

PZ274E
E-873 Q-Motion® Controller
User Manual

Version: 1.2.0

Date: 19.11.2019



This document describes the following product:

- **E-873.3QTU**
Q-Motion® controller for piezoelectric inertia drives, 3 axes, benchtop device (industry), TCP/IP, USB, I/O, joystick



The following company names and brands are registered trademarks of Physik Instrumente (PI) GmbH & Co. KG:

PI®, NanoCube®, PICMA®, PILine®, NEXLINE®, PiezoWalk®, NEXACT®, Picoactuator®, Plnano®, PIMag®, Q-Motion®

Notes on brand names and third-party trademarks:

Microsoft® and Windows® are registered trademarks or trademarks of Microsoft Corporation in the USA and/or other countries.

EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.

TwinCAT® is a registered trademark of and licensed by Beckhoff Automation GmbH.

LabVIEW, National Instruments and NI are trademarks of National Instruments. Neither the driver software nor the software programs offered by PI or other goods and services are connected to or sponsored by National Instruments.

Python® is a registered trademark of Python Software Foundation.

BiSS is a registered trademark of iC-Haus GmbH.

The following designations are protected company names, trademarks or registered trademarks of other owners:

Linux, MATLAB, MathWorks

The patents held by PI are found in our patent list: <http://www.physikinstrumente.com/en/about-pi/patents>

The software products provided by PI are subject to the General Software License Terms of Physik Instrumente (PI) GmbH & Co. KG and may contain and/or use third-party software components. Further information can be found in the General Software License Terms http://www.physikinstrumente.com/download/EULA_PhysikInstrumenteGmbH_Co_KG.pdf and in the Third-Party Software Notes http://www.physikinstrumente.com/download/TPSWNNote_PhysikInstrumenteGmbH_Co_KG.pdf on our website.

© 2019 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. The use of any text, images and drawings is permitted only in part and only when indicating the source.

Original instructions

First printing: 19.11.2019

Document number: PZ274E, ASt, Version 1.2.0

Subject to change without notice. This manual is superseded by any new release. The latest release is available for download (p. 4) on our website.

Contents

1	About this Document	1
1.1	Objective and Target Audience of this User Manual.....	1
1.2	Symbols and Typographic Conventions.....	1
1.3	Definition of Terms.....	2
1.4	Figures	3
1.5	Other Applicable Documents	4
1.6	Downloading Manuals.....	4
2	Safety	5
2.1	Intended Use	5
2.2	General Safety Instructions	5
2.3	Organizational Measures.....	6
3	Product Description	7
3.1	Product View	7
3.1.1	Front Panel	7
3.1.2	Type Plate	9
3.2	Scope of Delivery.....	9
3.3	Accessories	10
3.4	Overview of PC Software.....	10
3.5	Positioner Databases.....	12
3.6	ID Chip Detection.....	13
3.7	Communication Interfaces	14
3.8	Functional Principles	16
3.8.1	Block Diagram.....	16
3.8.2	Commandable Items.....	16
3.8.3	Important Components of the Firmware	18
3.8.4	Operating Modes.....	19
3.8.5	Physical Units.....	21
3.8.6	Motion Triggering	22
3.8.7	Servo Algorithm and Other Control Value Corrections	23
3.8.8	On-Target State	26
3.8.9	Reference Switch Detection	27
3.8.10	Limit Switch Detection.....	28
3.8.11	Travel Range and Soft Limits	29
3.8.12	Referencing.....	34

4	Unpacking	39
5	Installation	41
5.1	General Notes on Installation.....	41
5.2	Ensuring Ventilation	41
5.3	Mounting the E-873.....	41
5.4	Connecting the E-873 to the Protective Earth Conductor.....	43
5.5	Connecting the Power Supply to the E-873	44
5.6	Connecting the Positioner	44
5.7	Connecting an HID.....	45
5.8	Connecting Digital Inputs and Outputs	45
5.8.1	Connecting the Digital Outputs	45
5.8.2	Connecting the Digital Inputs	46
5.9	Installing the PC Software	47
5.9.1	Initial Installation	47
5.9.2	Installing Updates	48
5.9.3	Installing a Custom Positioner Database	50
5.10	Connecting the PC	51
5.10.1	Connecting the E-873 via the USB interface.....	51
5.10.2	Connecting the E-873 via the TCP/IP Interface	51
6	Startup	53
6.1	General Notes on Startup.....	53
6.2	Switching on the E-873.....	53
6.3	Establishing Communication	54
6.3.1	Establishing Communication via the USB Interface	54
6.3.2	Establishing Communication via the TCP/IP Interface	56
6.4	Starting Motion	62
6.5	Setting the Notch Filter	66
6.6	Optimizing the Servo Control Parameters	71
7	Operation	77
7.1	Protective Functions of the E-873	77
7.1.1	Protection Against Overheating	77
7.1.2	Behavior with Motion Errors	77
7.1.3	Behavior During System Errors.....	78
7.1.4	Re-establishing Readiness for Operation	78
7.2	Data Recorder.....	78
7.2.1	Setting up the Data Recorder	78
7.2.2	Starting the Recording.....	80
7.2.3	Reading Recorded Data	80
7.3	Digital Output Signals	80
7.3.1	Commands for Digital Outputs	80

7.3.2	Setting Up "Position Distance" Trigger Mode	82
7.3.3	Setting Up "On Target" Trigger Mode	84
7.3.4	Setting Up "Motion Error" Trigger Mode	84
7.3.5	Setting Up "In Motion" Trigger Mode	84
7.3.6	Setting Up "Position + Offset" Trigger Mode	85
7.3.7	Setting Up "Single Position" Trigger Mode	86
7.3.8	Setting Signal Polarity	87
7.4	Digital Input Signals	87
7.4.1	Commands and Parameters for Digital Inputs	88
7.4.2	Using Digital Input Signals in Macros	89
7.4.3	Using Digital Input Signals as Switch Signals	90
7.5	Controlling with HID	91
7.5.1	Functionality of HID Control	91
7.5.2	Commands and parameters for HIDs	92
7.5.3	Testing the HID	93
7.5.4	Setting Up and Enabling HID Control.....	95
7.5.5	Calibrating HID Axes	96
7.5.6	Saving the Configuration of HID Control Permanently.....	98
7.5.7	Available HIDs	99
7.6	Controller Macros.....	99
7.6.1	Overview: Macro Functionality and Example Macros.....	99
7.6.2	Commands and Parameters for Macros.....	100
7.6.3	Working with Macros	102
7.6.4	Making Backups and Loading Controller Macros	109
7.6.5	Macro Example: Stopping Motion by Pushbutton	110
8	GCS Commands	113
8.1	Notation.....	113
8.2	GCS Syntax for Syntax Version 2.0	113
8.3	Target and Sender Address	116
8.4	Variables	116
8.5	Command Overview	118
8.6	Command Descriptions for GCS 2.0	122
8.7	Error Codes.....	205
9	Adapting Settings	229
9.1	Settings of the E-873	229
9.2	Changing Parameter Values in the E-873	229
9.2.1	General Commands for Parameters	230
9.2.2	Saving Parameter Values in a Text File	231
9.2.3	Changing Parameter Values: General Procedure	232
9.3	Creating or Changing a Positioner Type	234
9.4	Parameter Overview.....	239

10	Maintenance	247
10.1	Cleaning the E-873.....	247
10.2	Updating Firmware.....	247
11	Troubleshooting	253
12	Customer Service	257
13	Technical Data	259
13.1	Specifications.....	259
13.1.1	Data Table.....	259
13.1.2	Maximum Ratings.....	260
13.1.3	Ambient Conditions and Classifications	261
13.2	System Requirements.....	261
13.3	Dimensions.....	262
13.4	Pin Assignment.....	263
13.4.1	Motor and Sensor.....	263
13.4.2	I/O.....	264
13.4.3	C-170.IO Cable for Connecting to the I/O Socket.....	264
13.4.4	Power Supply Connector 24 V DC.....	265
14	Old Equipment Disposal	267
15	EU Declaration of Conformity	269

1 About this Document

In this Chapter

Objective and Target Audience of this User Manual	1
Symbols and Typographic Conventions	1
Definition of Terms	2
Figures.....	3
Other Applicable Documents.....	4
Downloading Manuals	4

1.1 Objective and Target Audience of this User Manual

This manual contains information necessary for the intended use of the E-873.

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. 4) on our website.

1.2 Symbols and Typographic Conventions

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

CAUTION



Dangerous situation
Failure to comply could lead to minor injury.

- Precautionary measures for avoiding the risk.

NOTICE




Dangerous situation
Failure to comply could cause damage to equipment.

- Precautionary measures for avoiding the risk.

INFORMATION

Information for easier handling, tricks, tips, etc.

Symbol/Label	Meaning
1.	Action consisting of several steps whose sequential order must be observed
2.	
➤	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 sign on the product which refers to detailed information in this manual.
Start > Settings	Menu path in the PC software (example: to open the menu, the Start and Settings menu items must be clicked in succession)
POS?	Command line or a command from PI's General Command Set (GCS) (example: Command to get the axis position).
Device S/N	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 of Terms

Term	Explanation
Axis	Also referred to as "logical axis". The logical axis represents the motion of the mechanics in the firmware of the E-873. For mechanics that allow motion in several directions (e.g., in X, Y, and Z), each direction of motion corresponds to a logical axis.
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). The parameter values in the volatile memory determine the current behavior of the system. The parameter values in the volatile memory are also referred to as "Active Values" in the PC software from PI.
GCS	PI General Command Set; command set for PI controllers. Piezo drivers and servo controllers can be operated together with minimal programming effort thanks to GCS.

Term	Explanation
HID	HID (Human Interface Device) refers to an input or output device 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. Joysticks and gamepads are typical HID's.
HID control	Control of the motion parameters of the axes of the E-873 via the displacement of the axes of human interface devices.
Incremental position sensor	Sensor (encoder) for detecting changes of position or changes of angle. Signals from the incremental position sensor are used for axis position feedback. After the controller is switched on, referencing must be done before absolute target positions can be commanded and reached.
PC software	Software that is installed on the PC.
Nonvolatile memory	Memory module (read-only memory, e. g., EEPROM or flash memory), from which the default values of the parameters are loaded to the volatile memory when the controller is started. The parameter values in the nonvolatile memory are also referred to as "startup values" in the PC software from PI.
Control value	The control value is the input for the Q-Motion® driver electronics integrated in the E-873. The driver electronics generate the following output signals from the control value: <ul style="list-style-type: none"> ▪ Step mode: Conversion to the modified sawtooth signal for the positioner ▪ Linear mode: Linear conversion to an analog signal
Positioner	Mechanical system connected to the E-873. For positioners with just one motion axis the designation "axis" is synonymous with "positioner". Positioners that allow motion in several axes are also designated as "multi-axis positioners". For these positioners, a difference must be made between the individual axes.

1.4 Figures

For better understandability, the colors, proportions, and degree of detail in illustrations can deviate from the actual circumstances. Photographic illustrations may also differ and must not be seen as guaranteed properties.

1.5 Other Applicable Documents

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

Description	Document
Short instructions for the installation and startup of the E-873	MS242EK Short instructions for digital motor controllers
E-873 GCS driver library for use with NI LabVIEW software	PZ261E Software Manual
PI GCS 2.0 DLL0	SM151E Software Manual
GCS array data format description	SM146E Software Manual
PIMikroMove	SM148E Software Manual
PIStageEditor software for the management of positioner 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

The latest versions of the user manuals are available for download (p. 4) on our website.

1.6 Downloading Manuals

INFORMATION

If a manual is missing or problems occur with downloading:

- Contact our customer service department (p. 257).

Downloading manuals

1. Open the website **www.pi.ws**.
2. Search the website for the product number (e.g., P-882) or the product family (e.g., PICMA® bender).
3. Click the corresponding product to open the product detail page.
4. Click **Downloads**.

The manuals are shown under **Documentation**.

5. Click the desired manual and fill out the inquiry form.

The download link will then be sent to the email address entered.

2 Safety

In this Chapter

Intended Use.....	5
General Safety Instructions.....	5
Organizational Measures.....	6

2.1 Intended Use

The E-873 is a laboratory device as defined by DIN EN 61010-1. It is intended for indoor use and use in an environment that is free of dirt, oil, and lubricants.

In accordance with its design, the E-873 is intended for the operation of positioners with Q-Motion® piezo inertia drives.

The E-873 is intended for closed-loop operation with incremental position sensors. Furthermore, it can read out the reference switch signals of the positioners connected and process them further.

The E-873 must not be used for purposes other than those named in this user manual. In particular, the E-873 must not be used to drive ohmic or inductive loads.

2.2 General Safety Instructions

The E-873 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-873.

- Use the E-873 for its intended purpose only, and only when it is in perfect technical condition.
- Read the user manual.
- Eliminate any malfunctions that may affect safety immediately.

The operator is responsible for the correct installation and operation of the E-873.

- Install the E-873 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. 10)) to connect the E-873 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 together with the E-873.
The latest versions of the user manuals are available for download (p. 4) on our website.
- Add all information from the manufacturer to the user manual, for example supplements or technical notes.
- If you give the E-873 to a third party, include this user manual as well as other relevant information provided by the manufacturer.
- Do the work only if the user manual is complete. Missing information due to an incomplete user manual can result in minor injury and damage to equipment.
- Install and operate the E-873 only after you have read and understood this user manual.

Personnel qualification

The E-873 may only be installed, started, operated, maintained, and cleaned by authorized and appropriately qualified personnel.

3 Product Description

In this Chapter



Product View	7
Scope of Delivery	9
Accessories.....	10
Overview of PC Software	10
Positioner Databases	12
ID Chip Detection	13
Communication Interfaces.....	14
Functional Principles.....	16









3.1 Product View



3.1.1 Front Panel






Figure 1: Front panel of the E-873.3QTU

Item	Labeling	Type	Function
	0 1	Toggle switch	On/off switch: ○: E-873 switched off —: E-873 switched on
	24 V DC 4.5 A	M8 4-pin (m) (p. 265)	Connection for the supply voltage (Connection to mains the via adapter + power supply)

Item	Labeling	Type	Function
	STA	LED green	Controller state: <ul style="list-style-type: none"> On: E-873 is ready for normal operation Off: E-873 is not connected to the supply voltage or is switched off or is in firmware update mode
	ERR	LED red	Error indicator: <ul style="list-style-type: none"> On: Error (error code \neq 0) Off: No error (error code = 0) 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.
	Ethernet	RJ45	Ethernet interface for communication via TCP/IP
		Mini-B USB	Universal serial bus for connection to the PC
	I/O	Mini-DIN 9-pin (f) (p. 264)	Digital in-/outputs: <ul style="list-style-type: none"> Outputs: Control of external devices Inputs: Use in macros, as switch signals or for HID control
	Joystick	USB type A	USB connection for HID (Human Interface Device): Digital joystick
for axis 1 to 3 respectively: 	for 1 to 3 respectively: Axis # 0 V to 100 V	respectively: Sub-D 15 (f) (p. 263)	Motor&Sensor connections for positioners Only for Q-Motion® piezo inertia drives (stick-slip piezo motors)! Per channel: <ul style="list-style-type: none"> Outputs for piezo voltage Input of the signals of the incremental position sensor Input of the signals from the reference switch Output of the supply voltage for the position sensor and reference switch

Item	Labeling	Type	Function
		Threaded bolt for protective earth conductor	Protective earth connection (p. 43) The threaded bolt must be connected to a protective earth conductor, because the E-873 is not grounded via the power supply connector.

