description: Starts a reference move.

Moves the given axis to the negative physical limit of its travel range and sets the current position to a defined value. See below for details.

If multiple axes are given in the command, they are moved synchronously.

Format: FNL [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller, if omitted, all axes are involved.

Response: none

Troubleshooting: Illegal axis identifier

Notes: Servo mode must be switched on with SVO (p. 245) for the commanded axis prior to using this command (closed-loop operation).

If reference move was successful, absolute motion will afterwards be possible in closed-loop operation.

The negative physical limit of the travel range can be represented by the following items of the stage:

- Negative limit switch
- If the stage does not have integrated limit switches:  
negative hard stop

The difference in the values of the parameters 0x16 and 0x17 is set as the current position when the axis is at the negative physical limit of the travel range (value can be negative).

You can use a digital input instead of the negative limit switch as source of the negative limit switch signal for FNL. Further information see "Digital Input Signals" (p. 111).

The motion can be stopped by #24 (p. 162), STP (p. 245) and HLT (p. 211).

Use FRF? (p. 191) to check whether the reference move was successful.

For best repeatability, always perform the reference point definition in the same way.

If soft limits (parameters 0x15 and 0x30) are used to reduce the travel range, the limit switches or the hard stops cannot be used for reference moves.

Further information, see "Reference Point Definition" (p. 37) and "Travel Range and Soft Limits" (p. 32).

**FPL (Fast Reference Move To Positive Limit)**

Description: Starts a reference move.

Moves the given axis to the positive physical limit of its travel range and sets the current position to a defined value. See below for details.

If multiple axes are given in the command, they are moved synchronously.

Format: FPL [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller, if omitted, all axes are involved.

Response: none

Troubleshooting: Illegal axis identifier

Notes: Servo mode must be switched on with SVO (p. 245) for the commanded axis prior to using this command (closed-loop operation).

If reference move was successful, absolute motion will afterwards be possible in closed-loop operation.

The positive physical limit of the travel range can be represented by the following items of the stage:

- Positive limit switch
- If the stage does not have integrated limit switches:  
positive hard stop

The sum of the values of the parameters 0x16 and 0x2F is set as the current position when the axis is at the positive physical limit of the travel range.

You can use a digital input instead of the positive limit switch as source of the positive limit switch signal for FPL. Further information see "Digital Input Signals" (p. 111).

The motion can be stopped by #24 (p. 162), STP (p. 245) and HLT (p. 211).

Use FRF? (p. 191) to check whether the reference move was successful.

For best repeatability, always perform the reference point definition in the same way.

If soft limits (parameters 0x15 and 0x30) are used to reduce the travel range, the limit switches or the hard stops cannot be used for reference moves.

Further information, see "Reference Point Definition" (p. 37) and "Travel Range and Soft Limits" (p. 32).

### **FRF (Fast Reference Move To Reference Switch)**

Description:	Starts a reference move.
	Moves the given axis to the reference point switch and sets the current position to a defined value. See below for details.
	If multiple axes are given in the command, they are moved synchronously.
Format:	FRF [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller, if omitted, all axes are involved.
Response:	none
Troubleshooting:	Illegal axis identifier
Notes:	<p>Servo mode must be switched on with SVO (p. 245) for the commanded axis prior to using this command (closed-loop operation).</p> <p>If reference move was successful, absolute motion will afterwards be possible in closed-loop operation.</p> <p>The value of the parameter 0x16 is set as the current position when the axis is at the reference point switch.</p>

You can use a digital input instead of the reference point switch as source of the reference signal for the FRF command. Further information see "Digital Input Signals" (p. 111).

The motion can be stopped by #24 (p. 162), STP (p. 245) and HLT (p. 211).

Use FRF? (p. 191) to check whether the reference move was successful.

Use FNL (p. 187) or FPL (p. 189) instead of FRF to perform a reference move for an axis which has no reference point switch.

For best repeatability, always perform the reference point definition in the same way.

Further information, see "Reference Point Definition" (p. 37) and "Travel Range and Soft Limits" (p. 32).

### FRF? (Get Referencing Result)

Description: Gets whether the given axis is referenced or not.

Format: FRF? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller.

Response: {<AxisID>="<uint> LF}

where

<uint> indicates whether the axis has been successfully referenced (=1) or not (=0).

Troubleshooting: Illegal axis identifier

Notes: An axis is considered as "referenced" when the current position value is set to a known position. This is the case

when a reference move was successfully performed with FNL (p. 187), FPL (p. 189) or FRF (p. 190) or when the position was set directly with POS (p. 227) (depending on the mode of reference point definition set with RON (p. 229)).

### GOH (Go To Home Position)

Description:	Moves the given axes to the zero position.
	GOH [{<AxisID>}] is identical to MOV {<AxisID> 0}
	Servo mode must be switched on for the commanded axis prior to using this command (closed-loop operation).
	The motion can be stopped by #24 (p. 162), STP (p. 245) and HLT (p. 211).
Format:	GOH [{<AxisID>}]
Arguments:	<AxisID>: is an axis of the controller, if this is omitted, all axes are affected.
Response:	None
Troubleshooting:	Illegal axis identifier

### HAR? (Indicate Hard Stops)

Description:	Gets whether the hard stops of the axis can be used for reference moves.
Format:	HAR? [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller.

Response: {<AxisID>="<uint> LF}

where

<uint> indicates whether the hard stops of the axis can be used for reference moves (=1) or not (=0).

Troubleshooting: Illegal axis identifier

Notes: The firmware of the E-871 uses a parameter (ID 0x7A) to determine whether the hard stops of the axis can be used for reference moves. According to the value of this parameter, the E-871 enables or disables reference moves which use the hard stops.

Adapt the parameter value to your hardware using SPA (p. 238) or SEP (p. 234). Further information, see "Reference Point Definition" (p. 37).

### HDR? (Get All Data Recorder Options)

Description: Lists a help string which contains all information available about data recording (record options and trigger options, information about additional parameters and commands concerned with data recording).

Format: HDR?

Arguments: None

Response #RecordOptions  
{<RecOption>="<DescriptionString>[ of <Channel>]}

#TriggerOptions  
[<TriggerOption>="<DescriptionString>]

#Parameters to be set with SPA  
[<ParameterID>="<DescriptionString>]

#Additional information  
[<Command description>("<Command>")]

#Sources for Record Options

[{<RecOption>="<Source>}]

end of help

Example:

For the E-871, the HDR? response is as follows:

hdr?

#RecordOptions

0=Nothing is recorded

1=Commanded Position of Axis

2=Actual Position of Axis

3=Position Error of Axis

73=Motor Output of Axis

74=Kp of Axis

75=Ki of Axis

76=Kd of Axis

80=Signal Status Register of Axis

81=Analog input (Channel = 1 - 9)

#TriggerOptions

0=default setting

1=any command changing position (e.g. MOV)

2=next command

6=any command changing position (e.g. MOV),  
reset trigger after execution

7=with SMO command, reset trigger after  
execution

#Additional information

2 record tables

1024 datapoints per table

end of help

Note: TriggerOptions = 0 (default) means that recording is triggered by the STE command (p. 244).



**HDT (Set HID Default Lookup Table)**

**Description:** Assigns a lookup table to the given axis of the given HID device.

Lookup tables are used during the HID control of several motion parameters of the axes of the E-871; for details, see HIA (p. 197). A lookup table maps the displacement of the axis of an HID device to the controlled motion parameter (for further details, see HIE? (p. 200)).

**Format:** HDT {<HIDDeviceID> <HIDDeviceAxis> <HIDTableID>}

**Arguments:** <HIDDeviceID> is one HID device connected to the controller; see below for details.

<HIDDeviceAxis> is one axis of the HID device; for more details, see below.

<HIDTableID> is one lookup table of the controller; for more information, see below.

**Response:** None

**Notes:** Lookup tables are only assigned with HDT in the volatile memory (RAM) of the E-871. With the WPA command (p. 256), the currently valid assignment can be saved in the nonvolatile memory of the E-871.

The E-871 supports one HID device (identifier: 1) and four axes of this HID device (identifiers: 1 to 4). Information on the supported axes of the HID device can be queried with the HIS? command (p. 203). Further information see "Commandable Items" (p. 15) and "Connecting an HID Device" (p. 66).

Available lookup tables: The E-871 provides the following lookup tables with 256 points each:

Identifier	Type
1	linear
2	parabolic (default)
101	user-defined
102	user-defined

With the HIT command (p. 206), user-defined lookup tables can be filled with values.

#### HDT? (Get HID Default Lookup Table)

Description: Gets the currently assigned lookup table for the given axis of the given HID device.

Format: HDT? [{<HIDDeviceID> <HIDDeviceAxis>}]

Arguments: <HIDDeviceID> is one HID device connected to the controller; for more details, see HDT.

<HIDDeviceAxis> is one axis of the HID device; for more details, see HDT.

Response: {<HIDDeviceID> <HIDDeviceAxis>="<HIDTableID>LF}

where

<HIDTableID> is one lookup table of the controller; for more information, see HDT.

**HIA (Configure Control Done By HID Axis)**

**Description:** Configures the control of axes of the E-871 by axes of HID devices ("HID control").

Assigns an axis of an HID device to the given motion parameter of the given axis of the E-871.

The HID control is enabled or disabled with the HIN command (p. 202). HIA can only be used when the HID control is disabled for the affected axis of the E-871.

**Format:** HIA {<AxisID> <MotionParam> <HIDDeviceID> <HIDDeviceAxis>}

**Arguments:** <AxisID> is one axis of the controller.

<MotionParam> is one motion parameter of the controller axis; for more details, see below.

<HIDDeviceID> is one HID device connected to the controller; see below for details.

<HIDDeviceAxis> is one axis of the HID device; for more details, see below.

**Response:** None

**Notes:** The HID control is only configured with HIA in the volatile memory (RAM) of the E-871. With the WPA command (p. 256), the currently valid configuration can be saved in the nonvolatile memory of the E-871.

The E-871 supports one HID device (identifier: 1) and four axes of this HID device (identifiers: 1 to 4). Information on the supported axes of the HID device can be queried with the HIS? command (p. 203). Further information see "Commandable Items" (p. 15) and "Connecting an HID Device" (p. 66).

<MotionParam> indicates the motion parameter to be controlled and can take on the following values:

Value	Motion parameter
0	<p>Deletes the current configuration of the HID control.</p> <p>Without specification of &lt;HIDDeviceID&gt; and &lt;HIDDeviceAxis&gt;, it can be sent in abbreviated format as HIA &lt;AxisID&gt; 0</p>
1	<p>Absolute target position</p> <p>The lookup table value that corresponds to the current displacement of the HID device axis is mapped to the travel range of the E-871 axis to be controlled. The limits of the travel range are specified by the values of the parameters 0x30 and 0x15 and can be queried with TMN? and TMX?.</p>
2	<p>Relative target position</p> <p>Intended for use with AB rotary encoders or pulse generators (p. 66). Each received pulse (if present: each mechanical detent) triggers a relative motion of the distance that is set with the SST command (p. 243).</p> <p><b>No</b> lookup tables are used to control the relative target position.</p>

If the axis of the E-871 is configured for the HID control of the absolute or relative target position: The current configuration must be deleted by sending HIA with the value zero for <MotionParam> before a new configuration can be set.

If the HID control is enabled with the HIN command, it will remain without effect in the following cases:

- <MotionParam> has the value zero, i. e. no function to be controlled is selected for the axis of the E-871.
- <HIDDeviceID> has the value zero, i. e. no HID device is selected for the HID control.
- <HIDDeviceAxis> has the value zero, i. e. no axis of the HID device is selected for the HID control.

**HIA? (Get Configuration Of Control Done By HID Axis)**

**Description:** Gets the current control configuration for the given motion parameter of the given axis of the E-871, i. e. the currently assigned axis of an HID device.

**Format:** HIA? [{<AxisID> <MotionParam>}]

**Arguments:** <AxisID> is one axis of the controller.

<MotionParam> is one motion parameter of the controller axis; for more details, see HIA.

**Response:** {<AxisID> <MotionParam>="<HIDDeviceID>  
<HIDDeviceAxis>LF}

where

<HIDDeviceID> is one HID device connected to the controller; for more information, see HIA.

<HIDDeviceAxis> is one axis of the HID device; for more details, see HIA.

**HIB? (Get State Of HID Button)**

**Description:** Gets the current state of the given button of the given HID device.

**Format:** HIB? [{<HIDDeviceID> <HIDDeviceButton>}]

**Arguments:** <HIDDeviceID> is one HID device connected to the controller; see below for details.

<HIDDeviceButton> is one button of the HID device; for more details, see below.

Response: {<HIDDeviceID> <HIDDeviceButton> "="<HIDButtonState>}

where

<HIDButtonState> indicates the state of the button as an integer value:

The possible values depend on the button type. The value range for the individual buttons can be queried with the HIS? command (p. 203). When only the values 0 and 1 are permitted, they have the following meaning:

0 = Button not pressed, 1 = Button pressed

The meaning of values > 1 depends on the HID device.

Notes: The E-871 supports one HID device (identifier: 1) and two buttons of this HID device (identifiers: 1 and 2). Information on the supported buttons of the HID device can be queried with the HIS? command (p. 203). Further information see "Commandable Items" (p. 15) and "Connecting an HID Device" (p. 66).

### HIE? (Get Deflection Of HID Axis)

Description: Gets the current displacement of the given axis of the given HID device.

Format: HIE? [{<HIDDeviceID> <HIDDeviceAxis>}]

Arguments: <HIDDeviceID> is one HID device connected to the controller; see below for details.

<HIDDeviceAxis> is one axis of the HID device; for more details, see below.

Response: {<HIDDeviceID> <HIDDeviceAxis> "="<HIDDeflection>}

where

<HIDDeflection> indicates the current displacement of the axis of the HID device; for more information, see below.

Notes: The E-871 supports one HID device (identifier: 1) and four axes of this HID device (identifiers: 1 to 4). Information on the supported axes of the HID device can be queried with the HIS? command (p. 203). Further information see "Commandable Items" (p. 15) and "Connecting an HID Device" (p. 66).

<HIDDeflection> indicates the current displacement of the axis of the HID device as a floating-point number in the range from -1.0 to 1.0.

For HID device axes with hard stops, the value -1.0 corresponds to the maximum displacement in the negative direction and the value 1.0 corresponds to the maximum displacement in the positive direction.

The E-871 processes the information received by the HID device so that 256 different displacement values can be shown. When the HID control takes place for a motion parameter on the basis of lookup tables, exactly one point in the currently assigned lookup table is assigned to each of these displacement values (see HDT (p. 195) and HIT (p. 206) for more information).

Example:

Send: `HIE? 1 1 1 2`

Receive: `1 1=0.02`

`1 2=-0.7`

Note: Displacement of axes 1 and 2 of HID device 1:  
Axis 1 has the value 0.02, which corresponds to approximately the middle position.  
Axis 2 has the value -0.7, i. e. it is displaced in the negative direction by around 2/3.

**HIN (Set Activation State For HID Control)**

Description:	<p>Enables or disables the control by HID devices ("HID control") that are connected to the controller for the given axis of the E-871.</p> <p>The HID control is configured with the HIA command (p. 197).</p>
Format:	HIN {<AxisID> <HIDControlState>}
Arguments:	<p>&lt;AxisID&gt; is one axis of the controller.</p> <p>&lt;HIDControlState&gt; is the activation state of the HID control: 0 = Control by HID devices is disabled 1 = Control by HID devices is enabled</p>
Response:	None
Notes:	<p>The E-871 supports one HID device (identifier 1). Information on the connection options, see "Connecting an HID Device" (p. 66).</p> <p>The enabled HID control will not have any effect if it has not been suitably configured with HIA.</p> <p>When the HID control is disabled, the target position is set to the current position of the controlled axis.</p> <p>No HID control is possible in open-loop operation (servo mode switched off).</p> <p>Motion commands such as MOV (p. 223) are not allowed when the HID control is enabled for the axis. Further information see "Control with an HID Device" (p. 118).</p>



**HIN? (Get Activation State Of HID Control)**

Description: Gets the activation state of the control by HID devices ("HID control") that are connected to the controller for the given axis of the E-871.

Format: HIN? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller.

Response: {<AxisID>="<HIDControlState>LF}

where

<HIDControlState> is the activation state of the HID control:  
0 = Control by HID devices is disabled  
1 = Control by HID devices is enabled

**HIS? (Get Configuration Of HI Device)**

Description: Gets the given property for the given operating element of an HID device.

Format: HIS? [{<HIDDeviceID> <HIDItemID> <HIDPropID>}]

Arguments: <HIDDeviceID> is one HID device connected to the controller; see below for details.

<HIDItemID> is one operating element of the HID device; for more information, see below.

<HIDPropID> is one property of the operating element of the HID device; for more information, see below.

Response: {<HIDDeviceID> <HIDItemID>  
<HIDPropID>="<HIDPropValue>LF}

where

<HIDPropValue> is a string with the value to which the property of the operating element is set; for more information, see below.

Supported  
operating  
elements and  
their properties

The E-871 supports one HID device (identifier 1).  
Information on the connection options, see "Connecting an  
HID Device" (p. 66).

All supported operating elements of the HID device are  
consecutively numbered for <HIDItemID>, starting with 1,  
regardless of their type.

The following properties of the HID device are shown with  
HIS?:

For <HIDPropID> = 1:

<HIDPropValue> indicates the type and the identifier of the  
operating element. Possible types:

- "Axis" = Axis of an HID device, can be e. g. a joystick  
axis or an AB rotary encoder or pulse generator
- "Button" = Button of an HID device, can be e. g. a  
pushbutton

The type is followed by the identifier, separated by an  
underscore. The identifier must be used in all relevant  
commands, in order to specifically address the operating  
element.

For <HIDPropID> = 2:

<HIDPropValue> is the value for the current state of the  
operating element. The meaning of the value depends on  
the type of operating element:

- "Axis": Current displacement of the axis; for more  
information, see HIE? (p. 200)
- "Button": Current state of the button; for more  
information, see HIB? (p. 199)

For <HIDPropID> = 3:

<HIDPropValue> is the name of an axis of the HID device

For <HIDPropID> = 4:

<HIDPropValue> is the name of the HID device

For <HIDPropID> = 5:

<HIDPropValue> indicates the smallest possible value for the state of an operating element of the "button" type.

For <HIDPropID> = 6:

<HIDPropValue> indicates the largest possible value for the state of an operating element of the "button" type.

Example:

```
HIS?
1 1 1=Axis_1
1 1 2=-0.094
1 1 3=X
1 1 4=Analog Joystick input
1 2 1=Axis_2
1 2 2=0.055
1 2 3=10V
1 2 4=Analog Joystick input
1 3 1=Axis_3
1 3 2=0.000
1 3 3=AB1
1 3 4=Analog Joystick input
1 4 1=Axis_4
1 4 2=0.000
1 4 3=AB2
1 4 4=Analog Joystick input
1 5 1=Button_1
1 5 2=0
1 5 4=Analog Joystick input
1 5 5=0
1 5 6=1
1 6 1=Button_2
1 6 2=0
1 6 4=Analog Joystick input
1 6 5=0
1 6 6=1
```

Note: Although the HID device supported by the E-871 is called **Analog Joystick input**, axes 3 and 4 of the HID

device require digital input signals (AB, TTL), see "Connecting an HID Device" (p. 66).

### HIT (Fill HID Lookup Table)

**Description:** Fills the given lookup table with values.

Lookup tables are used during the HID control of several motion parameters of the axes of the E-871; for details, see HIA (p. 197). A lookup table maps the displacement of the axis of an HID device to the controlled motion parameter (for further details, see HIE? (p. 200)).

With the HDT command (p. 195), the lookup tables are assigned to the axes of HID devices.

**Format:** HIT {<HIDTableID> <HIDTableAddr> <HIDTableValue>}

**Arguments:** <HIDTableID> is one lookup table of the controller; for more information, see below.

<HIDTableAddr> is the index of a point in the lookup table, begins with 1, for number of points per table, see below.

<HIDTableValue> is the value of point n as a floating-point number in the range from -1.0 to 1.0; for further information, see below.

**Response:** None

**Notes:** The lookup tables are only filled by HIT in the volatile memory (RAM) of the E-871. With the WPA command (p. 256), the currently valid table contents can be saved in the nonvolatile memory of the E-871.

The value of one point can be sent to the E-871 per HIT command.

Available lookup tables: The E-871 provides the following lookup tables with 256 points each:

Identifier	Type
1	linear
2	parabolic (default)
101	user-defined
102	user-defined

HIT can only be used to fill user-defined tables. Tables with a  $\leq 100$  identifier are predefined and write-protected.