3.1.2 Type Plate


Labeling	Function
	Data matrix code (example; contains the serial number)
E-873.3QTU	Product name
PI	Manufacturer's logo
116056789	Serial number (example), individual for each E-873 Meaning of each position (from the left): 1 = internal information, 2 and 3 = year of manufacture, 4 to 9 = consecutive number
Country of origin: Germany	Country of origin
	Warning sign "Pay attention to the manual!"
	Old equipment disposal (p. 267)
CE	CE conformity mark
WWW.PI.WS	Manufacturer's address (website)

3.2 Scope of Delivery

Item	Component
E-873.3QTU	Q-Motion® controller, 3 channels, TCP/IP and USB interface, benchtop device (industry)
000023194	Wide input range power supply 24 V / 120 W for use with line voltages of 100 to 240 V AC and voltage frequencies of 50 or 60 Hz, with barrel connector socket
K050B0003	Adapter for the power adapter connection; barrel connector to M8 4-pin Connector (f)
3763	Power cord
000036360	USB cable (type A to mini-B) for connecting to the PC, 3 m

Item	Component
E-873.CD	Product CD with software and user manuals for the E-873
MS242EK	Short instructions for digital motor controllers

3.3 Accessories

Item	Component
C-815.553	Patch cable 1:1, for connecting controller and PC via a TCP/IP network
C-815.563	Patch cable twisted, for connecting controller and PC directly via TCP/IP
C-819.JD	Digital joystick for 2 axes, 3 programmable buttons, see "Available Human Interface Devices" (p. 99) for details
C-170.PB	<div>  <p>Connection to the I/O socket of the E-873, sends 4 TTL input signals and displays the state of the 4 digital outputs via the LEDs.</p> </div>
C-170.IO	I/O cable, 2 m, open end (p. 264)

To order, contact our customer service department (p. 257).

3.4 Overview of PC Software

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

- Windows: Versions 7, 8, 10 (32 Bit, 64 Bit)
- Linux: Kernel 2.6, GTK 2.0, from glibc 2.15

PC software	Operating system	Short description	Recommended use
Dynamic program library for GCS	Windows, Linux	Allows software programming for the E-873 with programming languages such as 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 the PIMikroMove and the NI LabVIEW driver (communication in Linux via the virtual COM port only).
NI LabVIEW drivers	Windows, Linux	NI LabVIEW is a software for data acquisition and process control (can be ordered separately from National Instruments). The E-873 NI LabVIEW software is a collection of virtual instrument drivers (VI drivers) for the E-873 controller. These drivers support the GCS.	For users who want to use NI LabVIEW to program their application.
NI LabVIEW Merge Tool	Windows	The NI LabVIEW merge tool allows you to combine product-specific NI LabVIEW drivers from PI with each other.	For users who want to operate several products from PI at the same time while using NI LabVIEW.
MATLAB drivers	Windows	MATLAB is a development environment and programming language for numerical calculations (can be ordered separately from MathWorks). The PI MATLAB driver consists of a MATLAB class that can be included in any MATLAB script. This class supports the PI General Command Set. The PI MATLAB driver does not require any additional MATLAB toolboxes.	For users who want to use MATLAB to program their application.
PIMikroMove	Windows	Graphical user interface for Windows, which can be used for controllers from PI: <ul style="list-style-type: none"> Start the system without programming effort Graphic representation of the motions Macro functionality for storing command sequences on the PC (host macros) Complete environment for command entry PIMikroMove uses the dynamic program library to supply commands to the controller.	For users who want to do simple automation tasks or test their equipment before or instead of programming an application. No command knowledge is necessary to operate PIMikroMove.

PC software	Operating system	Short description	Recommended use
PITerminal	Windows, Linux	Simple user interface that can be used for nearly all PI controllers.	For users who want to send GCS commands directly to the controller.
PIStageEditor	Windows	Program for opening and editing positioner databases.	For users who want to deal intensively with the contents of positioner 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 the firmware of the E-873.	For users who want to update the firmware.
USB driver	Windows	Driver for the USB interface	For connecting the controller to the PC via the USB interface.

3.5 Positioner Databases

You can select a suitable parameter set for your positioner from a positioner 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
PIMicosStages2.dat	No, updates can be downloaded from the PI website (p. 48).	Standard positioner database: Includes parameter sets for all standard positioners from PI miCos; is automatically saved to the PC when the PC software is installed.
PIStages2.dat	No, updates can be downloaded from the PI website (p. 48).	Standard positioner database: Includes parameter sets for all standard positioners from PI; is automatically saved to the PC when the PC software is installed.
PI_UserStages2.dat	Yes, new parameter sets can be created, edited, and saved (p. 234).	Is automatically created when you make a connection to your positioner for the first time using the PC software (i.e., when selecting the positioner in PIMikroMove or when using the commands <code>VST?</code> or <code>CST</code> from the dynamic program library).

Database file name	Editable?	Description
X-xxx.dat	No, you receive updates from our customer service department (p. 257).	Contains the parameter set for a custom positioner, for installation see "Installing a Custom Positioner Database" (p. 50).

The parameter values in the volatile memory of the E-873 can be queried and written to the nonvolatile memory; see "Adapting Settings" (p. 229)

INFORMATION

Positioners only contain some of the information that is required to operate a positioner with the E-873. Further information is loaded as parameter values to the volatile memory of the E-873 from the ID chip (p. 13) of the positioner when the E-873 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. 239).

3.6 ID Chip Detection

The positioners with inertia drives offered by PI miCos have an ID chip in the motor/sensor connector where the following data is stored as parameters:

- Information on the positioner: 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 positioner is loaded from the ID-Chip into the volatile memory of the E-873 when the E-873 is switched on (p. 53) or rebooted.

The parameter values in the volatile memory of the E-873 can be queried and written to the nonvolatile memory, see "Adapting Settings" (p. 229).

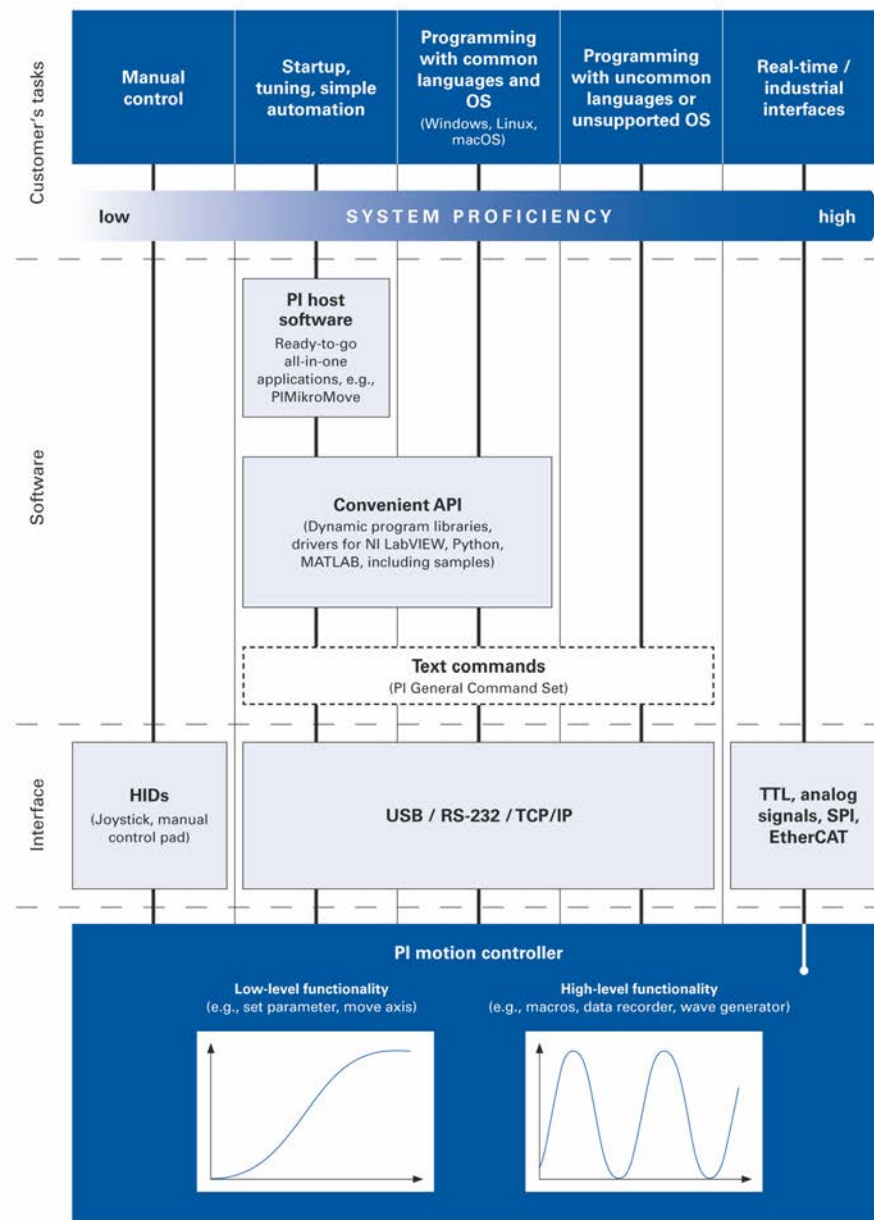
INFORMATION

The ID chip only contains some of the information that is required to operate the positioner with the E-873. When you use the PC software from PI, further information is loaded as parameter values from a positioner database (p. 12) into the volatile memory of the E-873.

Parameters that are loaded from the ID chip or from a positioner database are marked in color in the parameter overview (p. 239).

3.7 Communication Interfaces

Basically, PI systems can be controlled as follows:



Communication interfaces available

The E-873 can be controlled with ASCII commands from a PC. The following interfaces of the E-873 can be used for connecting to the PC:

- USB connection
- TCP/IP connection

The E-873 may only be connected to the PC via one of these interfaces.

Default communication settings

The communication settings of the E-873 can be changed via the `IFS` (p. 166) command. The default values for the communication interfaces are:

Interface	Property	Default value
TCP/IP	IPSTART	Startup behavior: 1 DHCP is used to obtain the IP address.
	IPADR	<IP address>:50000 e.g., 192.168.001.001:50000 Default port: 50000
	MACADR	MAC level address: 00:80:A3:F:D5:73

3.8 Functional Principles

3.8.1 Block Diagram

The E-873 can control up to three logical positioner axes. The following block diagram shows how the E-873 generates the piezo voltage for the axes connected.

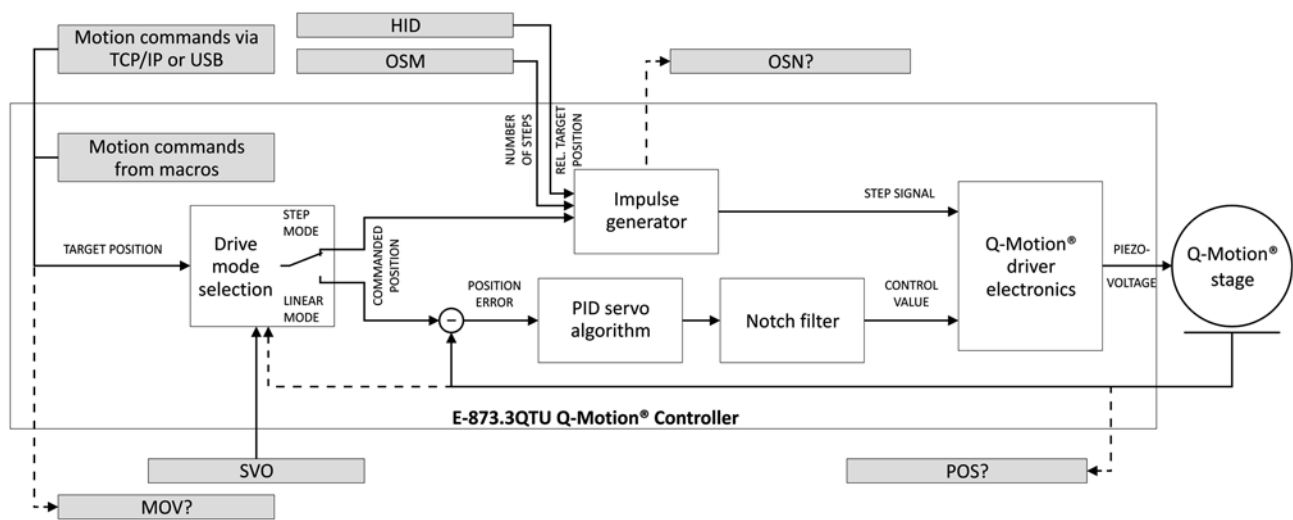


Figure 2: E-873.3QTU: Generating the piezo voltage

The E-873 supports positioners with Q-Motion® inertia drive and incremental sensor as well as all PiezoMike linear actuators.

3.8.2 Commandable Items

The following table contains the elements that can be commanded with GCS commands (p. 122).

Element	Number	Identifier	Description
Logical axes	3	1 to 3 (can be changed)	<p>A logical axis represents the motion of the positioner in the firmware of the E-873. It corresponds to an axis of a linear coordinate system.</p> <p>Motion for logical axes is commanded in the firmware of the E-873 (i.e., for the directions of motion of a positioner). The motion commands MOV and MVR, for example, are available in closed-loop operation. Motion in</p>

Element	Number	Identifier	Description
			<p>open-loop operation is triggered by OSM, OMA, and OMR. The axis identifier can be queried with the SAI? command and modified with the SAI command. It can consist of up to 8 characters; valid characters are 1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ_</p> <p>The new axis identifier is saved automatically to nonvolatile memory and is therefore still available even after a reboot or after the next switch-on.</p> <p>If the Stage Name 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 SAI? ALL.</p>
Digital outputs	4	1 to 4	<p>1 to 4 identify digital output lines 1 to 4 of the I/O (p. 264) socket.</p> <p>Refer to "Digital Output Signals" (p. 80) for further information.</p>
Digital inputs	4	1 to 4	<p>1 to 4 identify digital input lines 1 to 4 of the I/O socket (p. 264).</p> <p>Refer to "Digital Input Signals" (p. 87) for further information.</p>
HID	1	1	A digital HID can be connected to the Joystick socket (USB type A) of the E-873.
Axes and buttons of human interface devices	x	1 to x	<p>The number of commandable axes and buttons per HID depends on the human interfaces devices connected. After connecting a human interface device, information on the commandable axes and buttons can be queried with the HIS? command.</p> <p>See "Controlling with a Human Interface Device" (p. 91) for further information.</p>
Data recorder tables	4	1 to 4	The E-873 has 4 data recorder tables (query with TNR?) with 8192 data points per table.
Overall system	1	1 (not modifiable)	E-873 as an overall system

3.8.3 Important Components of the Firmware

The firmware of the E-873.3QTU provides the following functional units:

Firmware component	Description
Parameters	<p>Parameters reflect the properties of the positioner connected (e.g., travel range) and specify the behavior of the E-873 (e.g., settings for the servo algorithm).</p> <p>The parameters can be divided into the following categories:</p> <ul style="list-style-type: none"> Protected parameters whose default settings cannot be changed Parameters that must be set by the user to adapt to the application <p>For further information, see "Adapting Settings" (p. 229).</p> <p>In the case of positioners with ID chip, the values of some parameters are stored on the ID chip. They are loaded to the volatile memory when switching on or rebooting the E-873.</p>
Command levels	<p>The command levels determine the write permission for the parameters. The current command level can be changed with the CCL command. This may require entering a password.</p>
ASCII commands (GCS)	<p>Communication with the E-873 can be managed using the commands of the PI General Command Set (GCS; version 2.0). The GCS is independent of the hardware (controller, positioners connected).</p> <p>Examples of the use of GCS:</p> <ul style="list-style-type: none"> Configuring the E-873 Setting the operating mode Starting motion of the positioner Getting system and position values <p>You can find a list of the available commands in the "Command Overview" section (p. 118).</p>
Servo algorithm	<p>Closed-loop operation: The position error that results from the difference between the target position and the actual position (sensor feedback) runs through a PID servo algorithm.</p> <p>For further information, see "Servo Algorithm and Other Control Value Corrections" (p. 23).</p>
Data recorder	<p>The E-873 contains a real-time data recorder (p. 78). The data recorder can record various signals (e.g., position) from different data sources (e.g., logical axes or input channels).</p>
Macros	<p>The E-873 can save macros (p. 99). Command sequences can be defined and stored permanently in the nonvolatile memory of the device via the macro function. A startup macro can be defined that runs each time the E-873 is switched on or rebooted. The startup macro simplifies stand-alone operation (operation without a connection to the PC).</p> <p>Further information can be found in the "Controller Macros" section (p.</p>

Firmware component	Description
	99).

The firmware can be updated with a tool (p. 247).

3.8.4 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	The position error that results from the difference between the target position and the actual position (sensor feedback) runs through a PID servo algorithm (p roportional i ntegral d erivative). The commanded target position determines whether the motion is executed as a sequence of linear and step mode or only in linear mode. For further information, see "Servo Algorithm and Other Control Value Corrections" (p. 23).
Open-loop operation Servo mode off	The E-873 does not evaluate the signals of the position sensor in open-loop operation. For further information, see "Q-Motion® Drive Modes" (p. 19) and "Motion Triggering" (p. 22).

INFORMATION

The E-873 is intended for closed-loop operation with incremental position sensors (servo mode On). After switch-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 startup macro that starts the E-873 via the `SVO` command in closed-loop operation; see "Setting up a startup macro" (p. 107).

Q-Motion® Drive Modes

Q-Motion® inertia drives are piezo-based drives with a virtually unlimited travel range. They make use of the stick-slip effect (inertia effect) – a cyclical alternation of static and sliding friction between a moving runner and the piezo actuator generated by the piezo element – for a continuous feed of the runner. When at rest, the drive is self-locking, requires no current, and does not generate any heat. It holds the position with maximum force.

The E-873 supports the following drive modes for Q-Motion® inertia drives:

Operating Mode	Description
Step mode	The driver electronics of the E-873 convert the control value to a modified sawtooth signal with a maximum frequency of 25 kHz and output 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. The travel range is only limited by the physical limits of the positioner.
Linear mode	The driver electronics of the E-873 convert the control value 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-873 outputs the maximum permitted piezo voltage. The travel range is limited by the maximum expansion of the piezo actuator.

The E-873 is configured with the parameters listed in the table for the Q-Motion® inertia drive connected. All parameter values can be loaded from the database (p. 12) with the PC software from PI (recommended).

Parameters	Description and Possible Values
PIShift Upper Supply Voltage (V) 0x1F000000	Maximum value of the piezo voltage for the Q-Motion® inertia drive The value depends on the type of the drive.
PIShift Lower Supply Voltage (V) 0x1F000100	Minimum value of the piezo voltage for the Q-Motion® inertia drive The value depends on the type of the drive.
PIShift Forward Current (A) 0x1F000200	Maximum current consumption of the Q-Motion® inertia drive during forward motion The value depends on the type of the drive.
PIShift Backward Current (A) 0x1F000300	Maximum current consumption of the Q-Motion® inertia drive during backward motion The value depends on the type of the drive.
PIShift Frequency (Hz) 0x1F000400	Frequency of the piezo voltage for the step mode of the Q-Motion® inertia drive (= frequency of the modified sawtooth signal; "step frequency") Determines the velocity of the drive in step mode.

Parameters	Description and Possible Values
<i>PIShift Charge Cycle</i> 0x1F000500	Duty cycle of the current source during output of one period of the modified sawtooth signal Specification as part of a period, for which the current source is switched on 0 to 1 The value depends on the type of the drive.

3.8.5 Physical Units

The E-873 supports various units of length for positions. Adapting is done by a factor that converts the incremental encoder counts into the physical unit of length required. The conversion factor is set with the following parameters:

Parameters	Description and Possible Values
<i>Numerator Of The Counts-Per-Physical-Unit Factor</i> 0xE	Numerator and denominator of the factor for counts per physical length unit 1 to 1.000.000.000 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.
<i>Denominator Of The Counts-Per-Physical-Unit Factor</i> 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:

Parameters	Description and Possible Values
<i>Axis Unit</i> 0x07000601	Unit symbol Maximum of 20 characters. For example, the unit symbol is "MM", if the factor for the counts per physical unit of length is set with parameters 0xE and 0xF so that the encoder counts are converted into millimeters. The unit for rotation stages is normally "deg". The value of the parameter 0x07000601 is not evaluated by the E-873 but is used by the PC software for display purposes. Examples: 1 encoder count = 100 nm Counts per physical length unit: 10000:1 → Unit symbol: mm

Parameters	Description and Possible Values
	1 encoder count = 0.254 mm Counts per physical length unit: 100:1 → Unit symbol: inch

3.8.6 Motion Triggering

Motion in closed-loop operation

In closed-loop operation, motion is either triggered via commands (p. 118) or via a human interface device, e.g., a joystick.

Motion commands are not permitted when HID control (p. 91) is enabled for the axis.

Trigger of the motion	Commands	Description
Motion commands, sent from the command line or via the PC software	MOV, MVR	Motion to absolute or relative target position
	GOH	Motion to zero position
	STE	Initiates a jump over a specified distance and records the step response
	FNL, FPL, FRF	Starts reference moves
	FED	Starts motion to signal edges
Controller macros with motion commands	MAC	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.
		Additional macro commands and information see "Controller Macros" (p. 99).
HID control	HIN	Enables or disables control of the axes of the E-873 by the axes of human interface devices (HID control).
	HIA	Configures HID control for the axes of the E-873. The following motion parameters of the E-873 axis can be controlled via the axes of human interface devices: <ul style="list-style-type: none"> Relative target position (frequency that is used to move the controlled axes of the positioner)
		For further commands, see "Commands and Parameters for Human Interface Devices" (p. 92).

INFORMATION

Absolute target positions can only be commanded for positioners with incremental positions sensor when the axis has been previously referenced, refer to "Referencing" (p. 34).

Motion in open-loop operation

HID control is not possible in open-loop operation. Motions are triggered by the following commands:

Command	Description
STE	Initiates a jump with a specified number of steps and records the step response
OSM	Moves an axis by a specific number of steps
OMA	Moves an axis to an absolute target position
OMR	Moves an axis to a relative target position

The motion triggered by commands can be stopped using the following commands:

- #24, STP: abrupt stop
- HLT: gentle stop

In both cases, the error code 10 is set for information.

3.8.7 Servo Algorithm and Other Control Value Corrections

In closed-loop operation, the control value for the PIShift driver electronics integrated in the E-873 and therefore the settling behavior of the system is optimized by a PID servo algorithm (**proportional integral derivative**). 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	<p>Sequence of the following four steps:</p> <ol style="list-style-type: none"> 1. Linear mode 2. Fast step mode 3. Slow individual steps 4. Linear mode <p>The control value correction via the PID servo algorithm and the notch filter is only carried out during linear mode in step 4 of the motion.</p>
Yes	Linear mode with control value correction via PID servo algorithm and notch filter

Settings for the 4-step motion sequence

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

Parameters	Description and Possible Values
PIShift Step Size (Phys. Unit) 0x1F000700	<p>Size of the slow individual steps</p> <p>Also serves as a criterion for switching between the following phases of the motion sequence:</p> <ul style="list-style-type: none"> ▪ Fast step mode > slow individual steps ▪ Slow individual steps > linear mode
PIShift Delay (ms) 0x1F000701	<p>Delay time</p> <p>Specifies the following:</p> <ul style="list-style-type: none"> ▪ Length of time between the end of the linear mode in step 1 and the beginning of fast step mode ▪ Length of time between the end of the fast step mode and the beginning of the slow individual steps ▪ Length of time between slow individual steps ▪ Length of time between the last slow individual step and the beginning of linear mode

Settings for the servo algorithm

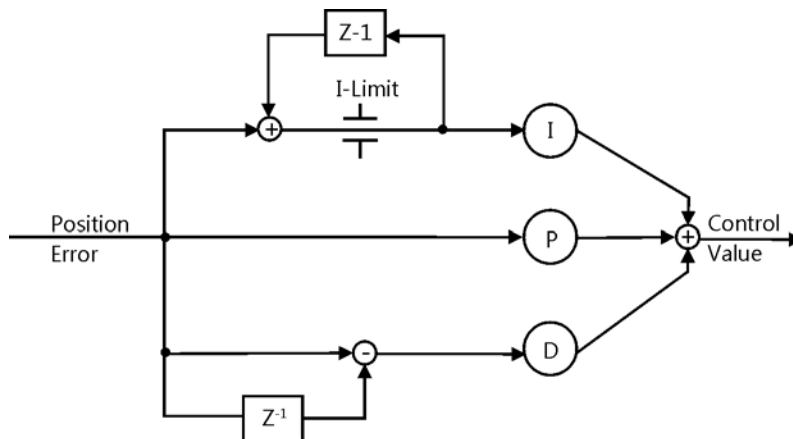


Figure 3: PID algorithm

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 the Servo Control Parameters" (p. 71).

Parameters	Description and Possible Values
P-Term 0x1	Proportional constant (dimensionless) 0 to 32767 Aim: Rapid correction of the position error
I-Term 0x2	Integration constant (dimensionless) 0 to 32767 Aim: Reduction of the static position error
D-Term 0x3	Differential constant (dimensionless) 0 to 32767 The default setting of this parameter value is 0 and should not be changed.
I-Limit 0x4	Limit of the integration constant (dimensionless) 0 to 32767
D Term Delay (No. Of Servo Cycles) 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-873 with the following parameters:

Parameters	Description and Possible Values
Numerator Of The Servo-Loop Input Factor 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.
Denominator Of The Servo-Loop Input Factor 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:

Parameters	Description and Possible Values
Notch Filter Frequency 1 (Hz) 0x94	Frequency of the first notch filter 40 to 20000 Hz The corresponding frequency component is reduced in the control value to compensate for undesired resonances in the mechanics. An adjustment can be particularly useful in the case of very high loads.
Notch Filter Edge 1 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.

To set the notch filter, see "Setting the Notch Filter" (p. 66).

3.8.8 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-873 determines the on-target state on the basis of 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. 84), the on-target state of the selected axis is output at the selected trigger output.

Parameters	Description and Possible Values
Settling Time (s) 0x3F	Delay time for setting the on-target state 0 to 1.000 s
Settling Window (encoder counts) 0x36	Settling window around the target position 0 to 2^{31} counts of the 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.

3.8.9 Reference Switch Detection

The E-873 receives reference switch signals of an axis on the following connections of the panel plug **Axis 1**, **Axis 2**, and **Axis 3** (motor connection Sub-D 15 (f) (p. 263)):

- Pin 1: Reference switch, differential (-)
- Pin 13: Reference switch, differential (+)

The following parameters can be used to configure how the E-873 detects the reference switch:

Parameters	Description and Possible Values
Invert Reference? 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 switch or a digital input which is used instead of the reference switch (p. 90).
Has Reference? 0x14	Does the positioner have a reference switch? 0 = Reference switch not installed 1 = Reference switch present (signal input as described above) This parameter activates or deactivates reference moves to the installed reference switch.

Parameters	Description and Possible Values
Reference Signal Type 0x70	Reference signal type 0 = Direction-sensing reference switch. The signal level changes when passing the reference switch. 1 = Pulse signal with a pulse width of several nanoseconds (parameter 0x47 must be set correctly). 2 = Index pulse. The approach is done 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). If the signal of the reference switch of the positioner is to be used for reference moves, the value of this parameter must be set to 2 or 3.

The signal from the reference switch of the positioner can be used for reference moves. After a reference move to the reference switch, the controller knows the absolute axis position; see "Referencing" (p. 34).

3.8.10 Limit Switch Detection

The E-873 does not receive limit switch signals at **Axis 1** to **Axis 3** connections (p. 263) for positioners.

If limit switch signals are to be used for an axis, they must be supplied via the digital inputs on the **I/O** socket; see "Using Digital Input Signals as Switch Signals" (p. 90).

The signals from the limit switches (also end-of-travel sensors) of a linear stage are used to stop motion in front of the hard stop at both ends of the travel range. In addition, soft limits can be set via parameters of the E-873; see "travel range and Soft Limits" (p. 29).

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 "Referencing" (p. 34).

The following parameters are available for evaluation of limit switch signals from the E-873:

Parameters	Description and Possible Values
Limit Mode 0x18	Signal logic of the limit switches 0 = pos-HI, neg-HI 1 = pos-LO, neg-HI 2 = pos-HI, neg-LO 3 = pos-LO, neg-LO posHI/pos-LO - positive limit switch active high/active low neg-HI/neg-LO - negative limit switch active high/active low

Parameters	Description and Possible Values
<i>Has No Limit Switches?</i> 0x32	Does the positioner have limit switches? 0 = Stage has limit switches 1 = stage has no limit switches The value of this parameter is 1 for the E-873. This parameter has no influence on the use of digital input lines as the source of the limit switch signal. That means, even when digital input lines are used as source for limit switch signals, it is not necessary to set this parameter to 0.
<i>Use Limit Switches Only For Reference Moves?</i> 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 1 = Use limit switches only for reference moves This parameter is intended for use with rotation stages.

3.8.11 Travel Range and Soft Limits

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

- Limit switches
- If the positioner does not have integrated limit switches: Hard stops

The following parameters of the E-873 reflect the physical travel range of the positioner and define soft limits:

Parameter	Description and Possible Values
<i>Maximum Travel In Positive Direction (Phys. Unit)</i> 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.
<i>Value At Reference Position (Phys. Unit)</i> 0x16	Position value at the reference switch (physical unit) The current position is set to this value if the axis has performed a reference move to the reference switch (start with FRF). 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 mechanics does not have a reference switch.
<i>Distance From Negative Limit To Reference Position (Phys. Unit)</i>	Distance between reference switch and negative physical limit of the travel range (physical unit) If the axis has performed a reference move to the negative physical limit of the travel range (start with FNL), the current position is set to

Parameter	Description and Possible Values
0x17	the difference between the values of parameters 0x16 and 0x17.
Distance From Reference Position To Positive Limit (Phys. Unit) 0x2F	Distance between reference switch and positive physical limit of the travel range (physical unit) If the axis has performed a reference move to the positive physical limit of the travel range (start with FPL), the current position is set to the sum of the values of parameters 0x16 and 0x2F.
Maximum Travel In Negative Direction (Phys. Unit) 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 between the parameters 0x16 and 0x17), the negative physical limit of the travel range cannot be used for reference moves. The value can be negative.
Range Limit Min 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.
Range Limit Max 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-873 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 positioner type is selected from the positioner database.
- 0x07000000 (**Range Limit Min**) and 0x07000001 (**Range Limit Max**):
 - Using these limits is recommended only if open-loop motion is required. 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 positioner database and are set in the default settings so that the limits are deactivated.