The first point of a lookup table corresponds to the maximum axis displacement of the HID device in the negative direction; the 256th point corresponds to the maximum displacement in the positive direction. The values for points 1 to maximally 127 have a negative sign by default, while the remaining values have a positive sign.

The ***Invert Direction Of Motion For Joystick-Controlled Axis?*** parameter (ID 0x61) can be used to invert the direction of motion for the axis of the E-871 when the HID control is enabled.

### HIT? (Get HID Lookup Table Values)

Description: Gets the values of the given points in the given lookup table.

Format: HIT? [<StartPoint> [<NumberOfPoints> [{<HIDTableID>}]]]

Arguments: <StartPoint> is the index of the first point to be queried in the lookup table, smallest possible value is 1.

<NumberOfPoints> gives the number of the points to be queried per lookup table; for more information, see HIT.

<HIDTableID> is one lookup table of the controller; for more information, see HIT.

Response: The lookup table content in GCS array format, see the separate manual for GCS array, SM 146E, and the example below.

Example:

```
hit?
# TYPE = 1
#
# SEPARATOR = 32
# DIM = 4
# NDATA = 256
# NAME0 = Table 1
# NAME1 = Table 2
# NAME2 = Table 101
# NAME3 = Table 102
# END_HEADER
-1.0000 -1.0000 -1.0000 -1.0000
-0.9922 -0.9834 -0.9834 -0.9834
-0.9834 -0.9678 -0.9678 -0.9678
-0.9756 -0.9521 -0.9521 -0.9521
-0.9678 -0.9355 -0.9355 -0.9355
...
-0.7314 -0.5352 -0.5352 -0.5352
-0.7236 -0.5234 -0.5234 -0.5234
-0.7158 -0.5117 -0.5117 -0.5117
-0.7070 -0.5000 -0.5000 -0.5000
-0.6992 -0.4893 -0.4893 -0.4893
...
-0.5605 -0.3145 -0.3145 -0.3145
-0.5527 -0.3057 -0.3057 -0.3057
-0.5449 -0.2969 -0.2969 -0.2969
-0.5361 -0.2881 -0.2881 -0.2881
-0.5283 -0.2793 -0.2793 -0.2793
-0.5205 -0.2705 -0.2705 -0.2705
...
-0.3496 -0.1221 -0.1221 -0.1221
-0.3418 -0.1162 -0.1162 -0.1162
```



```
-0.3330 -0.1113 -0.1113 -0.1113
-0.3252 -0.1055 -0.1055 -0.1055
-0.3174 -0.1006 -0.1006 -0.1006
...
-0.1465 -0.0215 -0.0215 -0.0215
-0.1387 -0.0195 -0.0195 -0.0195
-0.1299 -0.0166 -0.0166 -0.0166
-0.1221 -0.0146 -0.0146 -0.0146
-0.1143 -0.0127 -0.0127 -0.0127
...
-0.0244 -0.0010 -0.0010 -0.0010
-0.0166 0.0000 0.0000 0.0000
-0.0078 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
0.0000 0.0000 0.0000 0.0000
0.0078 0.0000 0.0000 0.0000
0.0166 0.0000 0.0000 0.0000
0.0244 0.0010 0.0010 0.0010
0.0322 0.0010 0.0010 0.0010
0.0410 0.0020 0.0020 0.0020
...
0.1299 0.0166 0.0166 0.0166
0.1387 0.0195 0.0195 0.0195
0.1465 0.0215 0.0215 0.0215
0.1543 0.0234 0.0234 0.0234
0.1631 0.0264 0.0264 0.0264
```

```

...
0.2764 0.0762 0.0762 0.0762
0.2842 0.0811 0.0811 0.0811
0.2930 0.0859 0.0859 0.0859
0.3008 0.0908 0.0908 0.0908
0.3086 0.0957 0.0957 0.0957
...
0.4883 0.2383 0.2383 0.2383
0.4961 0.2461 0.2461 0.2461
0.5039 0.2539 0.2539 0.2539
0.5117 0.2627 0.2627 0.2627
0.5205 0.2705 0.2705 0.2705
...
0.6914 0.4775 0.4775 0.4775
0.6992 0.4893 0.4893 0.4893
0.7070 0.5000 0.5000 0.5000
0.7158 0.5117 0.5117 0.5117
0.7236 0.5234 0.5234 0.5234
...
0.9678 0.9355 0.9355 0.9355
0.9756 0.9521 0.9521 0.9521
0.9834 0.9678 0.9678 0.9678
0.9922 0.9834 0.9834 0.9834
1.0000 1.0000 1.0000 1.0000

```

**HLP? (Get List Of Available Commands)**

Description: Lists a help string which contains all commands available.

Format: HLP?

Arguments: none

Response: List of commands available

Troubleshooting: Communication breakdown



**HLT (Halt Motion Smoothly)**

**Description:** Halts the motion of given axes smoothly. For details see the notes below.

Error code 10 is set.

#24 (p. 162) and STP (p. 245) in contrast abort current motion as fast as possible for the controller without taking care of maximum velocity and acceleration.

**Format:** HLT [{<AxisID>}]

**Arguments:** <AxisID>: is one axis of the controller, if omitted all axes are halted

**Response:** none

**Troubleshooting:** Illegal axis identifier

**Notes:** Setting the velocity, acceleration and deceleration is not supported by the E-871. HLT therefore behaves identically to STP and #24 in relation to deceleration.

HLT stops all motion caused by motion commands (e.g. MOV (p. 223), MVR (p. 224), GOH (p. 192), STE (p. 244), SMO (p. 236)), commands for reference point definition (FNL (p. 187), FPL (p. 189), FRF (p. 190)) and macros (MAC (p. 216)).

After the axis has been stopped, its target position is set to its current position.

**HPA? (Get List Of Available Parameters)**

**Description:** Responds with a help string which contains all available parameters with short descriptions. For further information, see "Parameter Overview" (p. 280).

**Format:** HPA?

**Arguments:** None

Response {<PamID>="<string> LF}

where

<PamID> is the ID of one parameter, hexadecimal format

<string> is a string which describes the corresponding parameter.

The string has following format:

<CmdLevel>TAB<MaxItem>TAB<DataType>TAB<FunctionGroupDescription>TAB<ParameterDescription>[[TAB<PossibleValue>="<ValueDescription>]]

where

<CmdLevel> is the command level which allows write access to the parameter value

<MaxItem> is the maximum number of items of the same type which are affected by the parameter. With the E-871, an "item" is an axis.

<DataType> is the data type of the parameter value, can be INT, FLOAT or CHAR

<FunctionGroupDescription> is the name of the function group to which the parameter belongs

<ParameterDescription> is the parameter name

<PossibleValue> is one value from the allowed data range

<ValueDescription> is the meaning of the corresponding value

The parameters listed with HPA? can be changed and/or saved using the following commands:

SPA (p. 238) influences the parameter settings in volatile memory (RAM).

WPA (p. 256) copies parameter settings from volatile to nonvolatile memory.

SEP (p. 234) writes parameter settings directly into nonvolatile memory (without changing settings in volatile memory).

RPA (p. 230) resets volatile memory to the values from nonvolatile memory.

### JRC (Jump Relatively Depending On Condition)

**Description:** Jumps relatively depending on a given condition of the following type: one given value is compared with a queried value according to a given rule.

Can only be used in macros.

**Format:** JRC <Jump> <CMD?> <OP> <Value>

**Arguments:** <Jump> is the size of the relative jump. -1 means the macro execution pointer jumps back to the previous line, 0 means the command is executed again, which is the same behavior as with WAC (p. 255). 1 jumps to the next line, making the command unnecessary, and 2 jumps over the next command. Only jumps within the current macro are allowed.

<CMD?> is one query command in its usual syntax. The response has to be a single value and not more. For an example see below.

<OP> is the operator to be used. The following operators are possible:

= <= < > >= !=

Important: There must be a blank space before and after the operator!

<Value> is the value to be compared with the response to <CMD?>.

Response: None

Troubleshooting: Check proper jump target

Example: Using the following macro, you can stop motion of axis 1 using a stop button connected to a digital input. The stop button is checked until the axis has reached the target position (ONT? query). When the stop button is pressed as long as the target position has not been reached yet: The response to the POS? 1 query is copied to the TARGET variable. Then this variable is used as second argument for the MOV command. Thus the stage stays where it just was. To clean up, TARGET is defined as empty with the VAR command which deletes the variable.

Write the "stop" macro:

```
MAC BEG stop
MOV 1 20
JRC 2 DIO? 1 = 1
JRC -1 ONT? 1 = 0
CPY TARGET POS? 1
MOV 1 ${TARGET}
VAR TARGET
MAC END
```

### LIM? (Indicate Limit Switches)

Description: Gets whether axes have limit switches.

Format: LIM? [{<AxisID>}]

Arguments: <AxisID>: is one axis of the controller

Response: {<AxisID>="<uint> LF}

where

<uint> indicates whether the axis has limit switches (=1) or not (=0).

Troubleshooting: Illegal axis identifier

Notes: The E-871 firmware detects the presence or absence of limit switches using a parameter (ID 0x32). According to the value of this parameter, the E-871 enables or disables stopping the motion at the limit switches and reference moves using the limit switches.

Adapt the parameter value to your hardware using SPA (p. 238) or SEP (p. 234). Further information see "Limit Switch Detection" (p. 31).

You can use the digital input lines instead of the limit switches as source of the negative or positive limit switch signal. Further information see "Digital Input Signals" (p. 111).

**MAC (Call Macro Function)**

**Description:** Calls a macro function. Permits recording, deleting and running macros on the controller.

**Format:** MAC <keyword> {<parameter>}

in particular:

MAC BEG <macroname>

MAC DEF <macroname>

MAC DEF?

MAC DEL <macroname>

MAC END

MAC ERR?

MAC NSTART <macroname> <uint> [<String1>  
[<String2>]]

MAC START <macroname> [<String1> [<String2>]]

**Arguments** <keyword> determines which macro function is called. The following keywords and parameters are used:

MAC BEG <macroname>

Starts recording a macro to be named *macroname* on the controller; may not be used in a macro; the commands that follow become the macro. End the recording with MAC END. Note that erroneous macro content cannot be detected by sending the ERR? command.

MAC END

Stops macro recording (cannot become part of a macro)

MAC ERR?

Reports the last error which occurred during macro execution.

Response: <macroname> <uint1>="<uint2> <"<"CMD">">

where <macroname> is the name of the macro, <uint1> is the line in the macro, <uint2> is the error code and <"<"CMD">"> is the erroneous command which was sent to the parser.

**MAC DEF <macroname>**

Sets specified macro as start-up macro. This macro will be automatically executed with the next switching-on or reboot of the controller. If <macroname> is omitted, the current start-up macro selection is canceled.

**MAC DEF?**

Asks for the start-up macro

Response: <macroname>

If no start-up macro is defined, the response is an empty string with the terminating character.

**MAC DEL <macroname>**

Deletes specified macro.

**MAC NSTART <macroname> <uint> [<String1> [<String2>]]**

Repeats the specified macro <uint> times. Another execution is started when the last one is finished.

<String1> and <String2> are optional arguments which give the values for local variables 1 and 2 used in the given macro. <String1> and <String2> can be given directly or via the values of variables. Macro execution will fail if the macro contains local variables but <String1> and <String2> are omitted in the MAC NSTART command. See "Variables" (p. 153) for further details.

**MAC START <macroname> [<String1> [<String2>]]**

Starts one execution of the specified macro. <String1> and <String2> have the same function as with MAC NSTART.

Response: None

Troubleshooting: Macro recording is active (keywords BEG, DEL) or inactive (END)

Macro contains a disallowed MAC command

Notes: During macro recording no macro execution is allowed.

When a macro is recorded for a controller whose address is different from 1, the target address must be part of each command line, but will not become part of the macro content. PIMikroMove automatically sends the target address during the macro recording so that it does not have to be entered there. Further information see "Working with Macros" (p. 133) and "Target and Sender Address" (p. 152).

When macros are recorded on the **Controller macros** tab in PIMikroMove, the `MAC BEG` and `MAC END` commands must be omitted.

A macro can be overwritten by a macro with the same name.

Macros can contain local and global variables. For more information, see "Variables" (p. 153).

A running macro sends no responses to any interface.

Depending on the value of parameter 0x72 (**Ignore Macro Error?**), the following options exist when an error is caused by a running macro:

0 = Macro execution is aborted (default).

1 = The error is ignored and the macro execution is continued.

Irrespective of the parameter setting, MAC ERR? always reports the last error that occurred during a macro execution.

The following commands provided by the E-871 can only be used in macros:

DEL (p. 173), JRC (p. 213), MEX (p. 221) and WAC (p. 255).

A macro can start another macro. The maximum number of



nesting levels is 5. A macro can call itself to form an infinite loop.

Any commands can be sent from the command line when a macro is running. The macro content and motion commands received from the command line can overwrite each other.

Macro execution can be stopped with #24 (p. 162) and STP (p. 245).

Simultaneous execution of multiple macros is not possible. Only one macro can be executed at a time.

A running macro may not be deleted.

You can query with #8 (p. 161) if a macro is currently running on the controller.

**Note: The number of write cycles in the nonvolatile memory is limited. Only record macros if this is necessary.**

### MAC? (List Macros)

Description:	Lists macros or content of a given macro.
Format:	MAC? [<macroname>]
Arguments	<macroname>: name of the macro whose content shall be listed; if omitted, the names of all stored macros are listed.
Response:	<string>
	If <macroname> was given, <string> is the content of this macro;
	If <macroname> was omitted, <string> is a list with the names of all stored macros
Troubleshooting:	Macro <macroname> not found

**MAN? (Get Help String For Command)**

Description:	Shows a detailed help text for individual commands.
Format:	MAN? <CMD>
Arguments:	<CMD> is the command mnemonic of the command for which the help text is to be displayed (see below).
Response:	A string that describes the command.
Notes:	<p>A detailed help text can be displayed for the following GCS commands:</p> <p>CTO, CTO?, HIA, HIA?, HIS, HIS?, HIT, HIT?, WPA</p>
Example:	<p>Send: MAN? CTO?</p> <p>Receive:</p> <pre>CTO {&lt;TrigOutID&gt; &lt;CTOPam&gt; &lt;Value&gt;} Set Configuration Of Trigger Output #AvailableCTOparameters &lt;CTOPam&gt; &lt;Description&gt; 1 Trigger Step 2 Axis 3 Trigger Mode 7 Polarity 8 Start Threshold 9 Stop Threshold 10 Trigger Position #AvailableTriggerModes &lt;Value&gt; &lt;Description&gt; 0 Position Distance 2 On Target 5 Motion Error 6 In Motion 7 Position+Offset 8 Single Position #AvailablePolarities &lt;Value&gt; &lt;Description&gt;</pre>

```
0 Active Low
```

```
1 Active High
```

```
end of help
```

### MEX (Stop Macro Execution Due To Condition)

**Description:** Stops macro execution due to a given condition of the following type: a given value is compared with a queried value according to a given rule.

Can only be used in macros.

When the macro interpreter accesses this command, the condition is checked. If it is true, the current macro is stopped; otherwise macro execution is continued with the next line. Should the condition be fulfilled later, the interpreter will ignore it.

See also the WAC command (p. 255).

**Format:** MEX <CMD?> <OP> <Value>

**Arguments** <CMD?> is one query command in its usual syntax. The response has to be a single value and not more. For an example see below.

<OP> is the operator to be used. The following operators are possible:

= <= < > >= !=

Important: There must be a blank space before and after the operator!

<Value> is the value that is compared with the response to <CMD?>.

**Response:** None

Example: Send: `MAC START LOOP`

Note: Macro LOOP has the following contents:

```
MAC START KEY1
MAC START KEY2
MEX DIO? 4 = 1
MAC START LOOP
```

Macro KEY1 has the following contents:

```
MEX DIO? 4 = 1
MEX DIO? 1 = 0
MVR 1 1.0
DEL 100
```

Macro KEY2 has the following content:

```
MEX DIO? 4 = 1
MEX DIO? 2 = 0
MVR 1 -1.0
DEL 100
```

Macro LOOP forms an infinite loop by permanently calling KEY1, KEY2 and itself.

KEY1 checks the state of the digital input channel 1 (located on the **I/O** socket (p. 310)). If it is not set (0), the macro is aborted, otherwise the macro will move axis 1 by 1.0 in positive direction (relative move).

KEY2 checks the state of the digital input channel 2 and moves axis 1 in negative direction accordingly.

Connecting the digital input channels 1, 2 and 4 with pushbuttons, e.g. with the C-170.PB pushbutton box, it is possible to implement interactive control of an axis without any software assistance. The delay (DEL 100) is required to avoid generation of multiple MVR commands while pressing the pushbutton for a short time.

Channel 4 is used as a global exit. Since MEX stops execution of the current macro only, it must also be included in the calling macro, which would otherwise continue.

**MOV (Set Target Position)**

Description:	Set new absolute target position for given axis.  Servo mode must be switched on for the commanded axis prior to using this command (closed-loop operation).
Format:	MOV {<AxisID> <Position>}
Arguments	<AxisID> is one axis of the controller  <Position> is the new absolute target position in physical units.
Response:	none
Notes:	<p>The target position must be within the soft limits. Use TMN? (p. 248) and TMX? (p. 249) to get the currently valid soft limits.</p> <p>The motion can be stopped by #24 (p. 162), STP (p. 245) and HLT (p. 211).</p> <p>During a motion, a new motion command resets the target to a new value and the old value may never be reached. This is also valid with macros: motion commands can be sent from the command line when a macro is running. The macro content and motion commands received from the command line can overwrite each other.</p> <p>Motion commands such as MOV are not allowed when the HID control is enabled for the axis. Further information see "Control with an HID Device" (p. 118).</p>
Example 1:	<p>Send: <code>MOV 1 10</code></p> <p>Note: Axis 1 moves to 10 (target position in mm)</p>

Example 2: Send: `MOV 1 243`

Send: `ERR?`

Receive: `7`

Note: The axis does not move. The error code "7" in the reply to the ERR? command (p. 185) indicates that the target position given in the motion command is out of limits.

### MOV? (Get Target Position)

Description: Returns last valid commanded target position.

Format: `MOV? [{<AxisID>}]`

Arguments: <AxisID> is one axis of the controller

Response: `{<AxisID>="<float> LF}`

where

<float> is the last commanded target position in physical units

Troubleshooting: Illegal axis identifier

Notes: The target position can be changed by commands that cause motion (e.g. MOV (p. 223), MVR (p. 224), MVE, GOH (p. 244), STE (p. 192)) or by the HID control (when the HID control is disabled, the target position is set to the current position for HID-controlled axes in closed-loop operation).

MOV? gets the commanded positions. Use POS? (p. 227) to get the current positions.

### MVR (Set Target Relative To Current Position)

Description: Moves the given axis relative to the last commanded target position.

Servo mode must be switched on for the commanded axis prior to using this command (closed-loop operation).

Format:	MVR {<AxisID> <Distance>}
Arguments:	<p>&lt;AxisID&gt; is one axis of the controller.</p> <p>&lt;Distance&gt; gives the distance that the axis is to move; the sum of the distance and the last commanded target position is set as new target position (in physical units).</p>
Response:	None
Notes:	<p>The target position must be within the soft limits. Use TMN? (p. 248) and TMX? (p. 249) to get the currently valid soft limits, and MOV? (p. 224) to get the current target.</p> <p>The motion can be stopped by #24 (p. 162), STP (p. 245) and HLT (p. 211).</p> <p>During a motion, a new motion command resets the target to a new value and the old value may never be reached. This is also valid with macros: motion commands can be sent from the command line when a macro is running. The macro content and motion commands received from the command line can overwrite each other.</p> <p>Motion commands such as MVR are not allowed when the HID control is enabled for the axis. Further information see "Control with an HID Device" (p. 118).</p>
Example:	<p>Send: MOV 1 0.5</p> <p>Note: This is an absolute motion.</p> <p>Send: POS? 1</p> <p>Receive: 1=0.500000</p> <p>Send: MOV? 1</p> <p>Receive: 1=0.500000</p> <p>Send: MVR 1 2</p> <p>Note: This is a relative motion.</p> <p>Send: POS? 1</p> <p>Receive: 1=2.500000</p> <p>Send: MVR 1 2000</p>

Note: New target position of axis 1 would exceed motion range. Command is ignored, i. e. the target position remains unchanged, and the axis does not move.

Send: `MOV? 1`

Receive: `1=2.500000`

Send: `POS? 1`

Receive: `1=2.500000`

### ONT? (Get On-Target State)

Description: Gets on-target state of given axis.

If all arguments are omitted, gets state of all axes.