Examples

The following examples refer to an axis of a positioner with incremental sensor, reference switch and hard stops or limit switches (the signals are supplied via digital input lines (p. 90)).

The distance between the negative and positive stop of the axis is 20 mm. The reference switch has a distance of 8 mm to the negative stop and a distance of 12 mm to the positive stop.

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

- Parameter 0x17: Distance between the negative end of the travel range and reference switch = 8 mm
- Parameter 0x2F: Distance between the reference switch and the positive end of the travel range = 12 mm

INFORMATION

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

Example 1: Maximum travel range available

After reference moves (p. 34), the current position is to have the following values:

- Moves to the negative end of the travel range (start with `FNL`): Current position = 0
- Move to the reference switch (start with `FRF`): current position = 8
- Moves to the positive end of the travel range (start with `FPL`): Current position = 20

For this reason, parameter 0x16, which specifies the position value at the reference switch for reference moves and is included in calculation of the position values at the travel range limits, 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

a) Travel range for positioners with hard stops

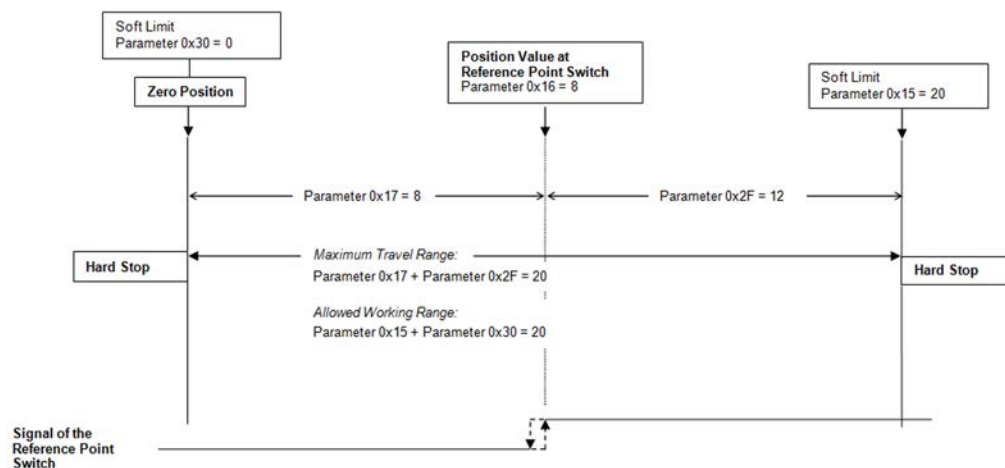


Figure 4: The travel range of the axis is not limited by soft limits.

b) Travel range for positioners with limit switch

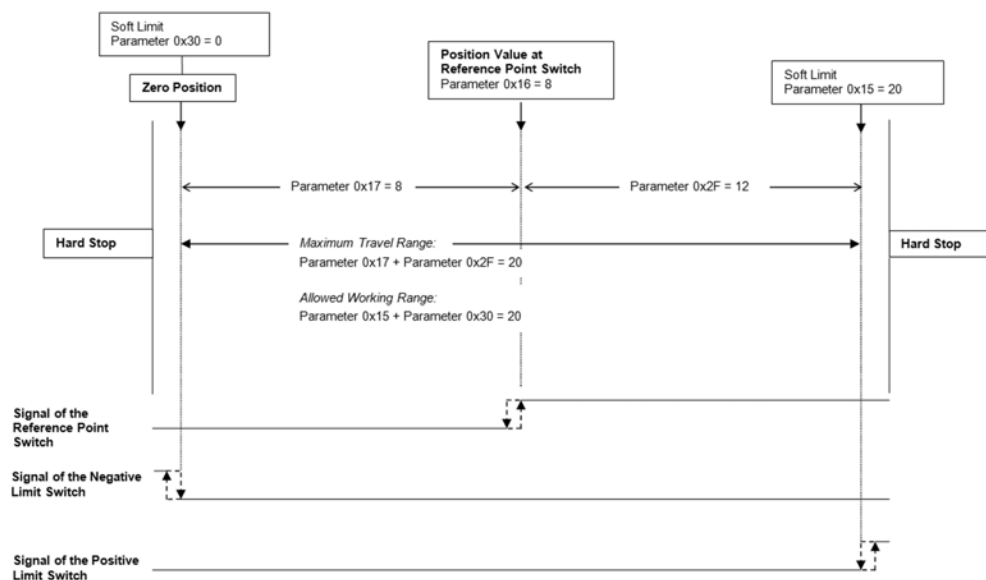


Figure 5: The travel range of the axis is not limited by soft limits.

After a reference move of the axis to the reference 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 end of the travel range and the reference 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

According to that, the axis can move 16.4 mm from the zero position in the positive direction and 2.1 mm in the negative direction respectively. The hard stops can no longer be used for reference moves.

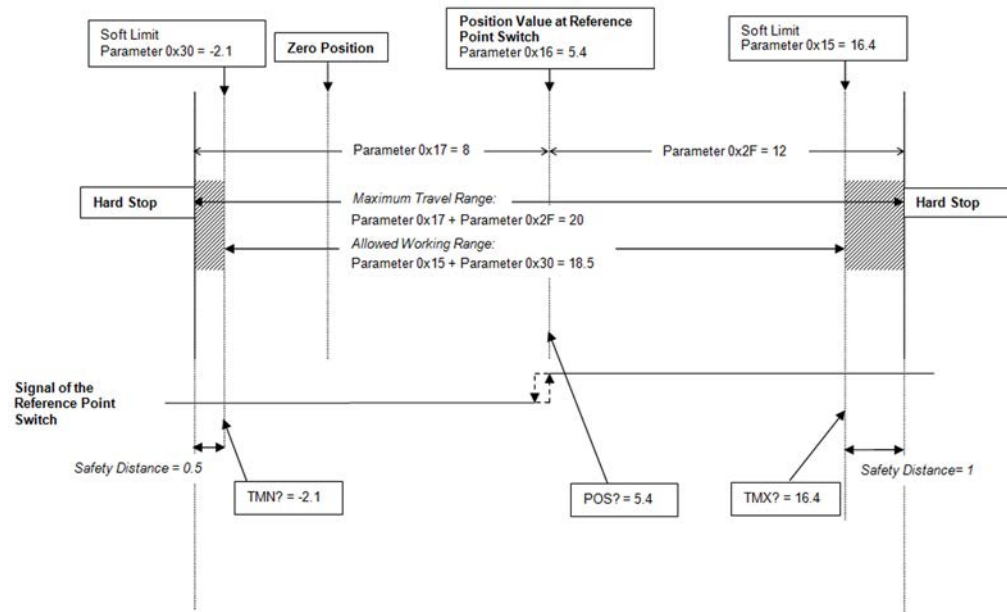
a) Travel range for positioners with hard stops

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

b) Travel range for positioners with limit switch

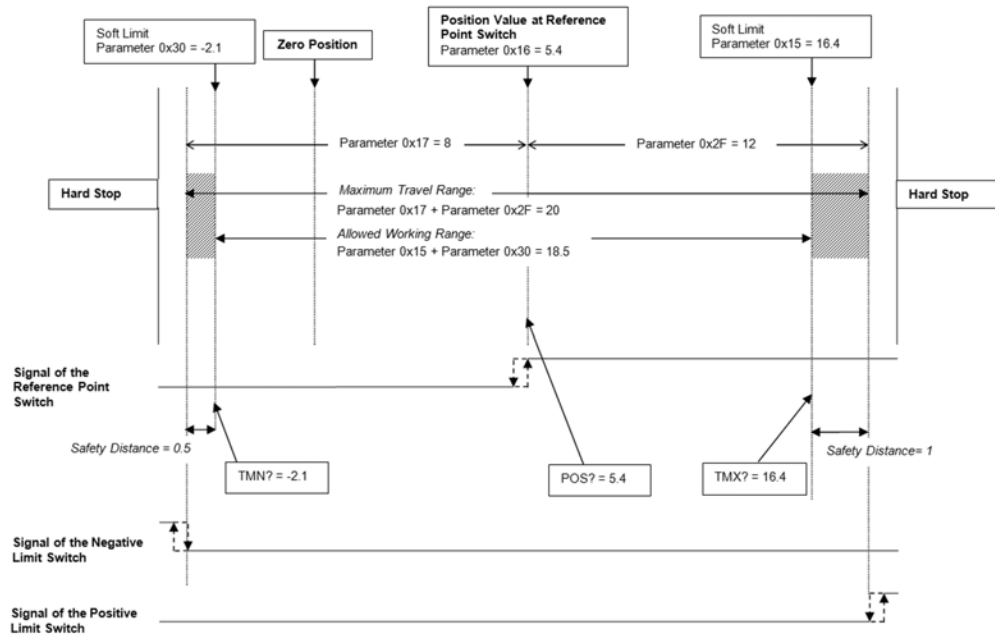


Figure 7: The travel range of the axis is limited by soft limits.

After a reference move of the axis to the reference 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

3.8.12 Referencing

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 when switching on. The axis must therefore be referenced beforehand so that absolute target positions can be commanded and reached.

Referencing can be done in different ways:

- **Reference move** (default): A reference move moves the axis to a defined point, e.g., to the reference switch or to a limit switch. At this point, the current position is set to a defined value. The controller now knows the absolute axis position.
- **Setting the absolute position manually:** If this referencing method was activated by the **RON** command (p. 184), you can set the current position of the axis to an arbitrary value at an arbitrary point using the **POS** command (p. 182). The axis is not moved here. The controller knows the absolute axis position afterwards.

INFORMATION

During startup using PIMikroMove, referencing is done via a reference move by default. Knowledge of the commands and parameters described here is not needed for referencing using PIMikroMove.

Commands

The following commands are available for referencing:

Command	Syntax	Function
RON	RON {<AxisID> <ReferenceOn>}	Specifies the referencing method: <ul style="list-style-type: none"> <ReferenceOn> = 0: An absolute position value can be assigned with POS to reference the axis or a reference move can be started with FRF, FNL or FPL. <ReferenceOn> = 1 (default): To reference the axis, a reference move must be started with FRF, FNL or FPL. Using POS is not permitted.
RON?	RON? [{<AxisID>}]	Queries the referencing method.
FRF	FRF [{<AxisID>}]	Starts a reference move to the reference switch. The approach depends on the value of the Reference Signal Type parameter (ID 0x70): <ul style="list-style-type: none"> 0 or 1: The approach is always done from the same side irrespective of the axis position when the command is sent. 2: The approach is done via the negative limit switch or hard stop. 3: The approach takes place via the positive limit switch or hard stop.
FRF?	FRF? [{<AxisID>}]	Queries whether the specified axis is already referenced. 1 = Axis is referenced 0 = Axis is not referenced
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 motion) and therefore references the axis.

Parameters

Reference moves can be configured with the following parameters:

Parameters	Description and Possible Values
Reference Travel Direction 0x47	Default direction for the reference move 0 = Automatic detection 1 = Negative direction 2 = Positive direction The E-873 is not equipped with a direction-sensing reference switch. The reference move therefore also takes place in a negative direction when the value 0 is set (default setting).
Distance Between Limit And Hard Stop (Phys. Unit) 0x63	Distance between the built-in limit switch and the hard stop Is used for positioners 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.
Distance From Limit To Start Of Ref Search (Phys. Unit) 0x78	Gap between the limit switch or hard stop and the start position for motion to the index pulse. For details, see explanation below the table.
Distance For Reference Search (Phys. Unit) 0x79	Maximum distance for motion to the index pulse For details, see explanation below the table.
Use Hard Stops For Referencing? 0x7A	Should the hard stops be used for reference moves? 0 = Do not use for reference moves 1 = Use for reference moves

The parameters 0x78 and 0x79 are used for reference moves when the two following conditions are met:

- The reference move is started with FRF.
- The **Reference Signal Type** parameter (0x70) has the value 2 or 3.

Sequence of the reference move:

1. The axis moves to the corresponding limit switch or hard stop.
2. The axis moves the distance away from the limit switch or hard stop as specified by parameter 0x78 .
3. The axis moves to the index pulse and covers the maximum distance specified by parameter 0x79.

INFORMATION

- For maximum repeatability, the reference move must always be done in the same way.

INFORMATION

The hard stops can only be used for reference if the travel range is not limited by soft limits (p. 29).

INFORMATION

For reference moves, you can also use the digital inputs of the E-873 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. 90) for more information.

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 referencing is not otherwise possible.
-

INFORMATION

If the current parameter settings of the E-873 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).

4 Unpacking

1. Unpack the E-873 with care.
2. Compare the contents with the scope of delivery according to the contract and the delivery note.
3. Inspect the contents for signs of damage. If any parts are damaged or missing, contact our customer service department (p. 257) immediately.
4. Keep all packaging materials in case the product needs to be returned.

5 Installation

In this Chapter

General Notes on Installation	41
Ensuring Ventilation.....	41
Mounting the E-873	41
Connecting the E-873 to the Protective Earth Conductor	43
Connecting the Power Supply to the E-873	44
Connecting the Positioner	44
Connecting an HID	45
Connecting Digital Inputs and Outputs.....	45
Installing the PC Software.....	47
Connecting the PC.....	51

5.1 General Notes on Installation

- Install the E-873 near the power source so that the power plug can be quickly and easily disconnected from the mains.
- Only use cables and connectors that meet local safety regulations.

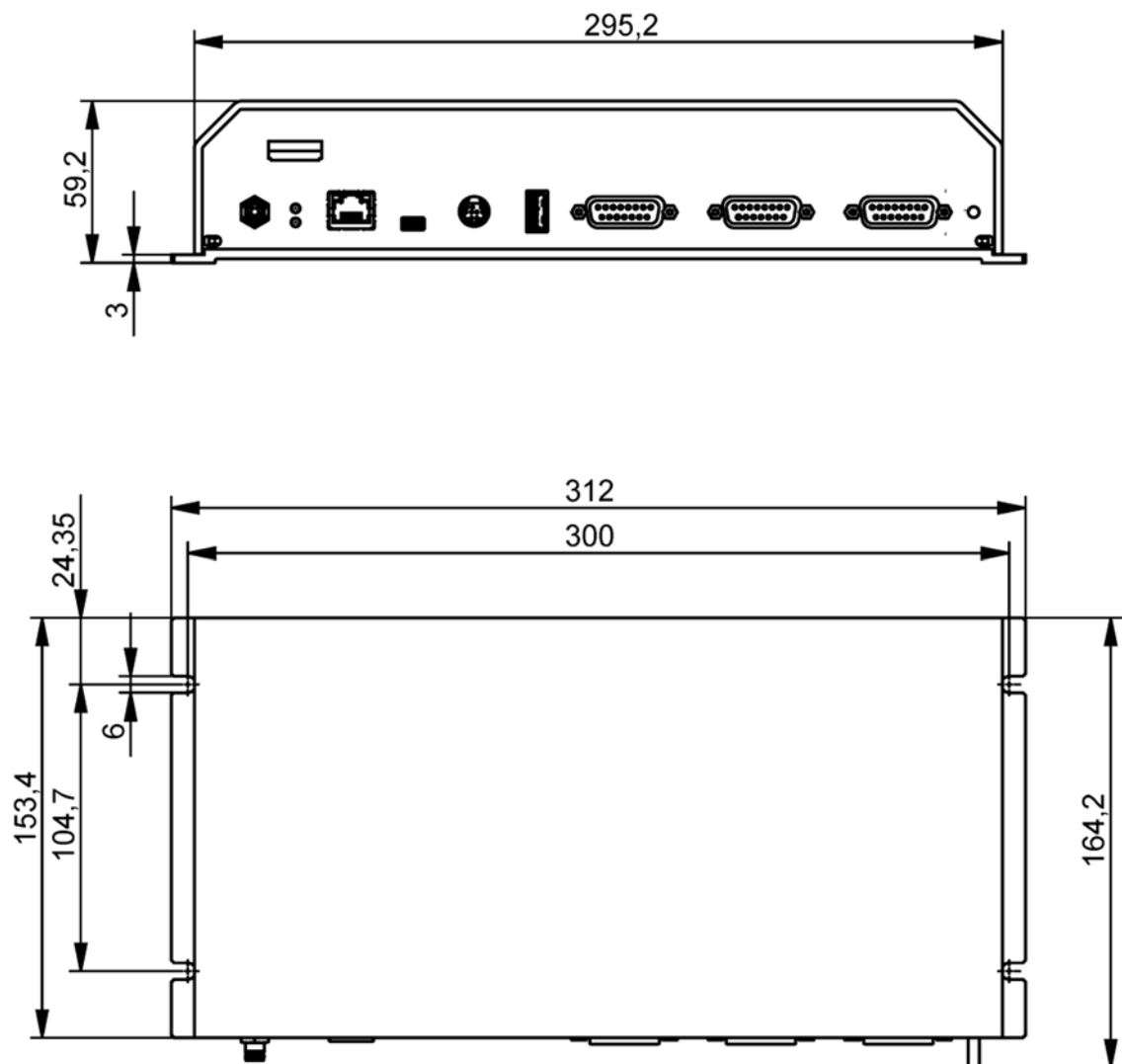
5.2 Ensuring Ventilation

High temperatures can overheat the E-873.

- Set up the E-873 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).

5.3 Mounting the E-873

The E-873 can be used as benchtop device or mounted in any orientation on a surface.



Tools and accessories

- Suitable screws
- Suitable screwdriver

Mounting the E-873.3QTU

1. Make the necessary holes in the surface.

The arrangement of the recesses in the mounting rails of the E-873 can be found in the figure.

2. Use two screws on each side to affix the E-873 to the recesses in the mounting rails.

5.4 Connecting the E-873 to the Protective Earth Conductor

The E-873 must be connected to a protective earth conductor because it is not grounded via the power adapter connection.

INFORMATION

- Pay attention to the applicable standards for connecting the protective earth conductor.


Requirements

- ✓ You have read and understood the general notes on installation (p. 41).
- ✓ The E-873 is not connected to the power supply.

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 attaching the protective earth conductor
- Mounting hardware for the protective earth conductor, sits on the protective earth connector (threaded bolt) in the following order on delivery of the E-873, starting from the housing:
 - Safety washer
 - Nut
 - Flat washer
 - Toothed washer
 - Nut
- Suitable wrench

Connecting the E-873 to the protective earth conductor

1. If necessary, attach a suitable cable lug to the protective earth conductor.
2. Remove the outer nut from the protective earth connector on the front panel of the E-873 (threaded bolt (p. 7) marked with .
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 attached to the protective earth conductor is wedged between the toothed washer and the nut.
 - c) Tighten the nut with at least three turns and a torque of 1.2 Nm to 1.5 Nm.

5.5 Connecting the Power Supply to the E-873

Requirements

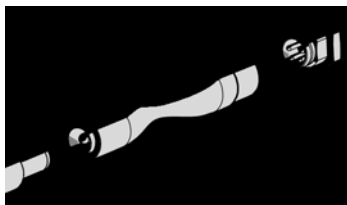
- ✓ The power cord is **not** connected to the power socket.

Tools and accessories

- 24-V wide-range-input power supply included (for line voltages between 100 and 240 V alternating current at 50 or 60 Hz)
Alternative: Suitable power supply that supplies 24 V direct current with a maximum of 2.5 A
- Adapter included for the power supply connection; barrel connector to M8 4-pin connector (f)
Alternative: Suitable adapter
- Power cord included
Alternative: Suitable power cord

Connecting the power supply to the E-873

- Connect the M8 connector (f) of the adapter to the 24 V connection (**24 V DC 4.5 A**) of the E-873.
- Connect the barrel connector of the adapter to the barrel connector socket of the power supply.



- Connect the power cord to the power supply.

5.6 Connecting the Positioner

NOTICE



Damage if a wrong motor is connected!

Connecting an unsuitable positioner to the E-873 can cause irreparable damage to the positioner or controller.

- Only connect the E-873 to positioners with Q-Motion® piezo inertia drive or PiezoMike linear actuators.

Requirements

- ✓ The E-873 is switched off, i.e., the power supply is **not** connected to the power socket via the power cord or the toggle switch on the front panel is at the **O** position.
- ✓ You have read and understood the user manual of the stage(s).

Tools and accessories

- Positioners with Q-Motion® inertia drive or PiezoMike linear actuators
- Optional: Suitable extension cable from PI
- Suitable screwdriver

Connecting the Positioner

1. Connect the positioner to one of the sockets **Axis 1** to **Axis 3** of the E-873.
2. Use the integrated screws to secure the connections against accidental disconnection.

5.7 Connecting an HID

Tools and accessories

- Suitable human interface device with type A USB connection, e.g.: Joystick or gamepad

Connecting a Human Interface Device

- Connect the human interface device directly to the USB **Joystick** connection of the E-873.

5.8 Connecting Digital Inputs and Outputs

The digital inputs and outputs on the **I/O** socket of the E-873 can be used as follows:

- Outputs: Triggering of external devices; see "Digital Output Signals" (p. 80).
- Inputs: Use in macros (p. 89) and/or as source for the reference and limit switch signals of the axis (p. 90)

5.8.1 Connecting the Digital 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. 10)
- Device to be triggered having digital input for TTL signals

Connecting a device to be triggered

- Connect an appropriate device to one of pins 5, 6, 7, and 8 of the **I/O** socket of the E-873.

5.8.2 Connecting the Digital Inputs**INFORMATION**

Digital input signals can be fed via pins 1, 2, 3, and 4 of the **I/O** socket into the E-873.

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. 10).
 - If the digital inputs are to be used as the source for the reference and limit switch signals of an 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.

Connecting a digital signal source

- Connect a suitable signal source to one of pins 1, 2, 3, or 4 of the **I/O** socket of the E-873.

5.9 Installing the PC Software

5.9.1 Initial Installation

Accessories

- PC with Windows (7, 8, 10; 32 bit, 64 bit) or Linux operating system and at least 30 MB free memory
- Product CD (included in the scope of delivery)

Installing the PC software in Windows

1. Start the installation wizard by double-clicking the **PI_E-873.CD_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-873 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:

- Driver for use with NI LabVIEW software
Exception: The *Analog LabVIEW drivers* component is provided for some PI controllers. This component is only available through user-defined installation.
- Dynamic program library for GCS
- PIMikroMove
- PC software for updating the firmware of the E-873
- PI Update Finder for updating the PC software
- For controllers that have a USB interface for communication with the PC: USB drivers

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

Installing the PC software in 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 in as superuser (root privileges).
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.

5.9.2 Installing Updates

PI is constantly improving the PC software.

- Always install the latest version of the PC software and the positioner database.

Requirements

- ✓ Active connection to the Internet.
- ✓ If your PC uses a Windows operating system:
 - You have installed (p. 47) the PI Update Finder from the product CD.
 - You have the A000T0028 Technical Note for the PI Update Finder ready. You will find the document on the product CD.
 - If the PC to be updated is **not** directly connected to the Internet: You have the A000T0032 Technical Note for the PI Update Finder ready. You will find the document on the product CD.
- ✓ If your PC uses a Linux operating system:
 - You have the access data (user name and password) for the E-873. Information regarding the access data can be found in the file "E-873_Releasenews_V_x_x_x.pdf" (x_x_x: Version number of the CD) in the \Manuals folder on the product CD.

Updating the PC software and PISTages2.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 in Linux

1. Open the website www.pi.ws.
2. Click **Login**.
3. Log in with the user name and password for the E-873.
4. Click **Search**.
5. Enter the product number up to the period (e. g., E-873) into the search field.
6. Click **Start search** or press the Enter key.
7. Open the corresponding product detail page in the list of search results:

- a) If necessary: Scroll down the list.
- b) If necessary: Click **Load more results** at the bottom of the list.
- c) Click the corresponding product in the list.
8. Click the **Downloads** tab.
The software files are shown under **Software Downloads**.
9. Click the archive file "CD Mirror" or the associated download link.
10. Select the option in the following request to save the file to your PC.
If you do not specify anything else, the "CD Mirror" archive file is stored in the default download directory of your PC.
11. Unpack the archive file into a separate installation directory.
12. Go to the **linux** subdirectory in the directory with the unpacked files.
13. Unpack the archive file in the **linux** directory by entering the command `tar -xvpf <name of the archive file>` on the console.
14. Read the accompanying information on the software update (readme file and/or "E-873_Releasenews_V_x_x_x.pdf" file) and decide whether the update makes sense for your application.
 - If no: Stop the update procedure.
 - If yes: Go through the following steps.
15. Log into the PC as superuser (root privileges).
16. Install the update.

INFORMATION

If software is missing in the **Downloads** area or problems occur with downloading:

- Contact our customer service department (p. 257).

Updating PIStages2.dat and PIMicosStages2.dat in 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-873_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 positioner database - **pistages2** or **pimicosstages2** - or the **Download** button below it.
8. Log into the PC as superuser (root privileges).

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 /<UnpackingDirectory>/pi_stages2_dat or /<UnpackingDirectory>/pimicosstages2_dat. In this subdirectory start the INSTALL.pi_stages2_dat or INSTALL.pimicosstages2_dat script.

5.9.3 Installing a Custom Positioner Database

With a custom positioner, you will receive, if necessary, a file from PI with a custom positioner database. You have to install this file on your PC so that you can load the parameter values for the custom positioner in the E-873.

Installing a custom positioner database on Windows

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

If you work with PIMikroMove:

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

The path in which the \PI directory is located was determined during the installation of the PC software, normally C:\ProgramData.

2. Copy the positioner 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 positioner database, both files have to be in the same directory.

Installing a custom positioner database on Linux

1. Log into the PC as superuser (root privileges).
2. Copy the positioner database file to the /usr/local/PI/pi_gcs_translator/ directory.

5.10 Connecting the PC

Communication between the E-873 and a PC is required to configure the E-873 and to command motions using the GCS commands.

The E-873 has the following interfaces for this purpose:

- USB interface
- TCP/IP interface

The E-873 may only be connected to the PC via one of these interfaces.

In this section, you learn how to establish corresponding cable connections between the E-873 and a PC as well as in a TCP/IP network.

The steps for establishing communication between E-873 and PC are described in the section "Startup":

- Establishing communication via the USB interface (p. 54)
- Establishing communication via the TCP/IP interface (p. 56)

5.10.1 Connecting the E-873 via the USB interface

Requirements

- ✓ The PC has a free USB interface.

Tools and accessories

- USB cable (type A to Mini-B) for connecting to the PC (000036360 in the scope of delivery)

Connecting the E-873 to the PC

- Connect the USB socket of the E-873 and the USB interface of the PC with the USB cable.

5.10.2 Connecting the E-873 via the TCP/IP Interface

Prerequisites

- ✓ If the E-873 is to be directly connected to the PC:
The PC has a free RJ45 Ethernet connection socket.
- ✓ If the E-873 and a PC are to be operated together in a network:
A free access point to the network is available for the E-873; a suitable hub or switch is connected to the network for this purpose if necessary.

Tools and accessories

- If the E-873 is to be directly connected to the PC:
crossover network cable (C-815.563 in the scope of delivery)
- If the E-873 is to be connected to a network access point:
straight-through network cable (C-815.553 in the scope of delivery)

Connecting the E-873 directly to the PC

- Connect the RJ45 socket on the front panel of the E-873 to the RJ45 Ethernet connection socket of the PC via the included crossover network cable.

Connecting the E-873 to the network in which the PC is also located

- Connect the RJ45 socket on the front panel of the E-873 with the network access point via the included straight-through network cable.

6 Startup

In this Chapter

General Notes on Startup	53
Switching on the E-873	53
Establishing Communication.....	54
Starting Motion.....	62
Setting the Notch Filter.....	66
Optimizing the Servo Control Parameters.....	71

6.1 General Notes on Startup

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-873 in the case of malfunction or failure of the system. If there are touch voltages, touching the E-873 can result in minor injuries from electric shock.

- Connect the E-873 to a protective earth conductor (p. 43) before startup.
- 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-873 to the protective earth conductor before restarting.

6.2 Switching on the E-873

INFORMATION

The E-873 is intended for closed-loop operation with incremental position sensors (servo mode On). After switch-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 startup macro that starts the E-873 via the `SVO` command in closed-loop operation; see "Setting up a startup macro" (p. 107).

INFORMATION

The ID chip is not read when you connect the positioner while the E-873 is switched on.

- After connecting a positioner, reboot the E-873 with the **RBT** (p. 183) command or with the corresponding PC software functions in order to read the data from the ID chip.

Requirements

- ✓ You have read and understood the general notes on startup (p. 53).
- ✓ The E-873 has been installed properly (p. 41).

Switching the E-873 on

1. Connect the power cord of the power adapter to the power socket.
2. Switch the controller on by pushing the toggle switch on the front panel of the device to the — position.

The E-873 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 positioner
3. Wait until the status LED lights up green.
- The **Status** LED on the front panel of the device indicates the status of the E-873:
- green: E-873 is ready for normal operation
 - off: The E-873 is not connected to the power supply or could be defective
- If the E-873 is connected to the power adapter (p. 44) properly and the **Status** LED does not light up after switching on, contact our customer service department (p. 257).

6.3 Establishing Communication

The procedure for PIMikroMove is described in the following.

6.3.1 Establishing Communication via the USB Interface

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.

Requirements

- ✓ You have read and understood the general notes on startup (p. 53).

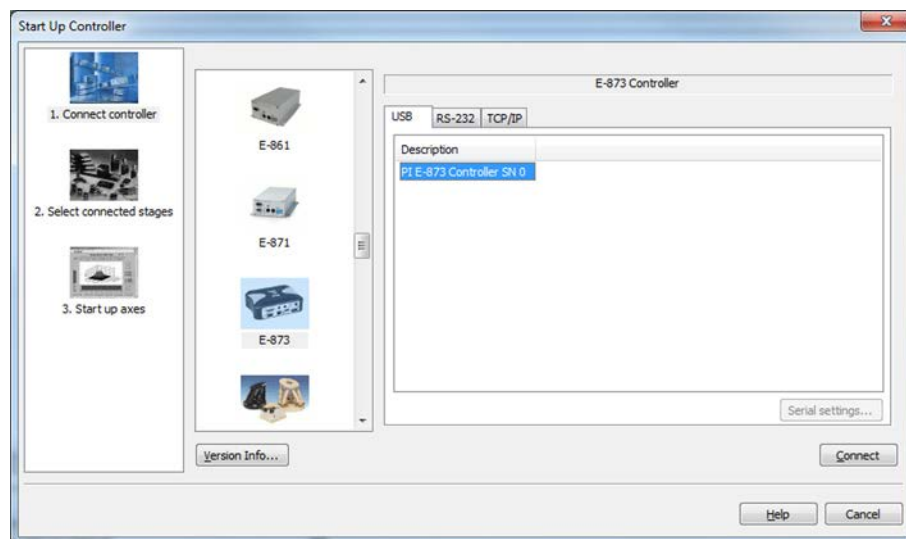
- ✓ The E-873 is connected to the USB interface of the PC (p. 51).
- ✓ The E-873 is switched on (p. 53).
- ✓ The PC is switched on.
- ✓ The required software and USB drivers are installed on the PC (p. 47).
- ✓ You have read and understood the manual of the PC software used. The software manuals are on the product CD.