Format: `ONT? [{<AxisID>}]`

Arguments: <AxisID> is one axis of the controller.

Response: `{<AxisID>="<uint> LF}`

where

<uint> = "1" when the specified axis is on target, "0" otherwise.

Troubleshooting: Illegal axis identifier

Notes: The detection of the on-target state is only possible in closed-loop operation (servo mode ON).

The on-target state is influenced by the settings for the settling window (parameter 0x36) and the delay time (parameter 0x3F). Details see "On-Target State" (p. 29).



**POS (Set Real Position)**

Description:	Sets the current position (does not cause motion).
Format:	POS { <AxisID> <Position> }
Arguments:	<AxisID> is one axis of the controller.  <Position> is the new current position in physical units.
Response:	none
Troubleshooting:	Illegal axis identifier
Notes:	It is only possible to set the current position with POS when the mode of reference point definition is set to "0", see RON (p. 229).  An axis is considered to be "referenced" when the position has been set with POS (for more information, see "Reference Point Definition" (p. 37)).  The minimum and maximum commandable positions (TMN? (p. 248), TMX? (p. 249)) are not adapted when a position is set with POS. This can result in target positions which are allowed by the E-871 but cannot be reached by the hardware. Target positions are also possible that can be reached by the hardware but are refused by the E-871. Furthermore, the zero position can be outside of the physical travel range after using POS.

**POS? (Get Real Position)**

Description:	Gets the current axis position.  If all arguments are omitted, the current position of all axes is queried.
Format:	POS? [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller.

Response:            {<AxisID>=" "<float> LF}

where

<float> is the current axis position in physical units.

Troubleshooting:   Illegal axis identifier

### **RBT (Reboot System)**

Description:        Reboots system. Controller behaves just like after switching-on.

Format:             RBT

Arguments:         none

Response:           none

Notes:             RBT cannot be used in macros. This is to avoid problems with start-up macro execution.

### **RMC? (List Running Macros)**

Description:        Lists macros which are currently running.

Format:             RMC?

Arguments:         none

Response:           {<macroname> LF}

where

<macroname> is the name of one macro which is saved on the controller and currently running. The response is an empty line when no macro is running.

**RON (Set Reference Mode)**

Description:	Sets mode of reference point definition of given axes.
Format:	RON {<AxisID> <ReferenceOn>}
Arguments:	<p>&lt;AxisID&gt; is one axis of the controller.</p> <p>&lt;ReferenceOn&gt; can be 0 or 1. 1 is default. Details see below.</p>
Response:	None
Troubleshooting:	Illegal axis identifier
Notes:	<p>&lt;ReferenceOn&gt; = 0: To define the reference point of the axis, an absolute position value can be assigned with POS (p. 227) or a reference move can be started with FRF (p. 190), FNL (p. 187) or FPL (p. 189). Relative motions with MVR are possible, even when the reference point for the axis has not been defined yet.</p> <p>&lt;ReferenceOn&gt; = 1: To define the reference point of the axis, a reference move must be started with FRF, FNL or FPL. Using POS is not permitted. Motions in closed-loop operation are only possible when the reference point for the axis has been defined.</p> <p>Further information, see "Reference Point Definition" (p. 37) and "Travel Range and Soft Limits" (p. 32).</p>

**RON? (Get Reference Mode)**

Description:	Gets mode of reference point definition of given axes.
Format:	RON? [{ <AxisID>}]
Arguments:	<AxisID> is one axis of the controller.

Response: {<AxisID>="<ReferenceOn> LF}

where

<ReferenceOn> is the currently set mode of reference point definition for the axis

Troubleshooting: Illegal axis identifier

Note: You can find further information in the description of the RON command (p. 229).

### RPA (Reset Volatile Memory Parameters)

Description: Resets the given parameter of the given item. The value from nonvolatile memory is written into volatile memory.

Related commands:

With HPA? (p. 211) you can obtain a list of the available parameters. SPA (p. 238) influences the parameter settings in volatile memory, WPA (p. 256) writes parameter settings from volatile to nonvolatile memory, and SEP (p. 234) writes parameter settings directly into nonvolatile memory (without changing the settings in volatile memory).

See SPA for an example.

Format: RPA [{<ItemID> <PamID>}]

Arguments: <ItemID> is the item for which a parameter is to be reset. See below for details.

<PamID> is the parameter ID, can be written in hexadecimal or decimal format. See below for details.

Response: none

Troubleshooting: Illegal item identifier, wrong parameter ID

**Notes:** The information from the ID chip of the stage and from stage databases is only loaded to the volatile memory of the E-871. The loaded data is overwritten by RPA. Use RPA only if you are sure that the E-871 functions correctly with the parameter values from the nonvolatile memory.

With the E-871, you can reset either all parameters or specifically one single parameter with RPA.

**Available item IDs and parameter IDs:** An item is an axis; the identifier can be changed with SAI (p. 232). Further information see "Commandable Items" (p. 15).

Valid parameter IDs are given in "Parameter Overview" (p. 280).

#### **RTR (Set Record Table Rate)**

**Description:** Sets the record table rate, i.e., the number of cycles to be used in data recording operations. Settings larger than 1 make it possible to cover longer time periods.

**Format:** RTR <RecordTableRate>

**Arguments:** <RecordTableRate> is the record table rate to be used for recording operations (unit: number of cycles), must be an integer value larger than zero.

**Response:** None

Notes: The duration of the recording can be calculated as follows:

Rec. duration = cycle time of the servo loop \* RTR value \*  
number of points

where

the cycle time of the servo loop for the E-871 is 50  $\mu$ s

the number of points for the E-871 is 1024 (length of data  
recorder table)

For more information see "Data Recorder" (p. 100).

The record table rate set with RTR is saved in volatile  
memory (RAM) only.

#### **RTR? (Get Record Table Rate)**

Description: Gets the current record table rate, i.e., the number of cycles  
used in data recording operations.

Format: RTR?

Arguments: None

Response: <RecordTableRate> is the table rate used for recording  
operations (unit: number of cycles).

#### **SAI (Set Current Axis Identifiers)**

Description: Sets the axis identifiers for the given axes.

After it was set with SAI, the new axis identifier must be  
used as <AxisID> in all axis-related commands.

Format: SAI {<AxisID> <NewIdentifier>}

Arguments:	<p>&lt;AxisID&gt; is one axis of the controller</p> <p>&lt;NewIdentifier&gt; is the new identifier to use for the axis, see below for details</p>
Response:	none
Notes:	<p>An axis could be identified with up to 8 characters. Use TVI? (p. 252) to ask for valid characters.</p> <p>The new axis identifier is saved automatically and thus still available after reboot or next switching-on.</p>

#### SAI? (Get List Of Current Axis Identifiers)

Description:	<p>Gets the axis identifiers.</p> <p>See also "Commandable Items" (p. 15).</p>
Format:	SAI? [ALL]
Arguments:	[ALL] is optional. For controllers which allow for axis deactivation, [ALL] ensures that the answer also includes the axes which are "deactivated".
Response:	<p>{&lt;AxisID&gt; LF}</p> <p>&lt;AxisID&gt; is one axis of the controller.</p>
Notes:	If the <b>Stage Name</b> parameter (0x3C) has the value NOSTAGE, the axis is "deactivated". A deactivated axis is not accessible for axis-related commands (e.g. motion commands or position queries) and is only included in the response to SAI? ALL.

**SEP (Set Nonvolatile Memory Parameters)**

**Description:** Sets a parameter of a given item to a different value in nonvolatile memory, where it becomes the new default.

After parameters were set with SEP, you can use RPA (p. 230) to activate them (write them to volatile memory) without controller reboot.

**Note that this command is for setting hardware-specific parameters. Wrong values may lead to improper operation or damage of your hardware!**

**Related commands:**

HPA? (p. 211) returns a list of the available parameters.

SPA (p. 238) writes parameter settings into volatile memory (without changing the settings in nonvolatile memory).

WPA (p. 256) writes parameter settings from volatile to nonvolatile memory.

See SPA for an example.

**Format:** SEP <Pswd> {<ItemID> <PamID> <PamValue>}

**Arguments**

<Pswd> is the password for writing to nonvolatile memory, default is "100".

<ItemID> is the item for which a parameter is to be changed in nonvolatile memory. See below for details.

<PamID> is the parameter ID, can be written in hexadecimal or decimal format. See below for details.

<PamValue> is the value to which the given parameter of the given item is set.

**Response:** none



Troubleshooting: Illegal item identifier, wrong parameter ID, invalid password

Notes: **Note that the number of write cycles in the nonvolatile memory is limited. Write default settings only if necessary.**

With the E-871, you can write only one single parameter per SEP command.

Available item IDs and parameter IDs: An item is an axis, the identifier can be changed with SAI (p. 232). For further information see "Commandable Items" (p. 15).

Valid parameter IDs are given in "Parameter Overview" (p. 280).

### SEP? (Get Nonvolatile Memory Parameters)

Description: Gets the value of a parameter of a given item from nonvolatile memory.

With HPA? (p. 211) you can obtain a list of the available parameters and their IDs.

Format: SEP? [{<ItemID> <PamID>}]

Arguments: <ItemID> is the item for which a parameter value from nonvolatile memory is to be queried. See below for details.

<PamID> is the parameter ID, can be written in hexadecimal or decimal format. See below for details.

Response: {<ItemID> <PamID>="<PamValue> LF}

where

<PamValue> is the value of the given parameter for the given item

Troubleshooting:	Illegal item identifier, wrong parameter ID
Notes:	With the E-871, you can query either all parameters or one single parameter per SEP? command.
Available item IDs and parameter IDs:	An item is an axis, the identifier can be changed with SAI (p. 232). For further information see "Commandable Items" (p. 15).  Valid parameter IDs are given in "Parameter Overview" (p. 280).

### SMO (Set Open-Loop Control Value)

Description:	Sets control value directly to move the axis. Profile generator (if present), sensor feedback and servo algorithm are not taken into account.  Servo mode must be switched off when using this command (open-loop operation).
Format:	SMO {<AxisID> <ControlValue>}
Arguments	<AxisID> is one axis of the controller.  <ControlValue> is the new control value (dimensionless). See below for details.
Response:	None
Troubleshooting:	Illegal axis identifier  Servo mode is switched on for one of the specified axes.

## Notes:

The unsigned control value must not be larger than the value of the **Maximum Motor Output** parameter (0x9).

<ControlValue> controls the PIShift drive electronics for the axis. The interpretation of <ControlValue> by the PIShift drive electronics depends on the value of parameter 0x1F000702, which determines the PIShift drive mode for open-loop operation.

- 0x1F000702 has the value 0 (step mode):  
<ControlValue> directly gives the step frequency in Hz and thus the velocity of the axis. The sign of the value determines the direction of motion.  
In addition to the parameter 0x9, the parameter 0x1F000400 also limits the step frequency.  
Depending on the current step frequency, the PIShift drive can develop noises.
- 0x1F000702 has the value 1 (linear mode):  
<ControlValue> specifies the output piezo voltage and thus the expansion of the piezo actuator in the PIShift drive. 32767 corresponds to the maximum value of the piezo voltage (parameter 0x1F000000), 0 corresponds to the minimum value of the piezo voltage (parameter 0x1F000100). Negative values are not permitted.

The **Range Limit Min** (0x07000000) and **Range Limit Max** (0x07000001) parameters can be used as soft limits for motions in open-loop operation with SMO: When the current position reaches these values, the control value is set to zero and the motion is thus stopped. The axis can be moved again as soon as the value for the soft limit has been increased or decreased.

## Example:

Send: SPA? 1 0x1F000702

Receive: 1 0x1F000702=0

Note: SMO triggers motions in step mode.

Send: SMO 1 -16000

Note: The axis moves in the negative direction with a step frequency of 16000 Hz.

**SMO? (Get Control Value)**

Description: Gets last valid control value of given axis.

Format: SMO? [{<AxisID>}]

Arguments <AxisID> is one axis of the controller

Response: {<AxisID>="<float> LF}

where

<float> is the last valid control value (dimensionless). For details see below.

Troubleshooting: Illegal axis identifier

Notes: The control value which is returned by SMO? can be the result of the servo algorithm and other corrections, or it can be the value set by an SMO command in open-loop operation. See the block diagram (p. 15) for further information.

See SMO (p. 236) for more information.

**SPA (Set Volatile Memory Parameters)**

Description: Sets a parameter of the given item in the volatile memory (RAM) to a specific value. Parameter changes are lost when the controller is switched off or rebooted.

Format: SPA {<ItemID> <PamID> <PamValue>}

Arguments:	<p>&lt;ItemID&gt; is the item for which a parameter is changed in volatile memory. See below for details.</p> <p>&lt;PamID&gt; is the parameter ID, can be written in hexadecimal or decimal format. See below for details.</p> <p>&lt;PamValue&gt; is the value to which the given parameter of the given item is set.</p>
Response:	<p>None</p> <p>Parameter changes are also lost when the parameters are restored with RPA (p. 230).</p> <p><b>Note that this command is for setting hardware-specific parameters. Wrong values may lead to improper operation or damage of your hardware!</b></p> <p>Related commands:</p> <p>HPA? (p. 211) returns a list of the available parameters.</p> <p>SEP (p. 234) writes parameter settings directly into nonvolatile memory (without changing the settings in volatile memory).</p> <p>WPA (p. 256) writes parameter settings from volatile to nonvolatile memory.</p> <p>RPA resets volatile memory to the value in nonvolatile memory.</p>
Troubleshooting:	<p>Illegal item identifier, wrong parameter ID, value out of range</p>
Notes:	<p>With the E-871, you can write only one single parameter per SPA command.</p>

Available item IDs and parameter IDs: An item is an axis; the identifier can be changed with SAI (p. 232). Further information see "Commandable Items" (p. 15).

Valid parameter IDs can be found in the parameter overview.

Example 1: Send: `SPA 1 0x1 10`

Note: Sets the P-Term of the servo algorithm for axis 1 to 10; the parameter ID is written in hexadecimal format

Send: `SPA 1 1 50`

Note: Sets the P-term of the servo algorithm for axis 1 to 50; the parameter ID is written in decimal format

Example 2: The notch filter frequency as well as the P and I parameters of the servo algorithm must be adapted to a new load that is applied to the connected mechanical system.

Send: `SPA 1 0x94 180`

Note: The notch filter frequency is set to 180 Hz for axis 1 (can be identified by recording the step response in open-loop operation (p. 89)). The setting is made in volatile memory only.

Now set the P and I terms of the servo algorithm in the volatile memory using SPA and then test the functioning of the system in closed-loop operation. If the closed-loop system performance proves satisfactory and you want to use this system configuration as default, save the parameter settings from volatile to nonvolatile memory.

Send: `WPA 100`

Note: See the command description for WPA (p. 256) for details on the extent of the saved settings.

**SPA? (Get Volatile Memory Parameters)**

**Description:** Gets the value of a parameter of a given item from volatile memory (RAM).

With HPA? (p. 211) you can obtain a list of the available parameters.

**Format:** SPA? [{<ItemID> <PamID>}]

**Arguments:** <ItemID> is the item for which a parameter is to be queried in volatile memory. See below for details.

<PamID> is the parameter identifier, can be written in hexadecimal or decimal format. See below for details.

**Response:** {<ItemID> <PamID> "="<PamValue> LF}

where

<PamValue> is the value of the given parameter for the given item

**Troubleshooting:** Illegal item identifier, wrong parameter ID

**Notes:** With the E-871, you can query either all parameters or one single parameter per SPA? command.

**Available item IDs and parameter IDs:** An item is an axis, the identifier can be changed with SAI (p. 232). For further information see "Commandable Items" (p. 15).

Valid parameter IDs are given in "Parameter Overview" (p. 280).

**SRG? (Query Status Register Value)**

**Description:** Returns register values for queried axes and registers.

**Format:** SRG? {<AxisID> <RegisterID>}

Arguments: <AxisID> is one axis of the controller.

<RegisterID>: is the ID of the specified register; see below for available registers.

Response: {<AxisID><RegisterID>="<Value> LF}

where

<Value> is the value of the register; see below for details.

Note: This command is identical in function to #4 (p. 159) which should be preferred when the controller is performing time-consuming tasks.

Possible register <RegisterID> can be 1.

IDs and response

values:

<Value> is the bit-mapped answer and returned as the sum of the individual codes, in hexadecimal format:

Bit	15	14	13	12	11	10	9	8
Description	On-target state	Determines the reference value	In motion	Servo mode on	-	-	-	Error flag
Bit	7	6	5	4	3	2	1	0
Description	Digital input line 4	Digital input line 3	Digital input line 2	Digital input line 1	-	Positive limit switch	Reference point switch	Negative limit switch

Example: Send: SRG? 1 1

Receive: 1 1=0x9002

Note: The response is given in hexadecimal format. It means that axis 1 is on target, the servo mode is ON for that axis, no error occurred, the states of the digital input lines 1 to 4 are low, and axis 1 is on the positive side of the reference point switch.



**SST (Set Step Size)**

Description: Sets the distance ("step size") for motions of the given axis that are triggered by a manual control unit.

Format: SST {<AxisID> <StepSize>}

Arguments: <AxisID> is one axis of the controller

<StepSize> is the distance, format: float

Response: None

Troubleshooting: Illegal value  
Illegal axis identifier

Note: The distance set with SST is used when relative motions of the axis of the E-871 are triggered by an axis of the HID device. For details, see HIA (p. 197).

<StepSize> is given in the physical unit of the axis position.

**SST? (Get Step Size)**

Description: Gets the distance ("step size") for motions of the given axis that are triggered by a manual control unit.

Format: SST? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

Response: {<AxisID>="<StepSize> LF}

where

<StepSize> is the distance in physical units, see SST (p. 243).

### STE (Start Step And Response Measurement)

Description:	<p>Starts performing a step and recording the step response for the given axis.</p> <p>The data recorder configuration, i. e. the assignment of data sources and record options to the recorder tables, can be set with DRC (p. 178).</p> <p>The recorded data can be read with the DRR? command (p. 181).</p>
Format:	STE <AxisID> <Amplitude>
Arguments:	<p>&lt;AxisID&gt; is one axis of the controller</p> <p>&lt;Amplitude&gt; is the size of the step. See below for details.</p>
Response:	None
Troubleshooting:	<p>In closed-loop operation, the target position must be within the soft limits. Use TMN? (p. 248) and TMX? (p. 249) to get the currently valid soft limits, and MOV? (p. 224) to get the current target.</p> <p>Motion commands such as STE are not allowed when the HID control is enabled for the axis. Further information see "Control with an HID Device" (p. 118).</p>
Notes:	<p>A "step" consists of a motion with the specified amplitude which is performed relative to the current position.</p> <p>How the value for &lt;Amplitude&gt; is interpreted depends on the current servo mode:</p> <ul style="list-style-type: none"> <li>▪ Closed-loop operation: &lt;Amplitude&gt; gives the distance for the step (in physical units). Floating point numbers are allowed.</li> <li>▪ Open-loop operation: &lt;Amplitude&gt; gives the number of steps to be moved (execution in step mode (p. 20)) and must be an integer value. The step frequency is given by the value of parameter 0x1F000400.</li> </ul>

**STP (Stop All Axes)**

Description: Stops all axes abruptly. For details see the notes below.

Sets error code to 10.

This command is identical in function to #24 (p. 162).

Format: STP

Arguments: None

Response: None

Troubleshooting: Communication breakdown

Notes: STE stops all motion caused by motion commands (e.g. MOV (p. 223), MVR (p. 224), GOH (p. 192), STE (p. 244), SMO (p. 236)), commands for reference point definition (FNL (p. 187), FPL (p. 189), FRF (p. 190)) and macros (MAC (p. 216)). Also stops macro execution.

After the axis has been stopped, its target position is set to its current position.

**SVO (Set Servo Mode)**

Description: Sets the servo mode for given axes (open-loop or closed-loop operation).

Format: SVO {<AxisID> <ServoState>}

Arguments: <AxisID> is one axis of the controller

<ServoState> can have the following values:  
0 = servo mode off (open-loop operation)  
1 = servo mode on (closed-loop operation)

Response: None

Troubleshooting: Illegal axis identifier

Notes: When switching from open-loop to closed-loop operation, the target is set to the current position to avoid jumps of the mechanical system.

The current state of the servo mode determines the applicable motion commands:

Servo mode ON: use the commands MOV (p. 223), MVR (p. 224), GOH (p. 192), STE (p. 244) or the HID control (p. 118).

Servo mode OFF: use SMO (p. 236) or STE (p. 244).

The servo mode must be switched on before reference moves can be started with FRF (p. 190), FNL (p. 187) or FPL (p. 189).

When the servo mode is switched off while the axis is moving, the axis stops.