Establishing communication via USB

1. Start PIMikroMove.

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

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



2. Select **E-873** in the field for controller selection.
3. Select the **USB** tab on the right-hand side of the window.
4. On the **USB** tab, select the **E-873** connected.
5. Click **Connect** to establish communication.

If communication was established successfully, PIMikroMove guides you through the configuration of the E-873 for the connected positioner; see "Starting Motion" (p. 62).

- If communication could not be established, look for a solution to the problem in "Troubleshooting" (p. 253).

6.3.2 Establishing Communication via the TCP/IP Interface

Adapting the interface parameters for communication via TCP/IP

Before communication is established, it can be necessary to adapt the interface parameters once, depending on the type of networking:

- **Network with DHCP server:** No adjustment of the factory settings of the E-873 interface parameters is necessary
- **Network without DHCP server or direct connection** (E-873 directly connected to the Ethernet connection socket of the PC):
 - The startup behavior of the E-873 for configuring the IP address must be changed so that the E-873 uses a static IP address.
 - IP addresses and subnet mask of the E-873 and PC must be compatible with each other.

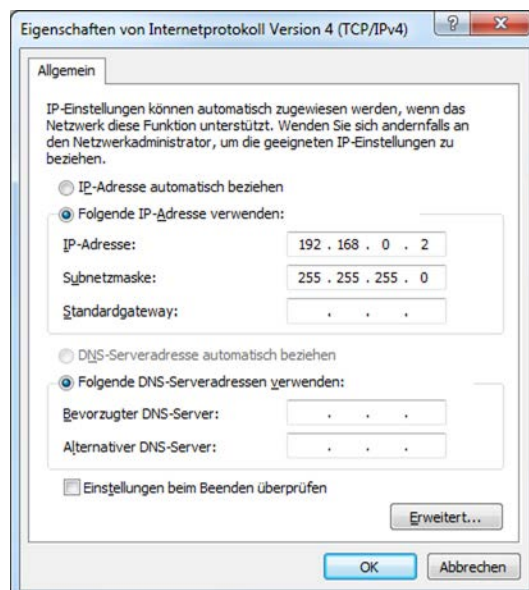
Requirements

- ✓ You have established communication between the E-873 and the PC via USB (p. 54) in PIMikroMoveto determine the settings of the E-873 and to change them if necessary.

Determining the IP address and subnet mask of the PC

1. Open the window on your PC so that the Internet Protocol properties can be displayed and set. The necessary steps depend on the operating system used.

If your operating system distinguishes between Internet Protocol version 4 (TCP/IPv4) and version 6 (TCP/IPv6), open the window for version 4.

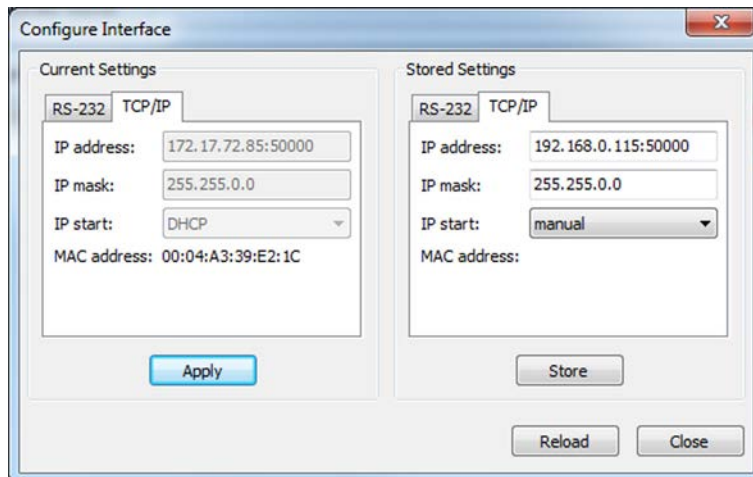


The figure shows example settings that may not apply to your system.

2. Write down the settings for the IP address and the subnet mask.

Determine the IP address and subnet mask of the E-873 and adapt the starting behavior

1. In the main window, open the **Configure Interface** window in PIMikroMove by selecting the **E-873 > Configure interface** menu item.
2. In the **Configure Interface** window, select the **TCP/IP** tab in the **Stored Settings** area.



The figure shows example settings that may not apply to your system.

3. Write down the values from the following fields of the **TCP/IP** tab in the **Stored Settings** area:
 - **IP address**
 - **IP mask**
4. On the **TCP/IP** tab, select the *manual* value in the **IP start** field in the **Stored Settings** area.
This changes the startup behavior of the E-873 so that it uses a static IP address.
5. Save the changed setting of the startup behavior in the nonvolatile memory of the E-873:
 - a) Click the **Store** button in the **Stored Settings** area. The **Store interface settings** dialog opens.
 - b) In the **Store interface settings** dialog, click **Store settings**. The dialog closes.

Adapting IP settings of the PC

- If the PC is already configured for using a static IP address and you want to leave the PC settings unchanged, continue with the section "Adapting the IP settings of the E-873".
1. Activate **Use following IP address** in the window, in which the TCP/IP properties (TCP/IPv4) are displayed and set (see "Determining the IP Address and Subnet Mask of the PC" (p. 56)).
 2. Adapt the IP address and the subnet mask to the settings of the E-873:

- a) Copy the subnet mask of the E-873 for the subnet mask on the PC.
- b) Copy the first three sections of the IP address of the E-873 (see "Determine the IP address and subnet mask of the E-873 and adapt the starting behavior" (p. 57)).
- c) Make sure that the last section of the IP address on the PC differs from the last section of the IP address of the E-873 and is not "255" or "0".

Example:

If the E-873 has the subnet mask 255.255.255.0 (IPMASK = 24), set the subnet mask 255.255.255.0 on the PC.

If the E-873 has the IP address 192.168.0.75, set the IP address 192.168.0.76 on the PC.

3. Confirm the settings with the **OK** button.
4. If further network devices have to be adapted:
Adapt the IP addresses and subnet masks as in the previous steps.
Assign a separate, unique IP address to each network device.
No IP address may occur twice in the same network.
5. Close the connection of the E-873 via the USB interface:
 - a) Close the **Configure Interface** window by clicking **Close**.
 - b) Select the **Connections > Close > E-873** menu item in the main window of PIMikroMove.
6. Switch off the E-873.
7. Continue with the section "Establishing Communication via TCP/IP in the PC Software" (p. 59).

Adapting the IP settings of the E-873

1. Adapt the settings of the E-873 to settings of the PC (see "Determining the IP Address and Subnet Mask of the PC" (p. 56)). To do this make the following settings on the **TCP/IP** tab in the **Stored Settings** area in the **Configure Interface** window of PIMikroMove.
 - a) Change the subnet mask in the **IP mask** field to the subnet mask of the PC.
 - b) Change the IP address (format: xxx.xxx.xxx.yyy) in the **IP address** field, whereby the following applies:
 - xxx.xxx.xxx. matches the first three sections of the IP address of the PC.
 - yyy differs from the last section of the IP address of the PC and every other device in the same network.
 - yyy is not "255" and not "0" and is in the address range which is given by the last section of the subnet mask.
 - The port number "50000" may not be changed.

Example:

If the IP address of the PC is 192.168.0.2 and no other device has the IP address 192.168.0.3, set the IP address 192.168.0.3:50000.

2. Save the changed settings in the nonvolatile memory of the E-873:
 - a) Click the **Store** button in the **Stored Settings** area. The **Store interface settings** dialog opens.
 - b) In the **Store interface settings** dialog, click **Store settings**. The dialog closes.
3. Close the **Configure Interface** window by clicking **Close**.
4. Close the connection of the E-873 via the USB interface by selecting the **Connections > Close > E-873** menu item in PIMikroMove in the main window.
5. Switch off the E-873.
6. Continue with the section "Establishing Communication via TCP/IP in the PC Software" (p. 59).

Establishing Communication via TCP/IP in the PC Software

INFORMATION

Communication via TCP/IP can fail if the network cable was connected to the Ethernet socket of the E-873 while the E-873 was switched on.

- If communication cannot be established, switch the E-873 off. Now reconnect the network cable and switch the E-873 on again.

INFORMATION

- For communication via TCP/IP, the E-873 only has one unchangeable port (50000) available that cannot be used for more than one connection at a time.

INFORMATION

The E-873 is displayed with its model name followed by a 9-digit serial number in the list of controllers found in the same network.

Example of the display of a E-873 to which no connection has been established via TCP/IP yet:

E-873.3QTU SN 123456789 -- listening on port 50000 -- 172.17.72.55 50000

123456789 is the serial number of the controller in this example.

- If several E-873 are connected with the same network via TCP/IP, identify the E-873 to be connected in the list of found controllers on the basis of its serial number. The serial number of the controller is on the type plate of the E-873.
- If you establish initial communication via USB (p. 54) (e.g., with PIMikroMove or PITerminal), you can get the serial number of the E-873 in the response to the ***IDN?** (p. 124) command.

Requirements

- ✓ You have read and understood the general notes on startup (p. 53).
- ✓ The E-873 is connected to the network or directly to the PC via its RJ45 Ethernet socket (p. 51).

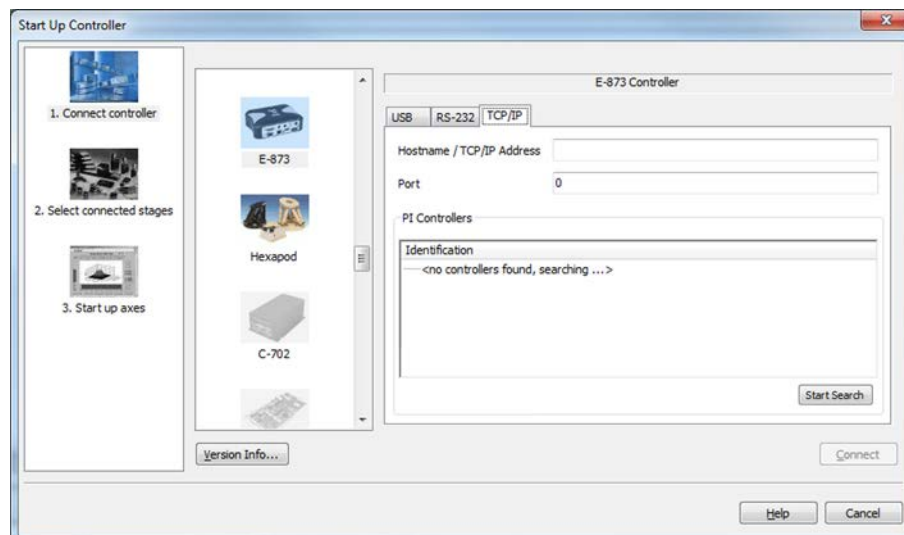
- ✓ If the E-873 is connected to a network:
The PC to be used for communication with the E-873 is appropriately connected to the same network as the E-873.
- ✓ If the used network does not have a DHCP server or if the E-873 is directly connected to the Ethernet connection socket of the PC:
By adapting the interface parameters, you have set the correct startup behavior for configuring the IP address of the E-873 and adapted the IP addresses and subnet masks of the E-873 and PC as well as all other network devices to each other. See "Preparing the PC and E-873 for a Network without a DHCP Server or for Direct Connection" (p. 56).
- ✓ If several E-873s are connected to the same network via their TCP/IP interfaces: You have the serial number of the E-873 ready with which the communication is to be established. The serial number can be found on the type plate (p. 9) of the E-873.
- ✓ The E-873 is switched on.
- ✓ The PC is switched on.
- ✓ The required software is installed on the PC (p. 47).
- ✓ You have read and understood the manual for the PC software used. The software manuals are on the product CD.

Establishing communication via TCP/IP

1. Start PIMikroMove.

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

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



2. Select **E-873** in the controller selection field.
3. Select the **TCP/IP** tab on the right-hand side of the window.

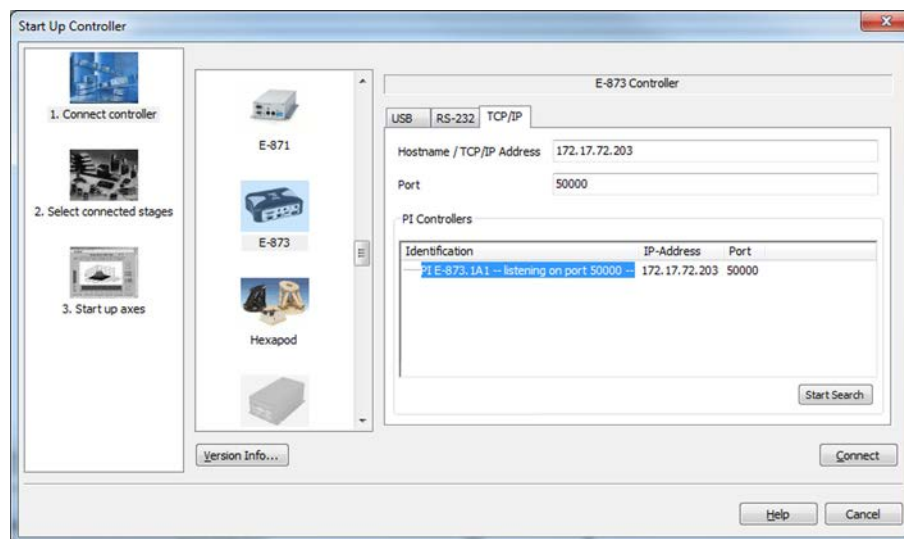
The software now searches the network for all controllers of the E-873 type.

Click **Start Search** if the search for \$\$\$\$ type controllers does not start automatically.

Searching the network for controllers of the \$\$\$\$ type has started. As long as the search is running, the **Connect** button is deactivated. If the search was successful, all \$\$\$\$ controllers in the same network are displayed in the **PI Controllers** field.

4. Click the entry of your E-873 model found in the list of controllers. This must show the status "listening on port 50000".
 - If several entries with the same name are shown, identify your E-873 on the basis of its nine-digit serial number.
 - If the E-873 is not displayed in the list of the controllers found, check the network settings (p. 253). Consult your network administrator if necessary.
 - Do **not** select a controller that is already connected via TCP/IP (status "connected to ..."). Otherwise, an error message will be displayed as soon as you want to establish communication with this controller.

After a controller is selected in the list, its data is shown in the **Hostname / TCP/IP Address** and **Port** fields.



5. Click the **Connect** button to establish communication.

If communication was established successfully, PIMikroMove guides you through the configuration of the E-873 for the connected positioner, see "Starting Motion" (p. 62).

- If communication could not be established, look for a solution to the problem in "Troubleshooting" (p. 253).

6.4 Starting Motion

PIMikroMove is used in the following to move the positioner. 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-873 for the connected positioner
- Switching servo mode on
- Refer to "Referencing" (p. 34) for details on doing a reference move.

It is then possible to run the first motion tests for the positioner.

NOTICE



Selecting an incorrect positioner type

Selecting an incorrect positioner type in the PC software can damage the positioner.

- Make sure that the type of positioner selected in the PC software matches the positioner that is connected.