Using a startup macro, you can configure the controller so that servo mode is automatically switched on upon switching-on or reboot. For more information, see "Setting up a Start-Up Macro" (p. 141).

### **SVO? (Get Servo Mode)**

Description: Gets the servo mode for the axes specified.

If all arguments are omitted, gets the servo mode of all axes.

Format: SVO? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller.

Response: {<AxisID>="<ServoState> LF}

where

<ServoState> is the current servo mode for the axis:

0 = servo mode off (open-loop operation)

1 = servo mode on (closed-loop operation)

Troubleshooting: Illegal axis identifier

### TAC? (Tell Analog Channels)

**Description:** Gets the number of installed analog lines.

**Format:** TAC?

**Arguments:** None

**Response:** <uint> indicates the total number of analog lines (inputs and outputs).

**Notes:** Gets the number of analog input lines located on the **I/O** socket (p. 310) of the E-871 (Input 1 to Input 4). Note that these lines can also be used for digital input. For further information see "Commandable Items" (p. 15).

### TAV? (Get Analog Input Voltage)

**Description:** Get voltage at analog input.

**Format:** TAV? [{<AnalogInputID>}]

**Arguments:** <AnalogInputID> is the identifier of the analog input channel; see below for details.

**Response:** {<AnalogInputID>="<float> LF}

where

<float> is the current voltage at the analog input in volts

**Notes:** Using the TAV? command, you can directly read the Input 1 to Input 4 lines on the **I/O** socket (p. 310) of the E-871. The identifiers of the lines are 1 to 4. For further information see "Commandable Items" (p. 15).

You can record the values of the analog input lines using the DRC record option 81 (p. 178).

**TIO? (Tell Digital I/O Lines)**

Description: Tells number of installed digital I/O lines

Format: TIO?

Arguments: none

Response: I=<uint1>  
O=<uint2>

where

<uint1> is the number of digital input lines.  
<uint2> is the number of digital output lines.

Notes: The digital output lines reported by TIO? are Output 1 to Output 4. The states of the Output 1 to Output 4 lines can be set using the DIO command (p. 176). Furthermore, you can program the Output 1 to Output 4 lines using the CTO command (p. 169) (trigger configuration) and the TRO command (p. 250) (trigger enabling/disabling).

The digital input lines reported by TIO? are Input 1 to Input 4. They can be read with DIO? (p. 177), #4 (p. 159) and SRG? (p. 241). In addition, you can use the lines Input 1 and 2 or Input 3 and 4 for the HID control of the axis of the E-871. Details see HIA (p. 197) and "Connecting an HID Device" (p. 66).

All the lines are located on the **I/O** socket (p. 310) of the E-871.

**TMN? (Get Minimum Commandable Position)**

Description: Get the minimum commandable position in physical units.

Format: TMN? [{ <AxisID>}]

Arguments: <AxisID> is one axis of the controller

Response {<AxisID>="<float> LF}

where

<float> is the minimum commandable position in physical units

Note: The minimum commandable position is defined by the parameter 0x30.

#### **TMX? (Get Maximum Commandable Position)**

Description: Get the maximum commandable position in physical units.

Format: TMX? [{ <AxisID>}]

Arguments: <AxisID> is one axis of the controller

Response {<AxisID>="<float> LF}

where

<float> is the maximum commandable position in physical units

Note: The maximum commandable position is defined by the parameter 0x15.

#### **TNR? (Get Number of Record Tables)**

Description: Gets the number of data recorder tables currently available on the controller.

Format: TNR?

Arguments: none

Response: <uint> is the number of data recorder tables which are currently available

Notes: The E-871 has 2 data recorder tables with 1024 data points per table.

For more information see "Data Recorder".

### **TRO (Set Trigger Output State)**

Description: Enables or disables the trigger output conditions which were set with CTO (p. 169) for the given digital output line.

Format: TRO {<TrigOutID> <TrigMode>}

Arguments: <TrigOutID> is a digital output line of the controller; see below for further details.

<TrigMode> can have the following values:

0 = Trigger output disabled

1 = Trigger output enabled

Response: None

Troubleshooting: Illegal identifier of the digital output line

Notes: <TrigOutID> corresponds to the digital output lines Output 1 to Output 4, IDs = 1 to 4; for further information see "I/O" (p. 310).

Do not use DIO (p. 176) on digital output lines for which the trigger output is enabled with TRO.

### **TRO? (Get Trigger Output State)**

Description: Returns if the trigger output configuration made with CTO (p. 169) is enabled or disabled for the given digital output line.

If all arguments are omitted, gets state of all digital output lines.

Format: TRO? [{<TrigOutID>}]



Arguments: <TrigOutID> is one digital output line of the controller, see TRO (p. 250) for more information.

Response: {<TrigOutID>="<TrigMode> LF}

where

<TrigMode> is the current state of the digital output line:

0 = Trigger output disabled

1 = Trigger output enabled

Troubleshooting: Illegal identifier of the digital output line

### TRS? (Indicate Reference Switch)

Description: Indicates whether axes have a reference point switch with direction sensing.

Format: TRS? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

Response: {<AxisID>="<uint> LF}

where

<uint> indicates whether the axis has a direction-sensing reference point switch (=1) or not (=0).

Troubleshooting: Illegal axis identifier

Notes: The E-871 firmware detects the presence or absence of a reference point switch via a parameter (ID 0x14). According to the value of this parameter, the E-871 enables or disables reference moves to the reference point switch (FRF command (p. 190)). Adapt the parameter value to your hardware using SPA (p. 238) or SEP (p. 234). Further information, see "Reference Point Switch Detection" (p. 30).

You can use a digital input line instead of the reference point switch as source of the reference point signal for the

FRF command. Further information see "Digital Input Signals" (p. 111).

#### **TVI? (Tell Valid Character Set For Axis Identifiers)**

Description: Returns a string with characters which can be used for axis identifiers.

Use SAI (p. 232) to change the axis identifiers and SAI? (p. 233) to ask for the current valid axis identifiers.

Format: TVI?

Arguments: None

Response: <string> is a list of characters

Notes: With the E-871, the string consists of  
1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ\_

#### **VAR (Set Variable Value)**

Description: Sets a variable to a certain value.

Local variables can be set using VAR in macros only. See "Variables" (p. 153) for details regarding local and global variables.

The variable is present in RAM only.

Format: VAR <Variable> <String>

Arguments: <Variable> is the name of the variable whose value is to be set.

<String> is the value to which the variable is to be set. If omitted, the variable is deleted.

The value can be given directly or via the value of a variable.

See “Variables” (p. 153) for conventions regarding variable names and values.

Response: None

Example: It is possible to set the value of one variable (e.g. TARGET) to that of another variable (e.g. SOURCE):

```
VAR TARGET ${SOURCE}
```

Use curly brackets if the name of the variable is longer than one character:

```
VAR A ONE
```

```
VAR VARB TWO
```

```
VAR $A 1
```

```
VAR ${VARB} 2
```

```
VAR $VARB 2 // this will result in an unwanted behavior
```

```
VAR?
```

```
A=ONE
```

```
VARB=TWO
```

```
ONE=1
```

```
TWO=2 // ${VARB}: is replaced by its value “TWO”.
```

```
ARB=2 // $VARB: $V is replaced by its (empty) value.
```

See ADD (p. 163) for another example.

**VAR? (Get Variable Values)**

Description: Gets values of variables.

If VAR? is combined with CPY (p. 167), JRC (p. 213), MEX (p. 221) or WAC (p. 255), the response to VAR? has to be a single value and not more.

More information regarding local and global variables can be found in "Variables" (p. 153).

Format: VAR? [{<Variable>}]

Arguments: <Variable> is the name of the variable to be queried. More information on name conventions can be found in "Variables" (p. 153).

If <Variable> is omitted, all global variables present in the RAM are listed.

Response: {<Variable>="<String>LF}

where

<String> gives the value to which the variable is set.

Notes: Local variables can be queried using VAR? only when a macro with local variables is running. See "Variables" (p. 153) for details regarding local and global variables.

Example: See ADD (p. 163) for an example.

**VER? (Get Versions Of Firmware And Drivers)**

Description: Gets the versions of the firmware of the E-871 as well as of further components like, for example, drivers and libraries.

Format: VER?

Arguments: None

Response {<string1>:"<string2> [<string3>]LF}

where

<string1> is the name of the component;  
 <string2> is the version information of the component  
 <string1>;  
 <string3> is an optional note.

### WAC (Wait For Condition)

Description: Waits until a given condition of the following type occurs: a specified value is compared with a queried value according a specified rule.

Can only be used in macros.

See also the MEX command (p. 221).

Format: WAC <CMD?> <OP> <value>

Arguments <CMD?> is one query command in its usual notation. The response has to be a single value and not more. For an example see below.

<OP> is the operator to be used. The following operators are possible:

= <= < > >= !=

Important: There must be a blank space before and after the operator!

<value> is the value to be compared with the response to <CMD?>.

Response: None

Example:

Send:

```
MAC BEG LPMOTION
MVR 1 1
WAC ONT? 1 = 1
MVR 1 -1
WAC ONT? 1 = 1
MAC START LPMOTION
MAC END
MAC START LPMOTION
```

Note: Macro LPMOTION is first recorded and then started. WAC ONT? 1 = 1 waits until the response to ONT? 1 is 1=1. To form an infinite loop, the macro calls itself.

### WPA (Save Parameters To Nonvolatile Memory)

Description:

Writes the currently valid value of a parameter of a given item from volatile memory (RAM) to nonvolatile memory. The values saved this way become the default values.

**Note: If the current parameter values are incorrect, this can cause a fault in the system. Make sure that the parameter settings are correct before you execute the WPA command.**

RAM settings not saved with WPA will be lost when the controller is switched off or rebooted or when RPA (p. 230) is used to restore the parameters.

With HPA? (p. 211) you can obtain a list of all available parameters.

Use SPA? (p. 238) to check the current parameter settings in volatile memory.

See SPA (p. 238) for an example.

Format:

WPA <Pswd> [{<ItemID> <PamID>}]

Arguments:	<p>&lt;Pswd&gt; is the password for writing to the nonvolatile memory. See below for details.</p> <p>&lt;ItemID&gt; is the item for which a parameter is to be saved from the volatile to the nonvolatile memory. See below for details.</p> <p>&lt;PamID&gt; is the parameter ID, can be written in hexadecimal or decimal format. See below for details.</p>		
Response:	None		
Troubleshooting:	<p>Illegal item identifier, wrong parameter ID, invalid password</p> <p><b>Note that the number of write cycles in the nonvolatile memory is limited. Write default settings only if necessary.</b></p>		
Notes:	<p>Parameters can be changed in the volatile memory with SPA (p. 238). Some parameters are loaded to the volatile memory of the E-871 from the ID chip (p. 41) of the connected stage when the E-871 is switched on or rebooted. When you use the PC software from PI, further information is loaded as parameter values from a stage database (p. 44) into the volatile memory of the E-871.</p> <p>WPA can also save parameter-independent settings that are set with the following commands:</p> <p>HDT (p. 195), assigns a lookup table to the axis of an HID device</p> <p>HIA (p. 197), configures the HID control</p> <p>HIT (p. 206), fills lookup tables with values</p> <p>The used password determines what is saved with WPA:</p>		
Valid passwords for writing in the nonvolatile memory:	100	Saves the currently valid values of all parameters and the currently valid settings for HDT, HIA and HIT	
	101	Saves the currently valid values of all parameters	
	HID	Saves the currently valid settings for HDT, HIA and HIT	

Available item IDs and parameter IDs:	It is not possible to specifically select individual items and parameters for saving with the E-871, i. e. <ItemID> and <PamID> are ignored.
---	--

## 9.7 Error Codes

The error codes listed here are those of the PI General Command Set. As such, some may be not relevant to your controller and will simply never occur.

### Controller errors

0	PI_CNTR_NO_ERROR	No error
1	PI_CNTR_PARAM_SYNTAX	Parameter syntax error
2	PI_CNTR_UNKNOWN_COMMAND	Unknown command
3	PI_CNTR_COMMAND_TOO_LONG	Command length out of limits or command buffer overrun
4	PI_CNTR_SCAN_ERROR	Error while scanning
5	PI_CNTR_MOVE_WITHOUT_REF_OR_NO_SERVO	Unallowable move attempted on unreferenced axis, or move attempted with servo off
6	PI_CNTR_INVALID_SGA_PARAM	Parameter for SGA not valid
7	PI_CNTR_POS_OUT_OF_LIMITS	Position out of limits
8	PI_CNTR_VEL_OUT_OF_LIMITS	Velocity out of limits
9	PI_CNTR_SET_PIVOT_NOT_POSSIBLE	Attempt to set pivot point while U,V and W not all 0
10	PI_CNTR_STOP	Controller was stopped by command
11	PI_CNTR_SST_OR_SCAN_RANGE	Parameter for SST or for one of the embedded scan algorithms out of range
12	PI_CNTR_INVALID_SCAN_AXES	Invalid axis combination for fast scan
13	PI_CNTR_INVALID_NAV_PARAM	Parameter for NAV out of range
14	PI_CNTR_INVALID_ANALOG_INPUT	Invalid analog channel
15	PI_CNTR_INVALID_AXIS_IDENTIFIER	Invalid axis identifier
16	PI_CNTR_INVALID_STAGE_NAME	Unknown stage name
17	PI_CNTR_PARAM_OUT_OF_RANGE	Parameter out of range
18	PI_CNTR_INVALID_MACRO_NAME	Invalid macro name
19	PI_CNTR_MACRO_RECORD	Error while recording macro
20	PI_CNTR_MACRO_NOT_FOUND	Macro not found
21	PI_CNTR_AXIS_HAS_NO_BRAKE	Axis has no brake
22	PI_CNTR_DOUBLE_AXIS	Axis identifier specified more than once



23	PI_CNTR_ILLEGAL_AXIS	Illegal axis
24	PI_CNTR_PARAM_NR	Incorrect number of parameters
25	PI_CNTR_INVALID_REAL_NR	Invalid floating point number
26	PI_CNTR_MISSING_PARAM	Parameter missing
27	PI_CNTR_SOFT_LIMIT_OUT_OF_RANGE	Soft limit out of range
28	PI_CNTR_NO_MANUAL_PAD	No manual pad found
29	PI_CNTR_NO_JUMP	No more step-response values
30	PI_CNTR_INVALID_JUMP	No step-response values recorded
31	PI_CNTR_AXIS_HAS_NO_REFERENCE	Axis has no reference sensor
32	PI_CNTR_STAGE_HAS_NO_LIM_SWITCH	Axis has no limit switch
33	PI_CNTR_NO_RELAY_CARD	No relay card installed
34	PI_CNTR_CMD_NOT_ALLOWED_FOR_STAGE	Command not allowed for selected stage(s)
35	PI_CNTR_NO_DIGITAL_INPUT	No digital input installed
36	PI_CNTR_NO_DIGITAL_OUTPUT	No digital output configured
37	PI_CNTR_NO_MCM	No more MCM responses
38	PI_CNTR_INVALID_MCM	No MCM values recorded
39	PI_CNTR_INVALID_CNTR_NUMBER	Controller number invalid
40	PI_CNTR_NO_JOYSTICK_CONNECTED	No joystick configured
41	PI_CNTR_INVALID_EGE_AXIS	Invalid axis for electronic gearing, axis cannot be slave
42	PI_CNTR_SLAVE_POSITION_OUT_OF_RANGE	Position of slave axis is out of range
43	PI_CNTR_COMMAND_EGE_SLAVE	Slave axis cannot be commanded directly when electronic gearing is enabled
44	PI_CNTR_JOYSTICK_CALIBRATION_FAILED	Calibration of joystick failed
45	PI_CNTR_REFERENCING_FAILED	Referencing failed
46	PI_CNTR_OPM_MISSING	OPM (Optical Power Meter) missing
47	PI_CNTR_OPM_NOT_INITIALIZED	OPM (Optical Power Meter) not initialized or cannot be initialized
48	PI_CNTR_OPM_COM_ERROR	OPM (Optical Power Meter) communication error
49	PI_CNTR_MOVE_TO_LIMIT_SWITCH_FAILED	Move to limit switch failed
50	PI_CNTR_REF_WITH_REF_DISABLED	Attempt to reference axis with referencing disabled
51	PI_CNTR_AXIS_UNDER_JOYSTICK_CONTROL	Selected axis is controlled by joystick
52	PI_CNTR_COMMUNICATION_ERROR	Controller detected communication error
53	PI_CNTR_DYNAMIC_MOVE_IN_PROCESS	MOV! motion still in progress
54	PI_CNTR_UNKNOWN_PARAMETER	Unknown parameter
55	PI_CNTR_NO_REP_RECORDED	No commands were recorded with REP

56	PI_CNTR_INVALID_PASSWORD	Password invalid
57	PI_CNTR_INVALID_RECORDER_CHAN	Data record table does not exist
58	PI_CNTR_INVALID_RECORDER_SRC_OPT	Source does not exist; number too low or too high
59	PI_CNTR_INVALID_RECORDER_SRC_CHAN	Source record table number too low or too high
60	PI_CNTR_PARAM_PROTECTION	Protected Param: Current Command Level (CCL) too low
61	PI_CNTR_AUTOZERO_RUNNING	Command execution not possible while autozero is running
62	PI_CNTR_NO_LINEAR_AXIS	Autozero requires at least one linear axis
63	PI_CNTR_INIT_RUNNING	Initialization still in progress
64	PI_CNTR_READ_ONLY_PARAMETER	Parameter is read-only
65	PI_CNTR_PAM_NOT_FOUND	Parameter not found in nonvolatile memory
66	PI_CNTR_VOL_OUT_OF_LIMITS	Voltage out of limits
67	PI_CNTR_WAVE_TOO_LARGE	Not enough memory available for requested wave curve
68	PI_CNTR_NOT_ENOUGH_DDL_MEMORY	Not enough memory available for DDL table; DDL cannot be started
69	PI_CNTR_DDL_TIME_DELAY_TOO_LARGE	Time delay larger than DDL table; DDL cannot be started
70	PI_CNTR_DIFFERENT_ARRAY_LENGTH	The requested arrays have different lengths; query them separately
71	PI_CNTR_GEN_SINGLE_MODE_RESTART	Attempt to restart the generator while it is running in single step mode
72	PI_CNTR_ANALOG_TARGET_ACTIVE	Motion commands and wave generator activation are not allowed when analog target is active
73	PI_CNTR_WAVE_GENERATOR_ACTIVE	Motion commands are not allowed when wave generator is active
74	PI_CNTR_AUTOZERO_DISABLED	No sensor channel or no piezo channel connected to selected axis (sensor and piezo matrix)
75	PI_CNTR_NO_WAVE_SELECTED	Generator started (WGO) without having selected a wave table (WSL).
76	PI_CNTR_IF_BUFFER_OVERRUN	Interface buffer overran and command couldn't be received correctly
77	PI_CNTR_NOT_ENOUGH_RECORDED_DATA	Data record table does not hold enough recorded data
78	PI_CNTR_TABLE_DEACTIVATED	Data record table is not configured for recording

79	PI_CNTR_OPENLOOP_VALUE_SET_WHEN_SERVO_ON	Open-loop commands (SVA, SVR) are not allowed when servo is on
80	PI_CNTR_RAM_ERROR	Hardware error affecting RAM
81	PI_CNTR_MACRO_UNKNOWN_COMMAND	Not macro command
82	PI_CNTR_MACRO_PC_ERROR	Macro counter out of range
83	PI_CNTR_JOYSTICK_ACTIVE	Joystick is active
84	PI_CNTR_MOTOR_IS_OFF	Motor is off
85	PI_CNTR_ONLY_IN_MACRO	Macro-only command
86	PI_CNTR_JOYSTICK_UNKNOWN_AXIS	Invalid joystick axis
87	PI_CNTR_JOYSTICK_UNKNOWN_ID	Joystick unknown
88	PI_CNTR_REF_MODE_IS_ON	Move without referenced stage
89	PI_CNTR_NOT_ALLOWED_IN_CURRENT_MOTION_MODE	Command not allowed in current motion mode
90	PI_CNTR_DIO_AND_TRACING_NOT_POSSIBLE	No tracing possible while digital IOs are used on this HW revision. Reconnect to switch operation mode.
91	PI_CNTR_COLLISION	Move not possible, would cause collision
92	PI_CNTR_SLAVE_NOT_FAST_ENOUGH	Stage is not capable of following the master. Check the gear ratio.
93	PI_CNTR_CMD_NOT_ALLOWED_WHILE_AXIS_IN_MOTION	This command is not allowed while the affected axis or its master is in motion.
94	PI_CNTR_OPEN_LOOP_JOYSTICK_ENABLED	Servo cannot be switched on when open-loop joystick control is enabled.
95	PI_CNTR_INVALID_SERVO_STATE_FOR_PARAMETER	This parameter cannot be changed in current servo mode.
96	PI_CNTR_UNKNOWN_STAGE_NAME	Unknown stage name
100	PI_LABVIEW_ERROR	PI LabVIEW driver reports error. See source control for details.
200	PI_CNTR_NO_AXIS	No stage connected to axis
201	PI_CNTR_NO_AXIS_PARAM_FILE	File with axis parameters not found
202	PI_CNTR_INVALID_AXIS_PARAM_FILE	Invalid axis parameter file
203	PI_CNTR_NO_AXIS_PARAM_BACKUP	Backup file with axis parameters not found
204	PI_CNTR_RESERVED_204	PI internal error code 204
205	PI_CNTR_SMO_WITH_SERVO_ON	SMO with servo on
206	PI_CNTR_UUDECODE_INCOMPLETE_HEADER	uudecode: incomplete header
207	PI_CNTR_UUDECODE_NOTHING_TO_DECODE	uudecode: nothing to decode
208	PI_CNTR_UUDECODE_ILLEGAL_FORMAT	uudecode: illegal UUE format
209	PI_CNTR_CRC32_ERROR	CRC32 error
210	PI_CNTR_ILLEGAL_FILENAME	Illegal file name (must be 8-0 format)