NOTICE



Oscillation!

Unsuitable setting of the E-873's servo control parameters can cause the positioner to oscillate. Oscillation can damage the positioner and/or the load fixed to it.

- Secure the positioner and all loads adequately.
- If the positioner is oscillating (unusual operating noise), immediately switch off the servo mode or disconnect the E-873 from the power source.
- Only switch on the servo mode after you have modified the servo control parameter settings of the E-873; see "Optimizing Servo Control Parameters" (p. 71).
- If, due to a very high load, oscillation occurs during the reference move, follow the instructions for the reference move in "Troubleshooting" (p. 253).

INFORMATION

After communication has been established between the E-873 and the PC, PIMikroMove guides you through the configuration of the E-873 for the connected positioner. Selection of the configuration steps offered by PIMikroMove is based on evaluation of the following parameter values in the volatile memory of the E-873:

- **Stage Name** (ID 0x3C): The value is used by PIMikroMove as a criterion for finding a suitable parameter set in the positioner databases.
- **Stage Type** (ID 0x0F000100): The value was loaded from the ID-Chip (p. 13) of the connected positioner when the E-873 is switched on.

Possible configuration steps:

- When the values of the parameters 0x3C and 0x0F000100 are identical, PIMikroMove assumes that all parameters of the E-873 have already been adapted to the connected positioner. 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 positioner database to the E-873. After the parameter set has been loaded, the **Start up controller** window goes to the **Start up axes step**. If a matching parameter set is not in the positioner databases, a corresponding notice will appear in the **Stage Type Configuration** window.
- If the value of the parameter 0x0F000100 is empty because the positioner does not have an ID chip, for example, the **Start up controller** window will go to the **Select connected stages** step.

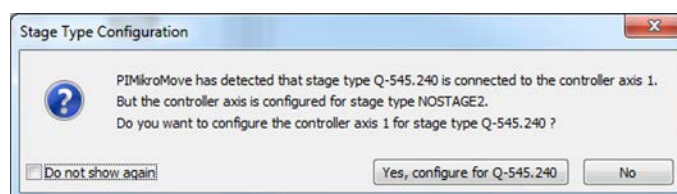
Requirements

- ✓ You have read and understood the general notes on startup (p. 53).
- ✓ PIMikroMove is installed on the PC (p. 47).
- ✓ You have read and understood the PIMikroMove manual. The manual is on the product CD.
- ✓ You have installed the latest version of the positioner database(s) onto your PC (p. 48).
- ✓ If PI has provided you with a custom positioner database for your positioner, then you have installed this database on your PC (p. 50).
- ✓ You have installed the positioner in the same way as it will be used in your application (corresponding load, orientation, and mounting).
- ✓ You have connected the E-873 to the positioner (p. 51).
- ✓ You have established communication with PIMikroMove between the E-873 and the PC (p. 54).

Starting motion with PIMikroMove

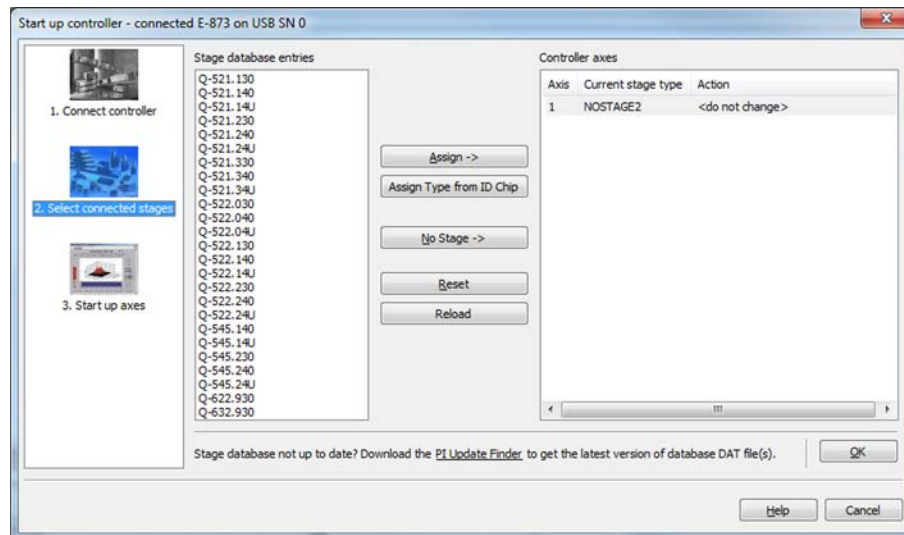
1. If one of the two following points applies, configure the E-873 for the connected positioner:
 - The **Stage Type Configuration** dialog has opened.
 - The **Select connected stages** step is displayed in the **Start up controller** window.

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



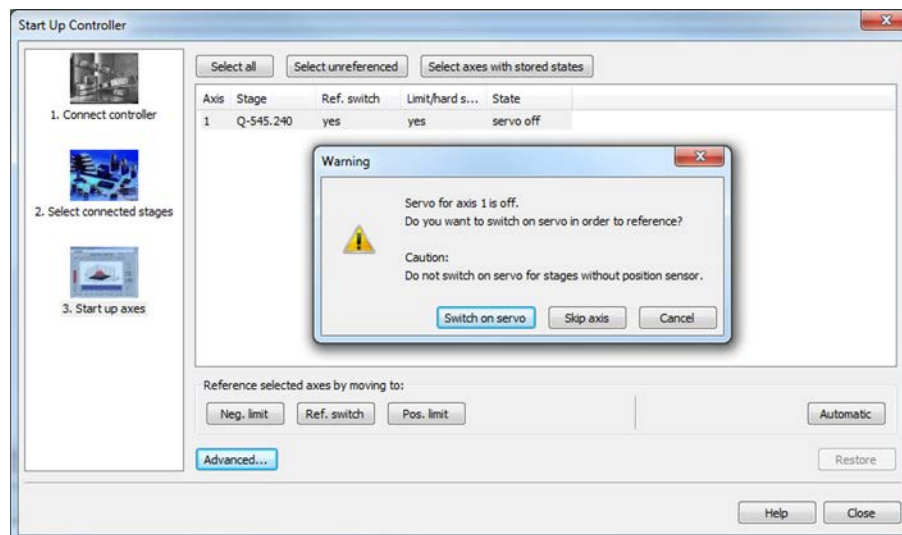
- Click the **Yes, configure for ...** button to load the appropriate parameter set from the positioner database into the E-873. This opens the **Save all changes permanently?** dialog.

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



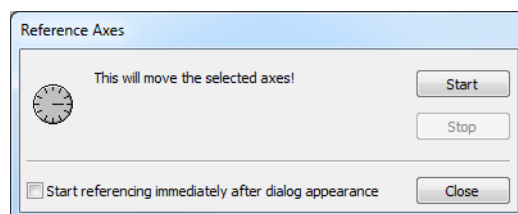
- a) Select the matching positioner type. You have two options:
 - Click **Assign Type from ID Chip**.
 - Mark the appropriate positioner type in the **Stage database entries** list and click **Assign**.
 - b) Confirm selection with **OK** to load the parameter settings for the selected positioner type from the positioner database into the E-873. This opens the **Save all changes permanently?** dialog.
 2. Specify how you want to load the parameter settings into the E-873 in the **Save all changes permanently?** dialog box:
 - Temporary load: Click **Keep the changes temporarily** to load the parameter settings into the volatile memory of the E-873. The settings are lost when the E-873 is switched off or rebooted.
 - Load as default values: Click **Save all settings permanently on controller** to load the parameter settings into the nonvolatile memory of the E-873. The settings are available immediately after switching on or rebooting the E-873 and do not need to be reloaded.
- The **Start up controller** window changes to the **Start up axes** step.
3. During the **Start up axes** step, do a reference move for the axis so that the controller knows the absolute axis position: You have the following options (options not supported by the positioner/controller either do not exist or cannot be activated):
 - If you want to start the reference move to the reference switch, click **Ref. switch**.
 - If you want to start the reference move to the negative limit switch, click **Neg. limit**.
 - If you want to start the reference move to the positive limit switch, click **Pos. limit**.

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



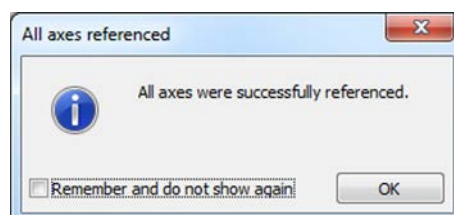
a) Switch on the servo mode by clicking on the **Switch on servo** button.

If the **Reference Axes** dialog is displayed after switching on the servo:



b) Click the **Start** button. The axis performs the reference move.

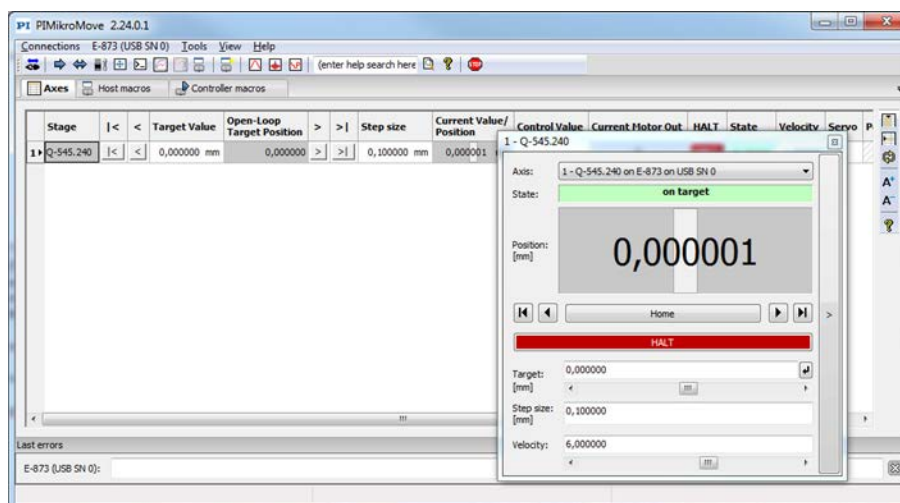
If the corresponding message is displayed after a successful reference move:



c) Close the message with **OK**.

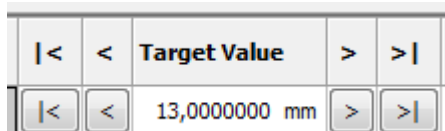
4. After a successful reference move, close the **Start up controller** window by clicking **Close**.

The main window of PIMikroMove opens.



5. Test the motion of the axis several times.

By clicking the corresponding arrow keys for the axis in the main window of PIMikroMove for example, it is possible to initiate motion over a particular distance (specification in **Step size** column) or to the limits of the travel range.



6.5 Setting the Notch Filter

The notch filter corrects the control value for the drive of the positioner connected to the E-873. The corrections by the notch filter take place in closed-loop and open-loop operation.

The frequency component that would cause natural oscillation of the mechanics in the control value is reduced by the notch filter. Adjusting the notch filter frequency can be useful, particularly in the case of 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. 71).

To set the notch filter, a step response is recorded in open-loop operation. Adapting the notch filter is done via the following parameters:

- Notch Filter Frequency 1 (0x94)
- Notch Filter Edge 1 (0x95)

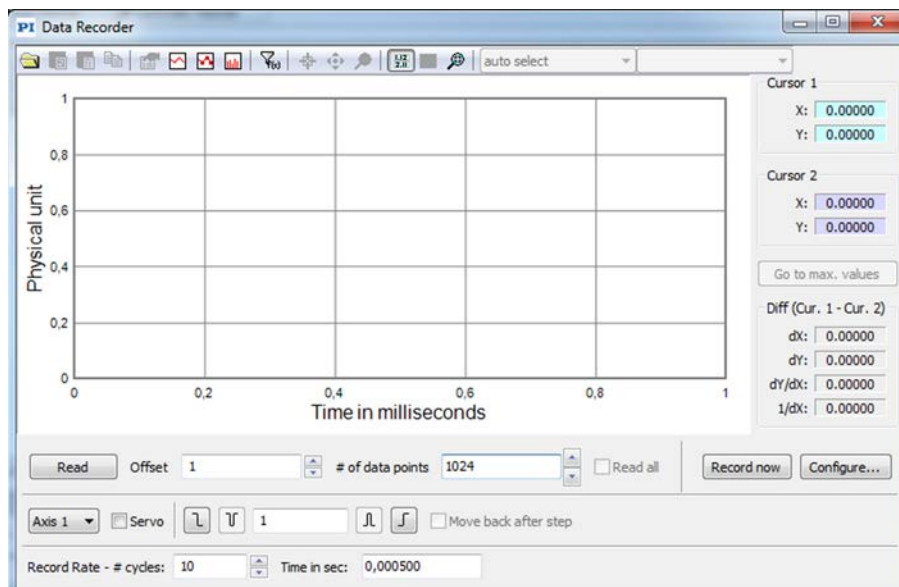
The procedure for PIMikroMove is described in the following.

Requirements

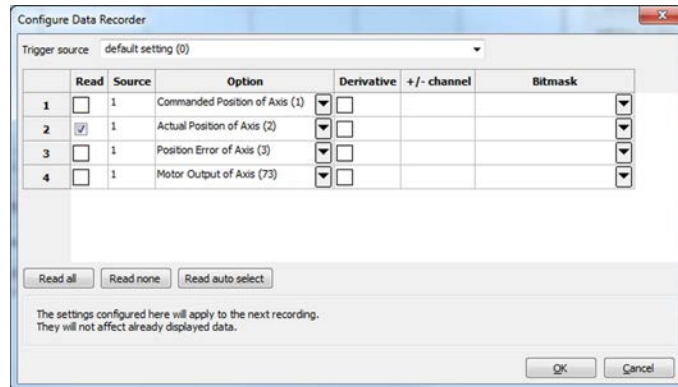
- ✓ You have installed the positioner in the same way as it will be used in your application (corresponding load, orientation, and mounting).
- ✓ Initial motion in the PIMikroMove was started (p. 62).
- ✓ All devices are still ready for operation.

Setting the notch filter


1. In the main window of PIMikroMove, open the **Data Recorder** window via the **E-873 > Show 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 10 for the record table rate in the **Record Rate** field.
 - c) Set 1024 as the value for the number of data points to be read for the graphic display.



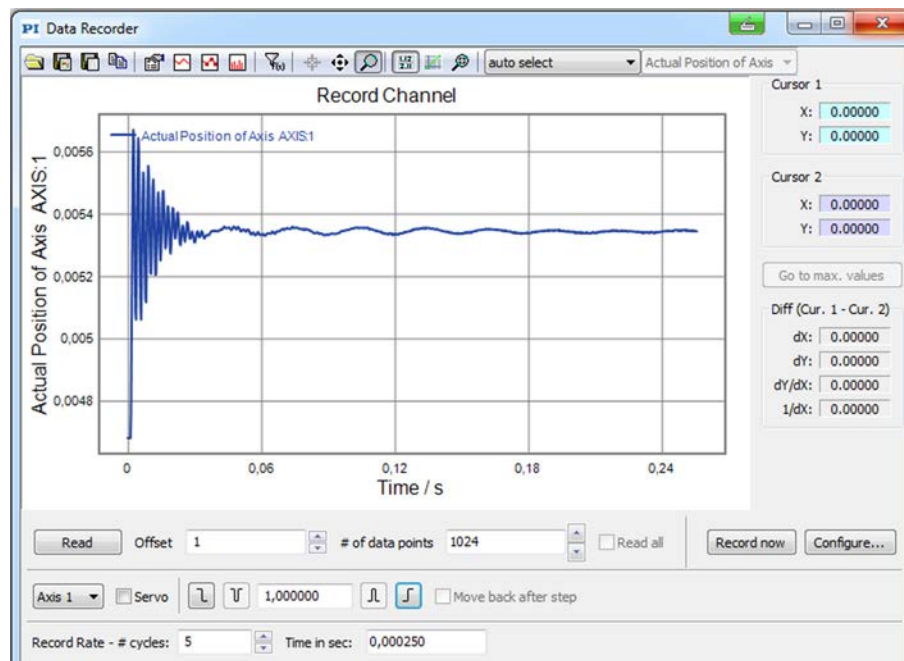
- d) Click the **Configure...** button and make sure that "Actual Position of Axis" is selected in the **Configure Data Recorder** window as the variable to be recorded.