211	PI_CNTR_FILE_NOT_FOUND	File not found on controller
212	PI_CNTR_FILE_WRITE_ERROR	Error writing file on controller
213	PI_CNTR_DTR_HINDERS_VELOCITY_CHANGE	VEL command not allowed in DTR command mode
214	PI_CNTR_POSITION_UNKNOWN	Position calculations failed
215	PI_CNTR_CONN_POSSIBLY_BROKEN	The connection between controller and stage may be broken
216	PI_CNTR_ON_LIMIT_SWITCH	The connected stage has driven into a limit switch, some controllers need CLR to resume operation
217	PI_CNTR_UNEXPECTED_STRUT_STOP	Strut test command failed because of an unexpected strut stop
218	PI_CNTR_POSITION_BASED_ON_ESTIMATION	While MOV! is running position can only be estimated!
219	PI_CNTR_POSITION_BASED_ON_INTERPOLATION	Position was calculated during MOV motion
230	PI_CNTR_INVALID_HANDLE	Invalid handle
231	PI_CNTR_NO_BIOS_FOUND	No bios found
232	PI_CNTR_SAVE_SYS_CFG_FAILED	Save system configuration failed
233	PI_CNTR_LOAD_SYS_CFG_FAILED	Load system configuration failed
301	PI_CNTR_SEND_BUFFER_OVERFLOW	Send buffer overflow
302	PI_CNTR_VOLTAGE_OUT_OF_LIMITS	Voltage out of limits
303	PI_CNTR_OPEN_LOOP_MOTION_SET_WHEN_SERVO_ON	Open-loop motion attempted when servo ON
304	PI_CNTR_RECEIVING_BUFFER_OVERFLOW	Received command is too long
305	PI_CNTR_EEPROM_ERROR	Error while reading/writing EEPROM
306	PI_CNTR_I2C_ERROR	Error on I2C bus
307	PI_CNTR_RECEIVING_TIMEOUT	Timeout while receiving command
308	PI_CNTR_TIMEOUT	A lengthy operation has not finished in the expected time
309	PI_CNTR_MACRO_OUT_OF_SPACE	Insufficient space to store macro
310	PI_CNTR_EUI_OLDVERSION_CFGDATA	Configuration data has old version number
311	PI_CNTR_EUI_INVALID_CFGDATA	Invalid configuration data
333	PI_CNTR_HARDWARE_ERROR	Internal hardware error
400	PI_CNTR_WAV_INDEX_ERROR	Wave generator index error
401	PI_CNTR_WAV_NOT_DEFINED	Wave table not defined
402	PI_CNTR_WAV_TYPE_NOT_SUPPORTED	Wave type not supported
403	PI_CNTR_WAV_LENGTH_EXCEEDS_LIMIT	Wave length exceeds limit
404	PI_CNTR_WAV_PARAMETER_NR	Wave parameter number error
405	PI_CNTR_WAV_PARAMETER_OUT_OF_LIMIT	Wave parameter out of range

406	PI_CNTR_WGO_BIT_NOT_SUPPORTED	WGO command bit not supported
500	PI_CNTR_EMERGENCY_STOP_BUTTON_ACTIVATED	The \"red knob\" is still set and disables system
501	PI_CNTR_EMERGENCY_STOP_BUTTON_WAS_ACTIVATED	The \"red knob\" was activated and still disables system - reanimation required
502	PI_CNTR_REDUNDANCY_LIMIT_EXCEEDED	Position consistency check failed
503	PI_CNTR_COLLISION_SWITCH_ACTIVATED	Hardware collision sensor(s) are activated
504	PI_CNTR_FOLLOWING_ERROR	Strut following error occurred, e.g. caused by overload or encoder failure
505	PI_CNTR_SENSOR_SIGNAL_INVALID	One sensor signal is not valid
506	PI_CNTR_SERVO_LOOP_UNSTABLE	Servo loop was unstable due to wrong parameter setting and switched off to avoid damage.
530	PI_CNTR_NODE_DOES_NOT_EXIST	A command refers to a node that does not exist
531	PI_CNTR_PARENT_NODE_DOES_NOT_EXIST	A command refers to a node that has no parent node
532	PI_CNTR_NODE_IN_USE	Attempt to delete a node that is in use
533	PI_CNTR_NODE_DEFINITION_IS_CYCLIC	Definition of a node is cyclic
534	PI_CNTR_NODE_CHAIN_INVALID	The node chain does not end in the \"0\" node
535	PI_CNTR_NODE_DEFINITION_NOT_CONSISTENT	The definition of a coordinate transformation is erroneous
536	PI_CNTR_HEXAPOD_IN_MOTION	Transformation cannot be defined as long as Hexapod is in motion
537	PI_CNTR_TRANSFORMATION_TYPE_NOT_SUPPORTED	Transformation node cannot be activated
538	PI_CNTR_NODE_TYPE_DIFFERS	A node can only be replaced by a node of the same type
539	PI_CNTR_NODE_PARENT_IDENTICAL_TO_CHILD	A node cannot be linked to itself
540	PI_CNTR_NODE_DEFINITION_INCONSISTENT	Node definition is erroneous or not complete (replace or delete it)
541	PI_CNTR_ZERO_NODE_CANNOT_BE_CHANGED_OR_REPLACED	0 is the root node and cannot be modified
542	PI_CNTR_NODES_NOT_IN_SAME_CHAIN	The nodes are not part of the same chain
543	PI_CNTR_NODE_MEMORY_FULL	Unused nodes must be deleted before new nodes can be stored
544	PI_CNTR_PIVOT_POINT_FEATURE_NOT_SUPPORTED	With some transformations pivot point usage is not supported
555	PI_CNTR_UNKNOWN_ERROR	BasMac: unknown controller error
601	PI_CNTR_NOT_ENOUGH_MEMORY	Not enough memory

602	PI_CNTR_HW_VOLTAGE_ERROR	Hardware voltage error
603	PI_CNTR_HW_TEMPERATURE_ERROR	Hardware temperature out of range
604	PI_CNTR_POSITION_ERROR_TOO_HIGH	Position error of any axis in the system is too high
606	PI_CNTR_INPUT_OUT_OF_RANGE	Maximum value of input signal has been exceeded
1000	PI_CNTR_TOO_MANY_NESTED_MACROS	Too many nested macros
1001	PI_CNTR_MACRO_ALREADY_DEFINED	Macro already defined
1002	PI_CNTR_NO_MACRO_RECORDING	Macro recording not activated
1003	PI_CNTR_INVALID_MAC_PARAM	Invalid parameter for MAC
1004	PI_CNTR_RESERVED_1004	PI internal error code 1004
1005	PI_CNTR_CONTROLLER_BUSY	Controller is busy with some lengthy operation (e.g. reference move, fast scan algorithm)
1006	PI_CNTR_INVALID_IDENTIFIER	Invalid identifier (invalid special characters, ...)
1007	PI_CNTR_UNKNOWN_VARIABLE_OR_ARGUMENT	Variable or argument not defined
1008	PI_CNTR_RUNNING_MACRO	Controller is (already) running a macro
1009	PI_CNTR_MACRO_INVALID_OPERATOR	Invalid or missing operator for condition. Check necessary spaces around operator.
1010	PI_CNTR_MACRO_NO_ANSWER	No answer was received while executing WAC/MEX/JRC/...
1011	PI_CMD_NOT_VALID_IN_MACRO_MODE	Command not valid during macro execution
1024	PI_CNTR_MOTION_ERROR	Motion error: position error too large, servo is switched off automatically
1063	PI_CNTR_EXT_PROFILE_UNALLOWED_CMD	User profile mode: command is not allowed, check for required preparatory commands
1064	PI_CNTR_EXT_PROFILE_EXPECTING_MOTION_ERROR	User profile mode: first target position in user profile is too far from current position
1065	PI_CNTR_PROFILE_ACTIVE	Controller is (already) in user profile mode
1066	PI_CNTR_PROFILE_INDEX_OUT_OF_RANGE	User profile mode: block or data set index out of allowed range
1071	PI_CNTR_PROFILE_OUT_OF_MEMORY	User profile mode: out of memory
1072	PI_CNTR_PROFILE_WRONG_CLUSTER	User profile mode: cluster is not assigned to this axis
1073	PI_CNTR_PROFILE_UNKNOWN_CLUSTER_IDENTIFIER	Unknown cluster identifier
1090	PI_CNTR_TOO_MANY_TCP_CONNECTIONS_OPEN	There are too many open tcpip connections

2000	PI_CNTR_ALREADY_HAS_SERIAL_NUMBER	Controller already has a serial number
4000	PI_CNTR_SECTOR_ERASE_FAILED	Sector erase failed
4001	PI_CNTR_FLASH_PROGRAM_FAILED	Flash program failed
4002	PI_CNTR_FLASH_READ_FAILED	Flash read failed
4003	PI_CNTR_HW_MATCHCODE_ERROR	HW match code missing/invalid
4004	PI_CNTR_FW_MATCHCODE_ERROR	FW match code missing/invalid
4005	PI_CNTR_HW_VERSION_ERROR	HW version missing/invalid
4006	PI_CNTR_FW_VERSION_ERROR	FW version missing/invalid
4007	PI_CNTR_FW_UPDATE_ERROR	FW update failed
5000	PI_CNTR_INVALID_PCC_SCAN_DATA	PicoCompensation scan data is not valid
5001	PI_CNTR_PCC_SCAN_RUNNING	PicoCompensation is running, some actions cannot be executed during scanning/recording
5002	PI_CNTR_INVALID_PCC_AXIS	Given axis cannot be defined as PPC axis
5003	PI_CNTR_PCC_SCAN_OUT_OF_RANGE	Defined scan area is larger than the travel range
5004	PI_CNTR_PCC_TYPE_NOT_EXISTING	Given PicoCompensation type is not defined
5005	PI_CNTR_PCC_PAM_ERROR	PicoCompensation parameter error
5006	PI_CNTR_PCC_TABLE_ARRAY_TOO_LARGE	PicoCompensation table is larger than maximum table length
5100	PI_CNTR_NEXLINE_ERROR	Common error in NEXLINE® firmware module
5101	PI_CNTR_CHANNEL_ALREADY_USED	Output channel for NEXLINE® cannot be redefined for other usage
5102	PI_CNTR_NEXLINE_TABLE_TOO_SMALL	Memory for NEXLINE® signals is too small
5103	PI_CNTR_RNP_WITH_SERVO_ON	RNP cannot be executed if axis is in closed loop
5104	PI_CNTR_RNP_NEEDED	Relax procedure (RNP) needed
5200	PI_CNTR_AXIS_NOT_CONFIGURED	Axis must be configured for this action
6000	PI_CNTR_SENSOR_ABS_INVALID_VALUE	Invalid preset value of absolute sensor
6001	PI_CNTR_SENSOR_ABS_WRITE_ERROR	Error while writing to sensor
6002	PI_CNTR_SENSOR_ABS_READ_ERROR	Error while reading from sensor
6003	PI_CNTR_SENSOR_ABS_CRC_ERROR	Checksum error of absolute sensor
6004	PI_CNTR_SENSOR_ABS_ERROR	General error of absolute sensor
6005	PI_CNTR_SENSOR_ABS_OVERFLOW	Overflow of absolute sensor position

### Interface errors

0	COM_NO_ERROR	No error occurred during function call
-1	COM_ERROR	Error during com operation (could not be specified)
-2	SEND_ERROR	Error while sending data
-3	REC_ERROR	Error while receiving data
-4	NOT_CONNECTED_ERROR	Not connected (no port with given ID open)
-5	COM_BUFFER_OVERFLOW	Buffer overflow
-6	CONNECTION_FAILED	Error while opening port
-7	COM_TIMEOUT	Timeout error
-8	COM_MULTILINE_RESPONSE	There are more lines waiting in buffer
-9	COM_INVALID_ID	There is no interface or DLL handle with the given ID
-10	COM_NOTIFY_EVENT_ERROR	Event/message for notification could not be opened
-11	COM_NOT_IMPLEMENTED	Function not supported by this interface type
-12	COM_ECHO_ERROR	Error while sending "echoed" data
-13	COM_GPIB_EDVR	IEEE488: System error
-14	COM_GPIB_ECIC	IEEE488: Function requires GPIB board to be CIC
-15	COM_GPIB_ENOL	IEEE488: Write function detected no listeners
-16	COM_GPIB_EADR	IEEE488: Interface board not addressed correctly
-17	COM_GPIB_EARG	IEEE488: Invalid argument to function call
-18	COM_GPIB_ESAC	IEEE488: Function requires GPIB board to be SAC
-19	COM_GPIB_EABO	IEEE488: I/O operation aborted
-20	COM_GPIB_ENEB	IEEE488: Interface board not found
-21	COM_GPIB_EDMA	IEEE488: Error performing DMA
-22	COM_GPIB_EOIP	IEEE488: I/O operation started before previous operation completed
-23	COM_GPIB_ECAP	IEEE488: No capability for intended operation
-24	COM_GPIB_EFSO	IEEE488: File system operation error
-25	COM_GPIB_EBUS	IEEE488: Command error during device call
-26	COM_GPIB_ESTB	IEEE488: Serial poll-status byte lost
-27	COM_GPIB_ESRQ	IEEE488: SRQ remains asserted
-28	COM_GPIB_ETAB	IEEE488: Return buffer full
-29	COM_GPIB_ELCK	IEEE488: Address or board locked
-30	COM_RS_INVALID_DATA_BITS	RS-232: 5 data bits with 2 stop bits is an invalid combination, as is 6, 7, or 8 data bits with 1.5 stop bits
-31	COM_ERROR_RS_SETTINGS	RS-232: Error configuring the COM port
-32	COM_INTERNAL_RESOURCES_ERROR	Error dealing with internal system resources (events, threads, ...)
-33	COM_DLL_FUNC_ERROR	A DLL or one of the required functions could not be loaded



-34	COM_FTDIUSB_INVALID_HANDLE	FTDIUSB: invalid handle
-35	COM_FTDIUSB_DEVICE_NOT_FOUND	FTDIUSB: device not found
-36	COM_FTDIUSB_DEVICE_NOT_OPENED	FTDIUSB: device not opened
-37	COM_FTDIUSB_IO_ERROR	FTDIUSB: IO error
-38	COM_FTDIUSB_INSUFFICIENT_RESOURCES	FTDIUSB: insufficient resources
-39	COM_FTDIUSB_INVALID_PARAMETER	FTDIUSB: invalid parameter
-40	COM_FTDIUSB_INVALID_BAUD_RATE	FTDIUSB: invalid baud rate
-41	COM_FTDIUSB_DEVICE_NOT_OPENED_FOR_ERASE	FTDIUSB: device not opened for erase
-42	COM_FTDIUSB_DEVICE_NOT_OPENED_FOR_WRITE	FTDIUSB: device not opened for write
-43	COM_FTDIUSB_FAILED_TO_WRITE_DEVICE	FTDIUSB: failed to write device
-44	COM_FTDIUSB_EEPROM_READ_FAILED	FTDIUSB: EEPROM read failed
-45	COM_FTDIUSB_EEPROM_WRITE_FAILED	FTDIUSB: EEPROM write failed
-46	COM_FTDIUSB_EEPROM_ERASE_FAILED	FTDIUSB: EEPROM erase failed
-47	COM_FTDIUSB_EEPROM_NOT_PRESENT	FTDIUSB: EEPROM not present
-48	COM_FTDIUSB_EEPROM_NOT_PROGRAMMED	FTDIUSB: EEPROM not programmed
-49	COM_FTDIUSB_INVALID_ARGS	FTDIUSB: invalid arguments
-50	COM_FTDIUSB_NOT_SUPPORTED	FTDIUSB: not supported
-51	COM_FTDIUSB_OTHER_ERROR	FTDIUSB: other error
-52	COM_PORT_ALREADY_OPEN	Error while opening the COM port: was already open
-53	COM_PORT_CHECKSUM_ERROR	Checksum error in received data from COM port
-54	COM_SOCKET_NOT_READY	Socket not ready, you should call the function again
-55	COM_SOCKET_PORT_IN_USE	Port is used by another socket
-56	COM_SOCKET_NOT_CONNECTED	Socket not connected (or not valid)
-57	COM_SOCKET_TERMINATED	Connection terminated (by peer)
-58	COM_SOCKET_NO_RESPONSE	Can't connect to peer
-59	COM_SOCKET_INTERRUPTED	Operation was interrupted by a nonblocked signal
-60	COM_PCI_INVALID_ID	No device with this ID is present
-61	COM_PCI_ACCESS_DENIED	Driver could not be opened (on Vista: run as administrator!)

### DLL errors

-1001	PI_UNKNOWN_AXIS_IDENTIFIER	Unknown axis identifier
-1002	PI_NR_NAV_OUT_OF_RANGE	Number for NAV out of range--must be in [1,10000]
-1003	PI_INVALID_SGA	Invalid value for SGA--must be one of 1, 10, 100, 1000
-1004	PI_UNEXPECTED_RESPONSE	Controller sent unexpected response
-1005	PI_NO_MANUAL_PAD	No manual control pad installed, calls to SMA and related commands are not allowed
-1006	PI_INVALID_MANUAL_PAD_KNOB	Invalid number for manual control pad knob

-1007	PI_INVALID_MANUAL_PAD_AXIS	Axis not currently controlled by a manual control pad
-1008	PI_CONTROLLER_BUSY	Controller is busy with some lengthy operation (e.g. reference move, fast scan algorithm)
-1009	PI_THREAD_ERROR	Internal error--could not start thread
-1010	PI_IN_MACRO_MODE	Controller is (already) in macro mode--command not valid in macro mode
-1011	PI_NOT_IN_MACRO_MODE	Controller not in macro mode--command not valid unless macro mode active
-1012	PI_MACRO_FILE_ERROR	Could not open file to write or read macro
-1013	PI_NO_MACRO_OR_EMPTY	No macro with given name on controller, or macro is empty
-1014	PI_MACRO_EDITOR_ERROR	Internal error in macro editor
-1015	PI_INVALID_ARGUMENT	One or more arguments given to function is invalid (empty string, index out of range, ...)
-1016	PI_AXIS_ALREADY_EXISTS	Axis identifier is already in use by a connected stage
-1017	PI_INVALID_AXIS_IDENTIFIER	Invalid axis identifier
-1018	PI_COM_ARRAY_ERROR	Could not access array data in COM server
-1019	PI_COM_ARRAY_RANGE_ERROR	Range of array does not fit the number of parameters
-1020	PI_INVALID_SPA_CMD_ID	Invalid parameter ID given to SPA or SPA?
-1021	PI_NR_AVG_OUT_OF_RANGE	Number for AVG out of range--must be >0
-1022	PI_WAV_SAMPLES_OUT_OF_RANGE	Incorrect number of samples given to WAV
-1023	PI_WAV_FAILED	Generation of wave failed
-1024	PI_MOTION_ERROR	Motion error: position error too large, servo is switched off automatically
-1025	PI_RUNNING_MACRO	Controller is (already) running a macro
-1026	PI_PZT_CONFIG_FAILED	Configuration of PZT stage or amplifier failed
-1027	PI_PZT_CONFIG_INVALID_PARAMS	Current settings are not valid for desired configuration
-1028	PI_UNKNOWN_CHANNEL_IDENTIFIER	Unknown channel identifier
-1029	PI_WAVE_PARAM_FILE_ERROR	Error while reading/writing wave generator parameter file
-1030	PI_UNKNOWN_WAVE_SET	Could not find description of wave form. Maybe WG.INI is missing?
-1031	PI_WAVE_EDITOR_FUNC_NOT_LOADED	The WGWaveEditor DLL function was not found at startup
-1032	PI_USER_CANCELLED	The user cancelled a dialog
-1033	PI_C844_ERROR	Error from C-844 Controller
-1034	PI_DLL_NOT_LOADED	DLL necessary to call function not loaded, or function not found in DLL

-1035	PI_PARAMETER_FILE_PROTECTED	The open parameter file is protected and cannot be edited
-1036	PI_NO_PARAMETER_FILE_OPENED	There is no parameter file open
-1037	PI_STAGE_DOES_NOT_EXIST	Selected stage does not exist
-1038	PI_PARAMETER_FILE_ALREADY_OPENED	There is already a parameter file open. Close it before opening a new file
-1039	PI_PARAMETER_FILE_OPEN_ERROR	Could not open parameter file
-1040	PI_INVALID_CONTROLLER_VERSION	The version of the connected controller is invalid
-1041	PI_PARAM_SET_ERROR	Parameter could not be set with SPA--parameter not defined for this controller!
-1042	PI_NUMBER_OF_POSSIBLE_WAVES_EXCEEDED	The maximum number of wave definitions has been exceeded
-1043	PI_NUMBER_OF_POSSIBLE_GENERATORS_EXCEEDED	The maximum number of wave generators has been exceeded
-1044	PI_NO_WAVE_FOR_AXIS_DEFINED	No wave defined for specified axis
-1045	PI_CANT_STOP_OR_START_WAV	Wave output to axis already stopped/started
-1046	PI_REFERENCE_ERROR	Not all axes could be referenced
-1047	PI_REQUIRED_WAVE_NOT_FOUND	Could not find parameter set required by frequency relation
-1048	PI_INVALID_SPP_CMD_ID	Command ID given to SPP or SPP? is not valid
-1049	PI_STAGE_NAME_ISNT_UNIQUE	A stage name given to CST is not unique
-1050	PI_FILE_TRANSFER_BEGIN_MISSING	A uuencoded file transferred did not start with "begin" followed by the proper filename
-1051	PI_FILE_TRANSFER_ERROR_TEMP_FILE	Could not create/read file on host PC
-1052	PI_FILE_TRANSFER_CRC_ERROR	Checksum error when transferring a file to/from the controller
-1053	PI_COULDNT_FIND_PISTAGES_DAT	The PiStages.dat database could not be found. This file is required to connect a stage with the CST command
-1054	PI_NO_WAVE_RUNNING	No wave being output to specified axis
-1055	PI_INVALID_PASSWORD	Invalid password
-1056	PI_OPM_COM_ERROR	Error during communication with OPM (Optical Power Meter), maybe no OPM connected
-1057	PI_WAVE_EDITOR_WRONG_PARAMNUM	WaveEditor: Error during wave creation, incorrect number of parameters
-1058	PI_WAVE_EDITOR_FREQUENCY_OUT_OF_RANGE	WaveEditor: Frequency out of range
-1059	PI_WAVE_EDITOR_WRONG_IP_VALUE	WaveEditor: Error during wave creation, incorrect index for integer parameter
-1060	PI_WAVE_EDITOR_WRONG_DP_VALUE	WaveEditor: Error during wave creation, incorrect index for floating point parameter
-1061	PI_WAVE_EDITOR_WRONG_ITEM_VALUE	WaveEditor: Error during wave creation, could not calculate value

-1062	PI_WAVE_EDITOR_MISSING_GRAPH_COMPONENT	WaveEditor: Graph display component not installed
-1063	PI_EXT_PROFILE_UNALLOWED_CMD	User Profile Mode: Command is not allowed, check for required preparatory commands
-1064	PI_EXT_PROFILE_EXPECTING_MOTION_ERROR	User Profile Mode: First target position in User Profile is too far from current position
-1065	PI_EXT_PROFILE_ACTIVE	Controller is (already) in User Profile Mode
-1066	PI_EXT_PROFILE_INDEX_OUT_OF_RANGE	User Profile Mode: Block or Data Set index out of allowed range
-1067	PI_PROFILE_GENERATOR_NO_PROFILE	ProfileGenerator: No profile has been created yet
-1068	PI_PROFILE_GENERATOR_OUT_OF_LIMITS	ProfileGenerator: Generated profile exceeds limits of one or both axes
-1069	PI_PROFILE_GENERATOR_UNKNOWN_PARAMETER	ProfileGenerator: Unknown parameter ID in Set/Get Parameter command
-1070	PI_PROFILE_GENERATOR_PAR_OUT_OF_RANGE	ProfileGenerator: Parameter out of allowed range
-1071	PI_EXT_PROFILE_OUT_OF_MEMORY	User Profile Mode: Out of memory
-1072	PI_EXT_PROFILE_WRONG_CLUSTER	User Profile Mode: Cluster is not assigned to this axis
-1073	PI_UNKNOWN_CLUSTER_IDENTIFIER	Unknown cluster identifier
-1074	PI_INVALID_DEVICE_DRIVER_VERSION	The installed device driver doesn't match the required version. Please see the documentation to determine the required device driver version.
-1075	PI_INVALID_LIBRARY_VERSION	The library used doesn't match the required version. Please see the documentation to determine the required library version.
-1076	PI_INTERFACE_LOCKED	The interface is currently locked by another function. Please try again later.
-1077	PI_PARAM_DAT_FILE_INVALID_VERSION	Version of parameter DAT file does not match the required version. Current files are available at <a href="http://www.pi.ws">www.pi.ws</a> .
-1078	PI_CANNOT_WRITE_TO_PARAM_DAT_FILE	Cannot write to parameter DAT file to store user defined stage type.
-1079	PI_CANNOT_CREATE_PARAM_DAT_FILE	Cannot create parameter DAT file to store user defined stage type.
-1080	PI_PARAM_DAT_FILE_INVALID_REVISION	Parameter DAT file does not have correct revision.
-1081	PI_USERSTAGES_DAT_FILE_INVALID_REVISION	User stages DAT file does not have correct revision.
-1082	PI_SOFTWARE_TIMEOUT	Timeout Error. Some lengthy operation did not finish within expected time.

## 10 Adapting Settings

### In this Chapter

Changing Parameters in the E-871 .....	271
Creating or Modifying a Stage Type .....	277
Parameter Overview .....	280

### 10.1 Changing Parameters in the E-871

#### INFORMATION

The values in the nonvolatile memory are loaded to the volatile memory as default values when the E-871 is switched on or rebooted and take effect immediately.

#### INFORMATION

The number of write cycles in the nonvolatile memory is restricted by the limited lifetime of the memory chip (EEPROM).

- Overwrite the default values only when it is necessary.
- Save the current parameter values to the PC (p. 272) before you perform changes in the nonvolatile memory.
- Contact our customer service department (p. 301) if the E-871 exhibits unexpected behavior.

#### INFORMATION

If the connected stage contains an ID chip (p. 41), data will be loaded from the ID chip to the volatile memory of the E-871 when the E-871 is switched on or rebooted.

The ID chip only contains some of the information that is required to operate the stage with the E-871. When you use the PC software from PI, further information is loaded as parameter values from a stage database (p. 44) into the volatile memory of the E-871.

Parameters that are loaded from the ID chip or from a stage database are marked in color in the parameter overview (p. 280).

### 10.1.1 General Commands for Parameters

The following commands are available for changing parameters:

Comm and	Function
SPA	Change parameters in the volatile memory.
SEP	Change parameters in the nonvolatile memory.
WPA	Copy a current parameter value from the volatile to the nonvolatile memory. Here it is used as a default value.
RPA	Copy a parameter value from the nonvolatile to the volatile memory.
CCL	Change to a higher command level, e.g. to obtain write permission for particular parameters.
SPA?	Get parameter values from the volatile memory.
SEP?	Get parameter values from the nonvolatile memory.

You can find details in the command descriptions (p. 159).

You get easier access to the parameter values with the PIMikroMove PC software.

### 10.1.2 Saving Parameter Values in a Text File

#### INFORMATION

The E-871 is configured via parameters, e. g. for adaptation to the connected mechanical system. Changing parameter values can cause undesirable results.

- Create a backup copy on the PC before changing the parameter settings of the E-871. You can then restore the original settings at any time.
- Create an additional backup copy with a new filename each time after you optimize the parameter values or adapt the E-871 to a particular stage.

#### Prerequisites

- ✓ You have read and understood the General Notes on Start-Up (p. 71).
- ✓ You have established communication between the E-871 and the PC with PIMikroMove or PITerminal via the RS-232 interface (p. 76) or the USB interface (p. 77).

### Saving parameter values in a text file

1. If you use PIMikroMove, open the window for sending commands:
  - In the main window select the **Tools > Command entry** menu item or press the **F4** key on the keyboard.

In PITerminal the main window from which commands can be sent is opened automatically after establishing communication.

2. Get the parameter values from which you want to create a backup copy.
  - If you want to save the parameter values from the volatile memory of the E-871: Send the **SPA?** command.
  - If you want to save the parameter values from the nonvolatile memory of the E-871: Send the **SEP?** command.

3. Click on the **Save...** button.

The **Save content of terminal as textfile** window opens.

4. In the **Save content of terminal as textfile** window, save the queried parameter values in a text file on your PC.

### 10.1.3 Changing Parameters: General Procedure

#### NOTICE



#### Improper parameter settings!

The values in the nonvolatile memory are loaded to the volatile memory as default values when switching on or rebooting the E-871 and take effect immediately.

Improper parameter settings can cause damage to the stage connected.

- Only change parameters after careful consideration.
- Save the current parameter values to the PC (p. 272) before you perform changes in the nonvolatile memory.

**INFORMATION**

The write access for the parameters of the E-871 is defined by command levels. After the controller is switched on or rebooted, the active command level is always level 0. For particular parameters, write access is only allowed on command level 1. On command levels > 1, write access is only available to PI service personnel.

The E-871 ignores the active command level in the following cases:

- The E-871 reads parameter values from the ID chip of the stage.
  - The stage type is selected in the PC software.
  - The current parameter values are written from the volatile to the nonvolatile memory (directly with WPA or in the PC software).
- If necessary, send the `CCL 1 advanced` command or enter the password `advanced` to change to command level 1.
  - Contact the customer service department if there seem to be problems with parameters of command level 2 or higher (p. 301).

**Available parameters**

The parameters available for adapting the E-871 to your application depend on the firmware of your E-871.

- Send the `HPA?` command (p. 211) to obtain a list of all available parameters with a short description.

**Parameters in the nonvolatile memory**

- Send the `SEP?` command (p. 235) to obtain a list of parameter values in the nonvolatile memory.

**INFORMATION**

The parameter values in the nonvolatile memory are automatically loaded to the volatile memory when switching on or rebooting the E-871.

- Change the parameters in the nonvolatile memory only if you are sure that the E-871 functions correctly with the parameter values.

- Change the parameters in the nonvolatile memory with the `SEP` command (p. 234).



### Parameters in the volatile memory

- Send the `SPA?` command (p. 241) to obtain a list of parameter values in the volatile memory.
- Change the parameters in the volatile memory with the `SPA` command (p. 238).

If you are working with PIMikroMove:

1. In the main window of PIMikroMove, open the single axis window for the connected stage by selecting the stage in the **View > Single Axis Window** menu.
2. Expand the view of the single axis window by clicking on the > button at the right edge of the window.
3. If the parameter to be modified is not included in the list on the right side of the window, click on **Select parameters...** and add it to the list.
4. Type the new parameter value into the appropriate input field in the list.
5. Press the `Enter` key of the PC to transfer the parameter value to the volatile memory of the controller. When this is done, the entry changes its color from blue to black.

### Writing parameters from the volatile memory to the nonvolatile memory

#### INFORMATION

To save parameter values in the nonvolatile memory, it is necessary to enter a password. Usable passwords:

- |     |   |
|-----|---|
| 100 | Saves the currently valid values of all parameters and the currently valid settings for HDT, HIA and HIT<br>Use with the WPA and SEP commands |
| 101 | Saves the currently valid values of all parameters<br>Use with the WPA command  |

1. Change the parameters in the volatile memory with the `SPA` command (p. 238).
2. Check whether the E-871 is operating correctly with the modified parameters.

- If yes:  
Write the modified parameter values to the nonvolatile memory with the `WPA` command (p. 256).
- If no:  
Change and check the parameters in the volatile memory again.

If you are working with PIMikroMove:

- Write the current values of all parameters to the nonvolatile memory:
  - a) In the main window of PIMikroMove, select the **E-871 > Save parameters to non-volatile memory** menu item. The **Save Parameters to Non-Volatile Memory** dialog opens.
  - b) In the selection field of the **Save Parameters to Non-Volatile Memory** dialog, either enter the password *101* or select the *all parameters (101)* entry. If you also want to write the currently valid settings for HDT, HIA and HIT to the nonvolatile memory of the E-871, either enter the password *100* or select the *all parameters, settings of HDT, HIA, HIT (100)* entry.
  - c) Click **OK** to save and close the dialog.

### Writing parameters from the nonvolatile memory to the volatile memory

#### INFORMATION

- Use this procedure only if you are sure that the E-871 functions correctly with the parameter values from the nonvolatile memory.
- 
- Write the parameter values from the nonvolatile memory to the volatile memory with the `RPA` command (p. 230).

## 10.2 Creating or Modifying a Stage Type

You can select a parameter set appropriate for your stage from a stage database in the PC software from PI. The software transfers the values of the selected parameter set to the volatile memory of the controller. Further information see "Stage Databases" (p. 44).

The PI\_UserStages2.dat stage database is intended for creating, editing and storing new parameter sets. This can be required in the following cases, for example:

- You want to operate a stage with different notch filter settings and/or servo-control parameters than those of the standard stage database (PIMicosStages2.dat or PISTages2.dat).
- You want to adapt the soft limits of the stage to your application.
- You have a custom stage.

### INFORMATION

You can create a new stage type with utmost ease by modifying in PIMikroMove an existing stage type and saving it under a new name.

### INFORMATION

If a stage type with the same name is present in the standard stage database (PIStages2.dat or PIMicosStages2.dat) and in the PI\_UserStages2.dat database, the parameter settings from the standard database will always be loaded when this stage type is selected in the PC software. The parameter settings from PI\_UserStages2.dat are not used in this case.

- When saving stage types, only assign names that are **not** already used in the PISTages2.dat or PIMicosStages2.dat stage database.

In the following, PIMikroMove is used for creating a new stage type and for modifying an existing stage type.

### Prerequisite

- ✓ PIMikroMove is installed on the PC (p. 55).
- ✓ You have read and understood the PIMikroMove manual. The manual is found on the product CD.
- ✓ You have installed the latest versions of the PIMicosStages2.dat and PISTages2.dat stage databases on your PC (p. 57).

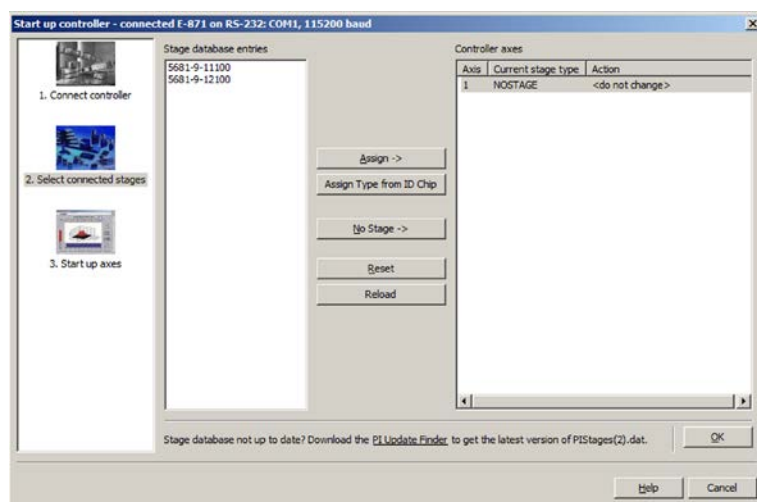
- ✓ If PI has provided you with a custom stage database for your stage, then you have installed this database on your PC (p. 59).
- ✓ You have established communication between the E-871 and the PC with PIMikroMove (p. 76).

### Creating a stage type in a stage database

1. In the **Start up controller** window in PIMikroMove, select the stage type that is to be used as the basis for the stage type to be created:

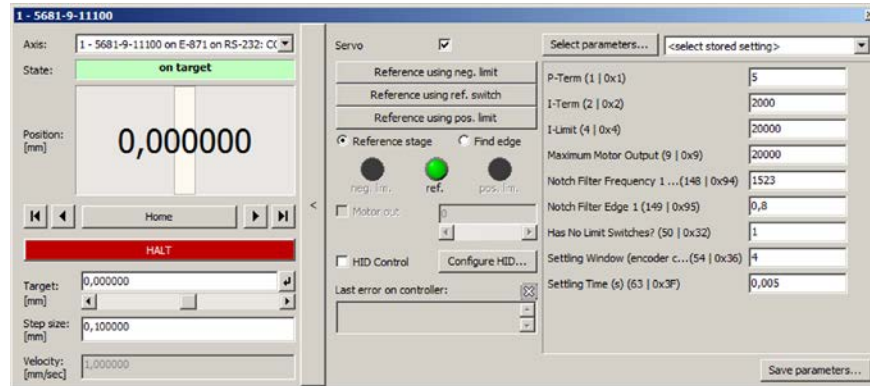
If the **Select connected stages** step is not displayed in the **Start up controller** window:

- In the left part of the window, click **Select connected stages**.



- a) Select the appropriate stage type: Click **Assign Type from ID Chip** or mark the appropriate stage type in the **Stage database entries** list.
  - b) When you have marked the appropriate stage type in the **Stage database entries** list in step a, click **Assign**.
  - c) Confirm the selection with **OK** to load the parameter settings for the selected stage type from the stage database to the volatile memory of the E-871. The **Start up controller** window goes to the **Start up axes** step.
2. In the **Start up axes** step, click on **Close** to close the **Start up controller** window.
  3. In the main window of PIMikroMove, open the single axis window for the stage type selected by selecting the stage type in the **View > Single Axis Window** menu.

4. Expand the view of the single axis window by clicking on the > button at the right edge of the window.



5. Enter new values for the stage parameters:
  - a) If the parameter to be modified is not included in the list on the right side of the window, click on **Select parameters...** and add it to the list.
  - b) Type the new parameter value into the appropriate input field in the list.
  - c) Press the **Enter** key of the PC to transfer the parameter value to the volatile memory of the controller. When this is done, the entry changes its color from blue to black.
6. Click the **Save parameters...** button underneath the parameter list.

The **Save Parameters as User Stage Type** dialog opens.

7. In the **Save Parameters as User Stage Type** dialog, save the modified parameter values as a new stage type:
  - a) Leave the entry in the **Parameters of axis:** field unchanged.
  - b) Enter the name for the new stage type in the **Save as:** field.
  - c) Click on **OK**.

The new stage type has been saved in the PI\_UserStages2.dat stage database. The display of the connected stage type has been updated in the single axis window and in the main window of PIMikroMove. The new stage type can be used immediately to configure the E-871 for the connected stage (e. g. by selection in the **Select connected stages** step).

## Modifying a stage type in a stage database

1. In the **Select connected stages** step in PIMikroMove, select a stage type which you created earlier as described above. Proceed with the selection as in step 1 of the **Creating a stage type in a stage database** instruction.
2. Execute steps 2 to 6 from **Creating a stage type in a stage database**.

3. In the **Save Parameters as User Stage Type** dialog, save the modified parameter values of the stage type:
  - a) Leave the entry in the **Parameters of axis:** field unchanged.
  - b) Leave the entry in the **Save as:** field unchanged.
  - c) Click on **OK**.
  - d) In the **Stage type already defined** dialog, click on **Change settings**. The **Save Parameters as User Stage Type** dialog closes automatically after a short time.

The parameter values of the stage type have been updated in the PI\_UserStages2.dat stage database and in the main window of PIMikroMove.

## 10.3 Parameter Overview

### INFORMATION

The write access for the parameters of the E-871 is defined by command levels. After the controller is switched on or rebooted, the active command level is always level 0. For particular parameters, write access is only allowed on command level 1. On command levels > 1, write access is only available to PI service personnel.

The E-871 ignores the active command level in the following cases:

- The E-871 reads parameter values from the ID chip of the stage.
  - The stage type is selected in the PC software.
  - The current parameter values are written from the volatile to the nonvolatile memory (directly with WPA or in the PC software).
- If necessary, send the **CCL 1 advanced** command or enter the password **advanced** to change to command level 1.
  - Contact the customer service department if there seem to be problems with parameters of command level 2 or higher (p. 301).

**INFORMATION**

To save parameter values in the nonvolatile memory, it is necessary to enter a password. Usable passwords:

- 100 Saves the currently valid values of all parameters and the currently valid settings for HDT, HIA and HIT  
Use with the WPA and SEP commands
- 101 Saves the currently valid values of all parameters  
Use with the WPA command

Meaning of the color emphasis in the parameter table:

Dark gray:	The value of the parameter is loaded from the ID chip of the stage (p. 41).
Light gray:	The value of the parameter can be loaded from a stage database (p. 44).
Colorless:	The value of the parameter can only be changed via command (SPA, SEP) or by using corresponding operating elements of the PC software (p. 273).

Parameter ID (hexa-decimal)	Data type	Command level for write access	Parameter name	Description
0x1	INT	0	P-Term	0 to 32767; details see "Servo Algorithm and Other Control Value Corrections" (p. 26).
0x2	INT	0	I-Term	0 to 32767; details see "Servo Algorithm and Other Control Value Corrections" (p. 26).
0x3	INT	0	D-Term	Details see "Servo Algorithm and Other Control Value Corrections" (p. 26).
0x4	INT	0	I-Limit	0 to 32767; details see "Servo Algorithm and Other Control Value Corrections" (p. 26).
0x5	INT	0	Kvff	Present for reason of compatibility only.

Parameter ID (hexa-decimal)	Data type	Command level for write access	Parameter name	Description
0x8	FLOAT	0	Maximum Position Error (Phys. Unit)	Maximum position error Present for reason of compatibility only.
0x9	INT	0	Maximum Motor Output	Maximum control value for driving the axis (dimensionless) 0 to 32767 Details see "Block diagram" (p. 15), SMO (p. 236) and SMO? (p. 238).
0xE	INT	0	Numerator Of The Counts-Per-Physical-Unit Factor	Numerator and denominator of the factor for counts per physical length unit
0xF	INT	0	Denominator Of The Counts-Per-Physical-Unit Factor	Details see "Physical Units" (p. 22).
0x13	INT	0	Is Rotary Stage?	Is this a rotary stage? 0 = Not a rotary stage 1 = Rotary stage No evaluation by the E-871, but only by the PC software: PIMikroMove determines which motions are permissible on the basis of this value.
0x14	INT	0	Has Reference?	Does the stage have a reference point switch? Details see "Reference Point Switch Detection" (p. 30).
0x15	FLOAT	0	Maximum Travel In Positive Direction (Phys. Unit)	Soft limit in positive direction See examples in "Travel Range and Soft Limits" (p. 34).
0x16	FLOAT	0	Value At Reference Position (Phys. Unit)	Position value on the reference point switch See examples in "Travel Range and Soft Limits" (p. 34).



Parameter ID (hexa-decimal)	Data type	Command level for write access	Parameter name	Description
0x17	FLOAT	0	Distance From Negative Limit To Reference Position (Phys. Unit)	Distance between reference point switch and negative limit switch See examples in "Travel Range and Soft Limits" (p. 34).
0x18	INT	0	Limit Mode	Signal logic of the limit switches Details see "Limit Switch Detection" (p. 31).
0x1B	INT	0	Profile mode	Type of dynamics profile 5 = Without dynamics profile The E-871 does not have a profile generator.
0x2F	FLOAT	0	Distance From Reference Position To Positive Limit (Phys. Unit)	Distance between reference point switch and positive limit switch See examples in "Travel Range and Soft Limits" (p. 34).
0x30	FLOAT	0	Maximum Travel In Negative Direction (Phys. Unit)	Soft limit in negative direction See examples in "Travel Range and Soft Limits" (p. 34).
0x31	INT	0	Invert Reference?	Should the reference signal be inverted? Details see "Reference Point Switch Detection" (p. 30).
0x32	INT	0	Has No Limit Switches?	Does the stage have limit switches? Details see "Limit Switch Detection" (p. 31).
0x33	INT	0	Motor Offset Positive	Present for reason of compatibility only.
0x34	INT	0	Motor Offset Negative	Present for reason of compatibility only.
0x36	INT	0	Settling Window (encoder counts)	Settling window around the target position Details see "On-Target State" (p. 29).

Parameter ID (hexa-decimal)	Data type	Command level for write access	Parameter name	Description
0x3C	CHAR	0	Stage Name	<p>Stage name</p> <p>Maximum of 20 characters; default value: NOSTAGE</p> <p>The value NOSTAGE "deactivates" the axis. A deactivated axis is not accessible for axis-related commands (e.g. motion commands or position queries).</p> <p>The value of this parameter is used by PIMikroMove as a criterion for finding a suitable parameter set in the stage databases. Details see "Starting Motions" (p. 84).</p>
0x3F	FLOAT	0	Settling Time (s)	<p>Delay time for setting the on-target state.</p> <p>Details see "On-Target State" (p. 29).</p>
0x47	INT	0	Reference Travel Direction	<p>Default direction for the reference move</p> <p>Details see "Reference Point Definition" (p. 37).</p>
0x48	INT	0	Motor Drive Offset	Present for reason of compatibility only.
0x5A	INT	0	Numerator Of The Servo-Loop Input Factor	<p>Numerator and denominator of the servo loop input factor</p> <p>Details see "Servo Algorithm and Other Control Value Corrections" (p. 26).</p>
0x5B	INT	0	Denominator Of The Servo-Loop Input Factor	
0x5C	INT	0	Source Of Reference Signal	<p>Reference signal source for the <b>FRF</b> or <b>FED</b> commands</p> <p>Details see "Commands and Parameters for Digital Inputs" (p. 112) and "Using Digital Input Signals as Switch Signals" (p. 114).</p>

Parameter ID (hexa-decimal)	Data type	Command level for write access	Parameter name	Description
0x5D	INT	0	Source Of Negative Limit Signal	Reference signal source for the <b>FNL</b> or <b>FED</b> commands Details see "Commands and Parameters for Digital Inputs" (p. 112) and "Using Digital Input Signals as Switch Signals" (p. 114).
0x5E	INT	0	Source Of Positive Limit Signal	Reference signal source for the <b>FPL</b> or <b>FED</b> commands Details see "Commands and Parameters for Digital Inputs" (p. 112) and "Using Digital Input Signals as Switch Signals" (p. 114).
0x5F	INT	0	Invert Digital Input Used For Negative Limit	Inverts the polarity of the digital inputs that are used for the source of the negative limit switch signal. Details see "Commands and Parameters for Digital Inputs" (p. 112) and "Using Digital Input Signals as Switch Signals" (p. 114).
0x60	INT	0	Invert Digital Input Used For Positive Limit	Inverts the polarity of the digital inputs that are used for the source of the positive limit switch signal. Details see "Commands and Parameters for Digital Inputs" (p. 112) and "Using Digital Input Signals as Switch Signals" (p. 114).
0x61	INT	0	Invert Direction Of Motion For Joystick-Controlled Axis?	Should the direction of motion for HID-controlled axes be inverted? Details see "Commands and Parameters for HID Control" (p. 119).
0x63	FLOAT	0	Distance Between Limit And Hard Stop (Phys. Unit)	Distance between internal limit switch and hard stop Details see "Reference Point Definition" (p. 37).

Parameter ID (hexa-decimal)	Data type	Command level for write access	Parameter name	Description
0x70	INT	0	Reference Signal Type	Reference signal type Details see "Reference Point Switch Detection" (p. 30).
0x71	INT	0	D-Term Delay (No. Of Servo Cycles)	D-term delay Details see "Servo Algorithm and Other Control Value Corrections" (p. 26).
0x72	INT	0	Ignore Macro Error?	Ignore macro error? Details see "Commands and Parameters for Macros" (p. 132).
0x77	INT	0	Use Limit Switches Only For Reference Moves?	Should the limit switches only be used for reference moves? Details see "Limit Switch Detection" (p. 31).
0x78	FLOAT	0	Distance From Limit To Start Of Ref. Search (Phys. Unit)	Distance between the limit switch or hard stop and the starting position for the reference move to the index pulse Details see "Reference Point Definition" (p. 37).
0x79	FLOAT	0	Distance For Reference Search (Phys. Unit)	Maximum distance for the reference move to the index pulse Details see "Reference Point Definition" (p. 37).
0x7A	INT	0	Use Hard Stops For Referencing?	Should the hard stops be used for reference moves? Details see "Reference Point Definition" (p. 37).
0x94	FLOAT	0	Notch Filter Frequency 1 (Hz)	Frequency of the first notch filter Details see "Servo Algorithm and Other Control Value Corrections" (p. 26).

Parameter ID (hexa-decimal)	Data type	Command level for write access	Parameter name	Description
0x95	FLOAT	0	Notch Filter Edge 1	Rise of the edge of the first notch filter Details see "Servo Algorithm and Other Control Value Corrections" (p. 26).
0x03003300	FLOAT	2	Sensor Interpolation	Interpolation rate for the signals of the incremental sensor
0x03003301	FLOAT	2	Sensor Hysteresis	Correction of the hysteresis of the incremental sensor
0x03003302	FLOAT	2	Sensor Board Gain	Gain value for the correction of the digitized signals of the incremental sensor
0x03003303	FLOAT	2	Sensor Digital Offset 0	Offset 0 for the correction of the digitized signals of the incremental sensor
0x03003304	FLOAT	2	Sensor Digital Offset 1	Offset 1 for the correction of the digitized signals of the incremental sensor
0x03003305	FLOAT	2	Sensor Digital Phase (deg)	Phase correction for the signals of the incremental sensor
0x03003306	FLOAT	2	Sensor Analog Gain	Gain value for the correction of the analog signals of the incremental sensor
0x03003307	FLOAT	2	Sensor Analog Offset 0	Offset 0 for the correction of the analog signals of the incremental sensor
0x03003308	FLOAT	2	Sensor Analog Offset 1	Offset 1 for the correction of the analog signals of the incremental sensor
0x07000000	FLOAT	0	Range Limit Min	Additional soft limit for the negative direction of motion (physical unit) Details see "Travel Range and Soft Limits" (p. 32).

Parameter ID (hexa-decimal)	Data type	Command level for write access	Parameter name	Description
0x07000001	FLOAT	0	Range Limit Max	Additional soft limit for the positive direction of motion (physical unit) Details see "Travel Range and Soft Limits" (p. 32).
0x07000601	CHAR	0	Axis Unit	Unit symbol Details see "Physical Units" (p. 22).
0x0F000100	CHAR	2	Stage Type	Type of the stage 10-digit number in the format xxxx-x-xxxxx
0x0F000200	CHAR	2	Stage Serial Number	Serial number of the stage 8-digit number
0x0F000300	CHAR	2	Stage Assembly Date	Assembly date of the stage Date format: DDMMYY
0x0F000400	INT	2	Stage HW Version	Version number of the stage hardware
0x1F000000	FLOAT	1	PIShift Upper Supply Voltage (V)	Maximum value of the piezo voltage for the PIShift inertia drive Details see "PIShift Drive Modes" (p. 20).
0x1F000100	FLOAT	1	PIShift Lower Supply Voltage (V)	Minimum value of the piezo voltage for the PIShift inertia drive Details see "PIShift Drive Modes" (p. 20).
0x1F000200	FLOAT	1	PIShift Forward Current (A)	Maximum current consumption of the PIShift inertia drive during the forward motion in step mode. Details see "PIShift Drive Modes" (p. 20).
0x1F000300	FLOAT	1	PIShift Backward Current (A)	Maximum current consumption of the PIShift inertia drive during the backward motion in step mode. Details see "PIShift Drive Modes" (p. 20).

Parameter ID (hexa-decimal)	Data type	Command level for write access	Parameter name	Description
0x1F000400	FLOAT	1	PIShift Frequency (Hz)	Frequency of the piezo voltage for the step mode of the PIShift inertia drive (= frequency of the modified sawtooth signal) Details see "PIShift Drive Modes" (p. 20).
0x1F000500	FLOAT	1	PIShift Charge Cycle	Duty cycle of the current source during the output of one period of the modified sawtooth in step mode Details see "PIShift Drive Modes" (p. 20).
0x1F000700	FLOAT	1	PIShift Step Size (Phys. Unit)	Size of the slow individual steps in closed-loop operation Details see "Servo Algorithm and Other Control Value Corrections" (p. 26).
0x1F000701	FLOAT	1	PIShift Delay (ms)	Delay time for closed-loop operation Details see "Servo Algorithm and Other Control Value Corrections" (p. 26).
0x1F000702	INT	1	PIShift Open-Loop Driving Mode	PIShift drive mode in open-loop operation Details see "PIShift Drive Modes" (p. 20).





# 11 Maintenance

## In this Chapter

Cleaning the E-871 .....	291
Updating Firmware .....	291

## 11.1 Cleaning the E-871

### NOTICE



#### Short circuits or flashovers!

The E-871 contains electrostatic sensitive devices that can be damaged by short circuits or flashovers when cleaning fluids enter the case.

- Before cleaning, remove the E-871 from the power source by pulling the power plug.
- Prevent cleaning fluid from entering the case.

- If necessary, clean the surfaces of the E-871 case with a towel that has been lightly dampened with a mild cleanser or disinfectant.
- Do **not** use any organic solvents.

## 11.2 Updating Firmware

### INFORMATION

The `*IDN?` command reads the version number of the firmware among other things.

Example of a response of the E-871:

```
©2013 Physik Instrumente (PI) GmbH & Co. KG, E-871.1A1,  
112062031, 01.000
```

- E-871.1A1: Device name
- 01.000: Firmware version

**INFORMATION**

If the E-871 is in firmware update mode (DIP switch 8 in "ON" (upper) position), the DIP switch settings for baud rate and controller address are ignored. When communication is established with the **PI Firmware Updater** program, the baud rate is set automatically.

---

**INFORMATION**

When updating the firmware, the parameter settings of the E-871 are not changed. When new parameters are introduced with the firmware update, a special initialization for the values of the new parameters is required prior to operating the E-871.

---

**INFORMATION**

When updating the firmware the macros saved on the E-871 are retained.

---

**Prerequisite**

- ✓ You have connected the E-871 to the PC via the RS-232 interface (p. 64).
- ✓ You have made sure that the E-871 is **not** part of a daisy chain.
- ✓ You have made sure that **no** cable is connected to the **RS-232 Out** socket.
- ✓ The **PI Firmware Updater** program is installed on the PC (p. 55).
- ✓ You have received a new firmware file from our customer service department (p. 301) and copied it to a directory on the PC.
- ✓ You have read and understood the documentation which you received from our customer service department together with the new firmware file. You have learned from the documentation whether new parameters are introduced with the firmware update.

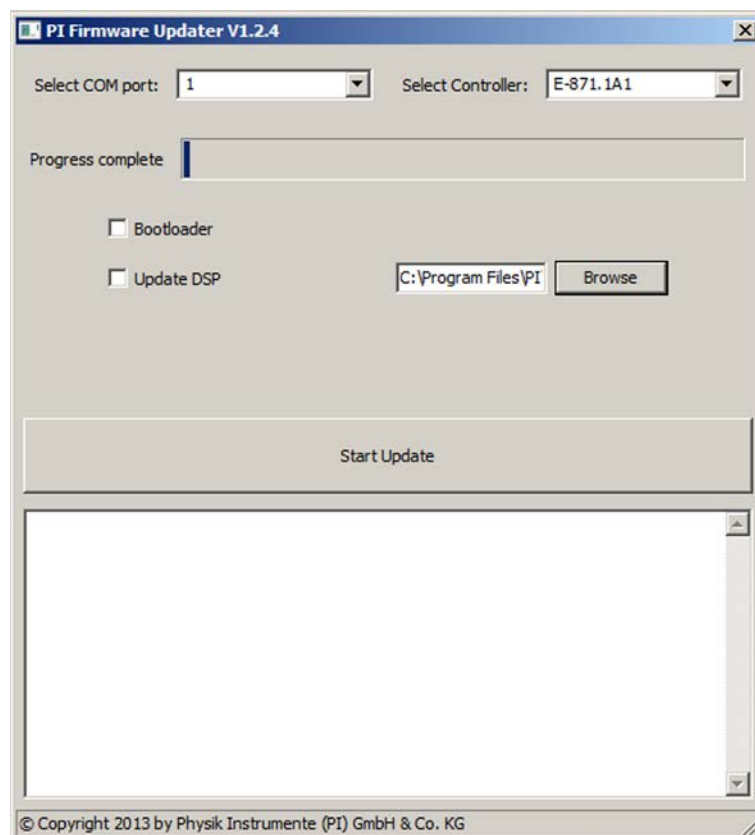
**Updating firmware of the E-871**

1. If new parameters are introduced with the firmware update: Save the current parameter values of the E-871 in a text file (p. 272).
2. Switch off the E-871 by pulling the power cord of the power supply.
3. Set DIP switch 8 on the E-871 to the firmware update mode (p. 74) (ON position).

4. Switch on the E-871 by connecting the power cord of the power supply to the power socket.
5. Start the **PI Firmware Updater** program on the PC.

The **PI Firmware Updater** window opens.

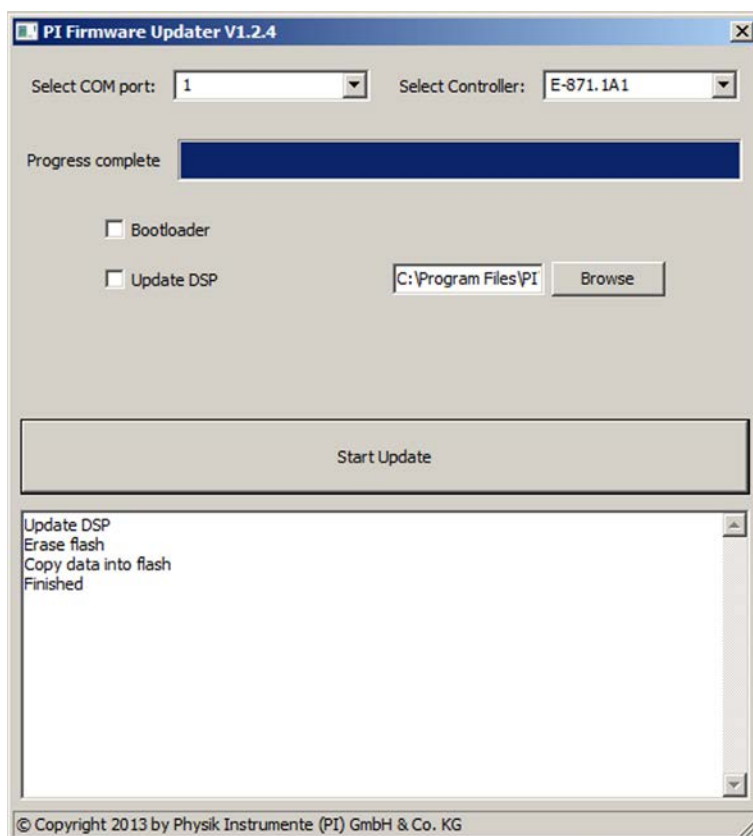
6. In the **Select COM** field, select the COM port of the PC to which you have connected the E-871.
7. In the **Select Controller:** field, select the *E-871.1A1* entry.
8. Select the new firmware file:
  - a) Click the **Browse** button.
  - b) In the file selection window, go to the directory in which you have stored the firmware file.
  - c) Here double-click the new firmware file.



9. Start the firmware update by clicking on the **Start Update** button.

The firmware of the E-871 is updated. The progress of the update is displayed in the message list and by the progress bar. Depending on the progress of the updating, checks are also shown and hidden in the **Bootloader** and **Update DSP** fields.

The update was successful when the **Finished** message appears in the message list.



10. Close the **PI Firmware Updater** program by clicking the cross in the top right corner of the window.
11. Switch off the E-871 by pulling the power cord of the power supply.
12. Set DIP switch 8 on the E-871 to normal operation (p. 74) (OFF position).
13. Switch on the E-871 by connecting the power cord of the power supply to the power socket.

Were new parameters introduced with firmware update?

- If no: Firmware update is finished.
- If yes: A special initialization for the values of the new parameters is required prior to operating the E-871, see below.

### Initializing new parameters of the E-871

1. Start PITerminal or PIMikroMove on the PC and, if necessary, open the window for sending commands.
2. Make sure that the current parameter values of the E-871 have been saved in a text file (see "Saving Parameter Values in a Text File" (p. 272)).

When the new parameters are initialized, **all** parameters of the E-871 are set to their default settings. Parameter values that are not saved are lost during the initialization process as a result.

3. Initialize the new parameters by sending the `ZZZ 100 parameter` command.
4. Adapt the parameter values (see General Procedure for Changing Parameters (p. 273)):
  - Reset the parameters that were already present prior to the firmware update to the saved values from the text file.
  - Set the parameters that were introduced with the firmware update to the appropriate values.



## 12 Troubleshooting

Problem	Possible Causes	Solution
The stage does not move	The cable is not connected correctly	➤ Check the cable connections.
	The stage or the stage cable is defective	➤ If available, replace the defective stage with another stage and test the new combination.
	Limit switch signal logic set incorrectly	In order for the stage to be able to move, the settings of the E-871 must correspond to the limit switch logic level of the stage; see "Limit Switch Detection" (p. 31). ➤ Set the <b>Limit Mode</b> parameter (ID 0x18) appropriately; see "Changing Parameters: General Procedure" (p. 273).
	Limit switch signals not compatible with the E-871	Stages from third-party suppliers may use unsuitable limit switch signals. ➤ Contact the customer service department (p. 301) and the manufacturer of the stage.
	Incorrect configuration	➤ Check the parameter settings of the E-871 with the <code>SPA?</code> (volatile memory) and <code>SEP?</code> (nonvolatile memory) commands. Details about parameter settings see "Adapting Settings" (p. 271).
	Incorrect command or incorrect syntax	➤ Send the <code>ERR?</code> command and check the error code this returns.
	Incorrect axis commanded	An axis identifier is even required in commands on systems with only one axis. ➤ Make sure that the correct axis identifier is used and that the commanded axis belongs to the correct stage.
	HID control is enabled	Motion commands are not allowed when the HID control is enabled for the axis. ➤ Disable the HID control with the <code>HIN</code> command (p. 202).

Problem	Possible Causes	Solution
Stage executes unintentional motion	HID device is not connected, but the HID control is enabled in the E-871	➤ Only activate the HID control when a HID device is actually connected to the E-871.
	HID axis is not calibrated	➤ Calibrate the axis of the HID device (p. 124).
	Start-up macro is executed	➤ Check whether a macro is specified as a start-up macro and, if necessary, cancel the selection of the start-up macro (p. 133).
Stage is oscillating or positions inaccurately	The load was changed.	➤ Reset the notch filter (p. 89) and the servo-control parameters (p. 94) corresponding to the load change.
Stage is oscillating already during the reference move	Very high load on the stage	<p>In case of a very high load, proceed with PIMikroMove during the reference move as follows:</p> <ol style="list-style-type: none"> <li>1. Do <b>not</b> start the reference move in the <b>Start up axes</b> step, but click on <b>Close</b> to close the <b>Start up controller</b> window instead.</li> <li>2. In the main window, open the single axis window for the stage connected by selecting the stage in the <b>View &gt; Single Axis Window</b> menu.</li> <li>3. Expand the view of the single axis window by clicking on the &gt; button at the right edge of the window.</li> <li>4. With the <b>Servo</b> check box, make sure that the servo mode is switched on.</li> <li>5. Start the reference move by clicking on one of the <b>Reference...</b> buttons.</li> <li>6. If the stage is oscillating: Immediately stop the reference move in the <b>Reference Axes</b> dialog, close the dialog and switch off the servo mode by removing the check from the respective check box in the single axis window.</li> <li>7. Enter suitable values for the settings of the notch filter, see "Setting the Notch Filter" (p. 89)</li> <li>8. Re-start the reference move.</li> </ol>



Problem	Possible Causes	Solution
Communication with the controller fails	The wrong communication cable is used or it is defective	<ul style="list-style-type: none"> <li>➤ Use a null-modem cable for the RS-232 connection (p. 64).</li> <li>➤ If necessary, check whether the cable works on a fault-free system.</li> </ul>
	USB driver not installed	<p>In order to be able to establish communication between the E-871 and PC via the USB interface, drivers from the product CD have to be installed. If communication via USB cannot be established or the PC operating system reports that new hardware was found:</p> <ol style="list-style-type: none"> <li>1. Log in on the PC with administrator rights.</li> <li>2. Insert the product CD in the drive of the PC.</li> <li>3. Follow the instructions on the PC screen or open the <b>Properties of PI motion controllers</b> window in the appropriate manner.</li> <li>4. Select the appropriate drivers in the E_871__USB_Driver_Setup directory on the product CD when prompted to do so.</li> </ol> <p>If the E-871 is still not detected by the PC after the drivers have been successfully installed:</p> <ul style="list-style-type: none"> <li>➤ Pull the USB cable out of the E-871 and plug it back in.</li> </ul>
	Baud rate not configured correctly	<ul style="list-style-type: none"> <li>➤ Check the settings of DIP switches 5 and 6 for the baud rate (p. 74).</li> <li>➤ In a daisy chain network make sure that the same baud rate is set for every controller.</li> </ul>
	Controller address not configured correctly	<ul style="list-style-type: none"> <li>➤ Check the settings of DIP switches 1 to 4 for the controller address (p. 73).</li> </ul>
	A different program is accessing the interface.	<ul style="list-style-type: none"> <li>➤ Close the other program.</li> </ul>

Problem	Possible Causes	Solution
	Problems with special software	<ul style="list-style-type: none"> <li>➤ Check whether the system works with different software, such as a terminal program or a development environment.</li> </ul> <p>You can test the communication by starting a terminal program (such as e. g. PI Terminal) and entering <code>*IDN?</code> or <code>HLP?</code>.</p> <ul style="list-style-type: none"> <li>➤ Make sure that you end the command with an LF (line feed).</li> </ul> <p>A command is not executed until the LF has been received.</p>
The customer software does not run with the PI drivers	Incorrect combination of driver routines/Vis	<ul style="list-style-type: none"> <li>➤ Check whether the system works with a terminal program.</li> </ul> <p>If so:</p> <ul style="list-style-type: none"> <li>➤ Read the information in the corresponding software manual and compare the sample code on the product CD with your program code.</li> </ul>
LEDs do not light up even though the E-871 is switched on	Firmware update mode set	<ol style="list-style-type: none"> <li>1. Switch off the E-871 by pulling the power cord of the power supply.</li> <li>2. Set DIP switch 8 on the E-871 to normal operation (p. 74) (OFF position).</li> <li>3. Switch on the E-871 by connecting the power cord of the power supply to the power socket.</li> </ol>

If the problem that occurred with your system is not listed in the table above or cannot be solved as described, contact our customer service department (p. 301).

## 13 Customer Service

For inquiries and orders, contact your PI sales engineer or send us an e-mail (<mailto:info@pi.ws>).

If you have questions concerning your system, have the following information ready:

- Product codes and serial numbers of all products in the system
- Firmware version of the controller (if present)
- Version of the driver or the software (if present)
- Operating system on the PC (if present)

The latest versions of the user manuals are available for downloading (p. 5) on our website.



## 14 Technical Data

### In this Chapter

Specifications .....	303
System Requirements .....	306
Dimensions .....	306
Pin Assignment.....	308

### 14.1 Specifications

#### 14.1.1 Data Table




	<b>E-871.1A1</b>
Function	Piezomotor controller for PIShift drives and positioning systems
Channels	1
<b>Motion and control</b>	
Servo characteristics	PID controller, parameter changes on-the-fly
Encoder input	Analog encoder input sine-cosine, interpolation selectable up to 20000; Interpolation circuit for differential transmission 1 V <sub>pp</sub> and 2.5 V offset of the encoder signal
Stall detection	Servo off
Input limit switch	2 × TTL (pull-up/pull-down, programmable)
Input reference switch	1 × TTL and Zero+ & Zero- for integrated reference in the encoder
<b>Electrical properties</b>	
Max. output power	30 W
Output voltage	0 to 100 V, drive-dependent selection
Max. operating current	1.5 A
<b>Interface and operation</b>	
Communication interfaces	USB, RS-232 (9-pin (m) Sub-D)
Motor connector	HD Sub-D 15-pin (f)




Sensor connection	HD Sub-D 15-pin (m)
Controller network	Up to 16 units on single interface*
I/O ports	4 analog/digital in, 4 digital out
Command set	PI General Command Set (GCS)
User software	PIMikroMove, PITerminal
Software drivers	LabVIEW drivers, shared libraries for Windows and Linux
Supported functionality	Point-to-point motion, start-up macro, data recorder for recording parameters as motor input voltage, position or position error; internal safety circuitry: watchdog timer; ID chip
Manual control (optional)	Pushbutton box, joystick (for 2 axes), Y-cable for 2-D motion
<b>Miscellaneous</b>	
Operating voltage	24 V, in the scope of delivery: external power supply with 24 V / 2.0 A
Operating temperature range	0 to 50°C
Mass	1.1 kg
Dimensions	205 mm × 130 mm × 55 mm (incl. mounting rails)

\* 16 units via USB; 6 units via RS-232.

### 14.1.2 Maximum Ratings

The E-871 is designed for the following operating data:

Input on:	Maximum Operating Voltage	Operating Frequency	Maximum Current Consumption
			
M8 panel plug, 4-pin (m)	24 V	==	2.5 A

Output on:	Maximum Output Voltage 	Maximum Output Current 	Maximum Output Frequency 
HD Sub-D socket, 15-pin (f): pins 1 and 6	100 V	$\pm 650$ mA	25 kHz

### 14.1.3 Ambient Conditions and Classifications

The following ambient conditions and classifications must be observed for the E-871:

Area of application	For indoor use only
Maximum altitude	2000 m
Relative humidity	Highest relative humidity 80% for temperatures up to 31°C Decreasing linearly to 50% relative humidity at 40°C
Storage temperature	0°C to 70°C
Transport temperature	–25°C to +85°C
Overvoltage category	II
Protection class	I
Degree of pollution	2
Measurement category	I
Degree of protection according to IEC 60529	IP20

## 14.2 System Requirements

The following system requirements must be met to operate the E-871:

- PC with Windows (XP, Vista, 7) or Linux operating system and at least 30 MB free memory
- Communication interface to the PC:
  - Free COM port on the PC
  - or -
  - USB A socket on the PC
- E-871 with power supply
- Mechanical system (stage) with PIShift inertia drive
- RS-232 null-modem cable or USB cable for connecting the controller to the PC
- Product CD with PC software

## 14.3 Dimensions

Dimensions in mm. Note that the decimal places are separated by a comma in the drawings.

Standard tolerance according to DIN ISO 2768 - f - H

Roughness Ra 1.6

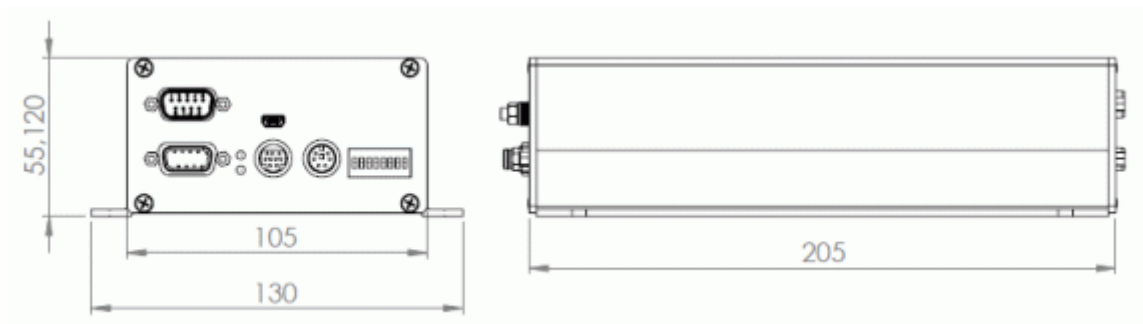


Figure 23: E-871 dimensions



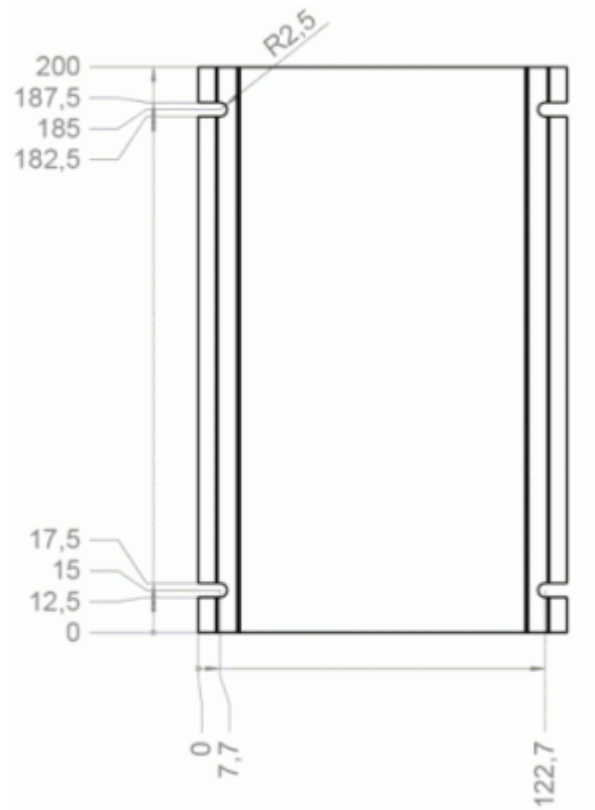


Figure 24: E-871 mounting rails with recesses

## 14.4 Pin Assignment

### 14.4.1 Motor

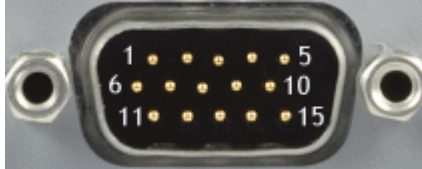
HD Sub-D socket, 15-pin, female



Pin	Signal	Function
1	PIEZO_V- Output	Piezo voltage for PIShift inertia drive (0 to 100 V)
2	-	not connected
3	IDCHIP_IO_C Bidirectional	ID chip
4	LIMN_C Input	Negative Limit Switch
5	LIMP_C Input	Positive Limit Switch
6	Piezo_V+ Output	Piezo voltage for PIShift inertia drive (0 to 100 V)
7	-	not connected
8	-	not connected
9	-	Reserved
10	REF_C Input	Reference switch, single-ended signal
11	GND	GND
12	GND	GND
13	GND	GND
14	GND	GND
15	GND	GND

## 14.4.2 Sensor

### HD Sub-D panel plug, 15-pin, male



Pin	Signal	Function
1	REF_C Input	Reference switch, single-ended signal
2	VDD_SWITCHED Output	5 V Supply voltage for encoder, reference point and limit switches
3	ZERO_P_C Input	Reference point switch, differential
4	COS_P_C Input	Encoder B (+)
5	SIN_P_C Input	Encoder A (+)
6	LIMP_C Input	Positive Limit Switch
7	LIMN_C Input	Negative Limit Switch
8	ZERO_N_C Input	Reference point switch, differential
9	COS_N_C Input	Encoder B (-)
10	SIN_N_C Input	Encoder A (-)
11	IDCHIP_IO_C Bidirectional	ID chip
12	VDD_SWITCHED Output	5 V Supply voltage for encoder, reference point and limit switches
13	GND	GND
14	GND	GND
15	GND	GND

### 14.4.3 I/O

#### Mini-DIN socket, 9-pin, female

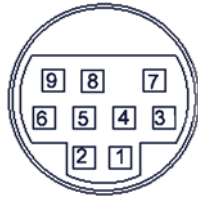


Figure 25: Front view of Mini-DIN, 9-pin

Pin	Function
1	Input 1 (analog: 0 to +5V / digital: TTL)
2	Input 2 (analog: 0 to +5V/ digital: TTL)
3	Input 3 (analog: 0 to +5V/ digital: TTL)
4	Input 4 (analog: 0 to +5V/ digital: TTL)
5	Output 1 (digital: TTL)
6	Output 2 (digital: TTL)
7	Output 3 (digital: TTL)
8	Output 4 (digital: TTL)
9	Vcc (+5 V)

### 14.4.4 C-170.IO Cable for Connecting to the I/O Socket

#### Mini-DIN connector, 9-pin, male, open end

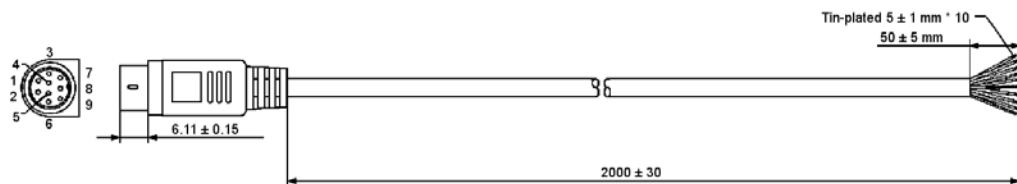


Figure 26: C-170.IO cable

#### Specifications

Temperature range: -25 °C to +85 °C

Nominal current: 1 A AC/DC

Insulation resistance: 50 MW min.

Nominal voltage: 50 V AC/DC

Voltage impulse: 500 V AC for 1 minute

Pin	Wire color	Function on the I/O socket of the E-871
1	black	Input 1 (analog: 0 to +5V / digital: TTL)
2	white	Input 2 (analog: 0 to +5V / digital: TTL)
3	red	Input 3 (analog: 0 to +5V / digital: TTL)
4	yellow	Input 4 (analog: 0 to +5V / digital: TTL)
5	purple	Output 1 (digital, TTL)
6	blue	Output 2 (digital, TTL)
7	green	Output 3 (digital, TTL)
8	brown	Output 4 (digital, TTL)
9	gray	Vcc (+5V)
Sheath	Shield, coated black (thicker than the wire connected to pin 1)	GND

### 14.4.5 Joystick

#### Mini-DIN socket, 6-pin, female (PS/2)



Figure 27: Front view of Mini-DIN socket, 6-pin

Pin	Function
1	GND
2	Input: axis 2 of HID device 1 (-10 to 10 V)
3	Output: Vcc (3.3 V)
4	Input: axis 1 of HID device 1 (0 to 3.3 V)
5	Input: button 1 of HID device 1 (0 or 3.3 V)
6	Input: button 2 of HID device 1 (0 or 3.3 V)

### 14.4.6 C-819.20Y Cable for C-819.20 Joystick

The C-819.20Y cable makes it possible to connect 2 controllers to the C-819.20 joystick.



Figure 28: Y cable C-819.20Y for joystick with 2 controllers

**Mini-DIN connector, 6-pin, female on 2 Mini-DIN connectors, 6-pin, male**

Mini-DIN 6-pin, female (to joystick)	Signal	Mini-DIN, 6-pin, male, X branch (to controller 1)	Mini-DIN, 6-pin, male, Y branch (to controller 2)
Pin 1	GND	Pin 1	Pin 1
Pin 2	Button for joystick Y axis	Not connected	Pin 6
Pin 3	Joystick power source	Pin 3	Not connected
Pin 4	Joystick X axis	Pin 4	Not connected
Pin 5	Joystick Y axis	Not connected	Pin 4
Pin 6	Button for joystick X axis	Pin 6	Not connected

### 14.4.7 RS-232 In and RS-232 Out

**RS-232 In: Sub-D panel plug, 9-pin, male**



**RS-232 Out: Sub-D socket, 9-pin, female**



Pin	Function
1	Not connected
2	RxD (PC to controller)
3	TxD (controller to PC)
4	Not connected
5	GND
6	Not connected
7	Not connected
8	Not connected
9	Not connected

#### **INFORMATION**

The pins of the **RS-232 In** and **RS-232 Out** sockets are connected together in the E-871 1:1.

#### **INFORMATION**

In a daisy chain network that is connected to the PC via the RS-232 interface of the first controller, only the PC feeds the RxD line. Depending on how performant the RS-232 driver of the PC is, the range of the network may be limited to 6 devices.

**INFORMATION**

The E-871 copies every signal that it receives from the PC via USB to the RxD line of the **RS-232 In** and **RS-232 Out** sockets. The E-871 copies the signal of the TxD line via USB to the PC.

**14.4.8 Power supply connector 24 V DC**

Phoenix M8 panel plug, 4-pin, male



Pin	Function
1	GND (power)
2	GND (power)
3	Input: 24 V DC
4	Input: 24 V DC



## 15 Old Equipment Disposal

In accordance with the applicable EU law, electrical and electronic equipment may not be disposed of with unsorted municipal wastes in the member states of the EU.

When disposing of your old equipment, observe the international, national and local rules and regulations.

To meet the manufacturer's product responsibility with regard to this product, Physik Instrumente (PI) GmbH & Co. KG ensures environmentally correct disposal of old PI equipment that was first put into circulation after 13 August 2005, free of charge.

If you have old PI equipment, you can send it postage-free to the following address:

Physik Instrumente (PI) GmbH & Co. KG  
Auf der Römerstr. 1  
D-76228 Karlsruhe, Germany