Close the window with **OK**.

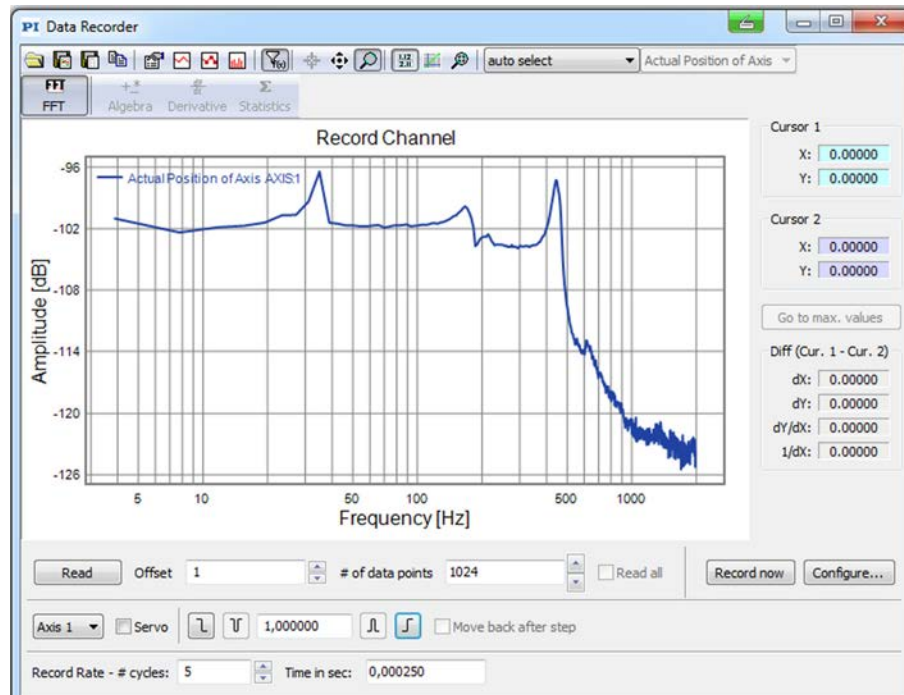
4. In the **Data Recorder** window, start the step in the positive direction as well as the recording by clicking the  button.




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



5. Determine the resonant frequency of the axis from the graphic display of the step response:
- a) Display the Data Toolbar via the  button.

- b) Calculate the FFT (Fast Fourier Transformation) of the step response by clicking the  button. The FFT is displayed graphically.



- c) If necessary, enlarge the view by clicking the  button and, while pressing the left mouse button, dragging the mouse pointer, which has turned into a magnifying glass, over a section of the graphic display (clicking the right mouse button in the graphics field reduces the view back to the original size).
- d) Display the cursors in the graphic display by clicking the  button.
- e) Activate the cursor movement with the mouse by clicking the  button.
- f) Position cursor 1 and 2 over the resonant frequencies by clicking the **Go to max. values** button.
The resonant frequency can be identified by the distinct maximum in the FFT graph. The example shows an initial resonant frequency at 425 Hz and a second at 171 Hz. These maximum values can be read in the respective field **X**: In the **Cursor 1** and **Cursor 2** areas to the right of the graphic display.

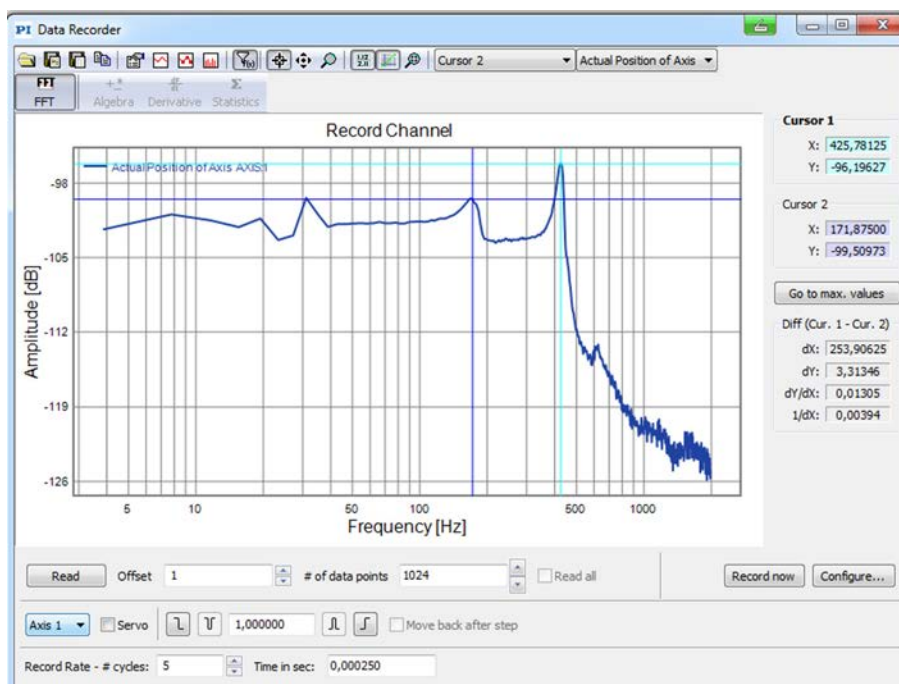
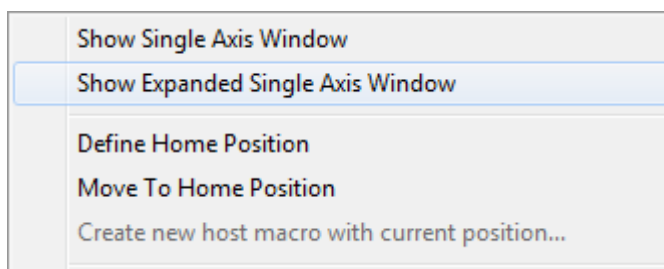


Figure 8: Step response as FFT graph: Resonant frequencies marked by the cursor

6. Go back to the main window of PIMikroMove and open the expanded single axis window for the connected positioner by clicking the right-hand button on the corresponding line of the **Axes** tab and selecting the **Show Expanded Single Axis Window** in the context menu.



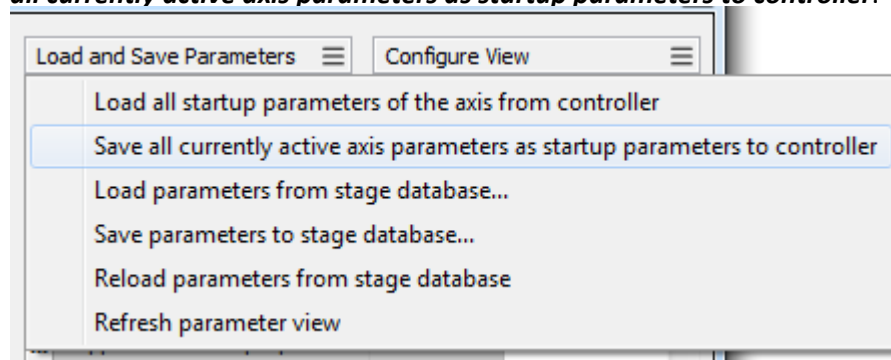
7. Set the notch filter:

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

If, as is the case in the example here, two resonant frequencies are determined (425 Hz and 171 Hz), the **Notch Filter Frequency 1** parameter (ID 0x94) should be set to a value that is roughly in the middle between both resonant frequencies - here, it is 298 Hz. In this case, to increase the bandwidth of the notch filter, the value of the **Notch Filter Edge 1** parameter (ID 0x95) must be reduced so that it is around 0.3 to 0.2.

Proceed as follows to enter the values:

- a) If the **Notch Filter Frequency 1** parameter (ID 0x94) and **Notch Filter Edge 1** parameter (ID 0x95) are not in the list on the right-hand side of the window, click on **Configure View -> Select parameters...** and add them to the list.
 - b) Type the new parameter value into the corresponding input field in the **Active Value** column of the list.
 - c) Press the **Enter** key on the PC keyboard or click outside the input field with the mouse to transfer the parameter value to the volatile memory of the controller.
Note: If a parameter value in the volatile memory (**Active Value** column) is different to the parameter value in the nonvolatile memory (**Startup Value** column), the line in the list is highlighted in color.
8. Save the new settings. You have the following options:
- Save a parameter set in the positioner database on the PC by clicking on **Load and Save Parameters -> Save parameters to stage database...**, see "Creating or Modifying a Positioner Type" (p. 234).
 - Transfer the current values of the listed parameters from the volatile memory to the non-volatile memory of the E-873 by clicking **Load and Save Parameters -> Save all currently active axis parameters as startup parameters to controller**.



6.6 Optimizing the Servo Control Parameters

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

Typically, optimization is determined empirically, i.e., various values are used in closed-loop operation and the behavior of the positioner is monitored. Optimization is done with the following parameters:

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

The D term parameter is set to 0 by default and should not be changed.

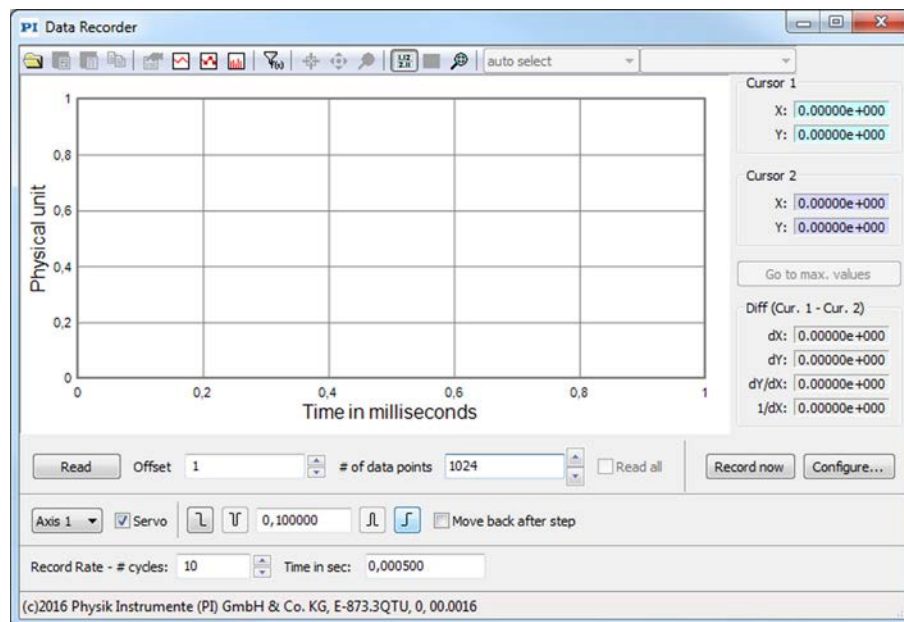
The following describes the procedure for optimizing the servo control parameters in PIMikroMove.

Requirements

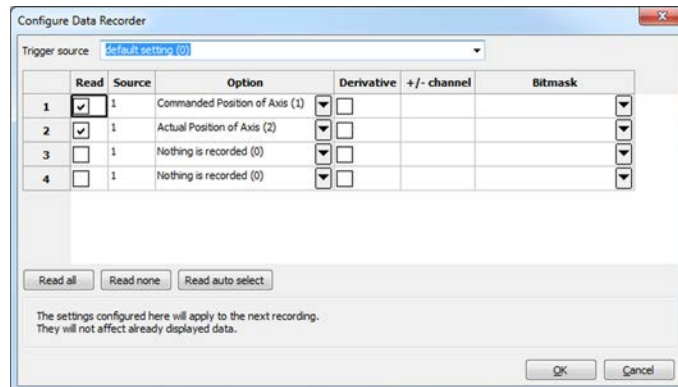
- ✓ You have installed the positioner in the same way as it will be used in your application (corresponding load, orientation, and fixing).
- ✓ You started initial motion (p. 62) with PIMikroMove.
- ✓ If necessary, you have set the notch filter (p. 66).
- ✓ All devices are still ready for operation.

Checking the servo control parameters: Measuring the step response


1. Open the **Data Recorder** window in the main window of PIMikroMove via the **E-873 > Show data recorder** menu item.
2. Select the axis to be checked via the **Axis #** button and switch the servo mode on for the axis by clicking in the **Servo** check box (tick in the checkbox).
3. Configure the data recorder.
 - a) Set the size of the step to be performed to a value that is typical for your application, e.g., 0.100000 (specified in physical units).
 - b) Set the value 10 for the record table rate in the **Record Rate - # cycles** field.
 - c) Set the value 8192 (or less) for the number of data points to be read for the graphic display in the **# of data points** field.



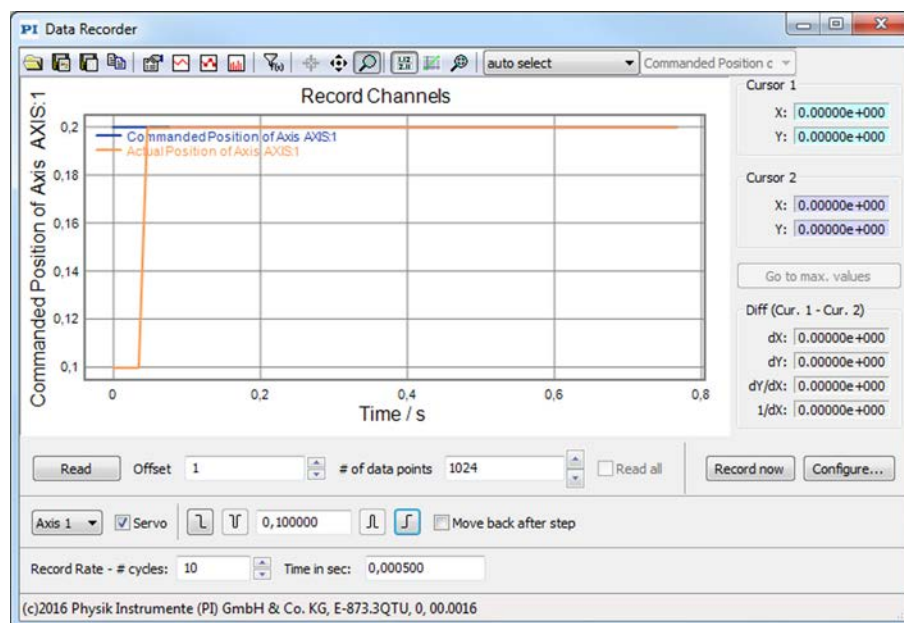
- d) Click the **Configure...** button and make sure that "Commanded Position of Axis" and "Actual Position of Axis" are selected in the **Configure Data Recorder** window as the variables to be recorded.




Close the window with **OK**.

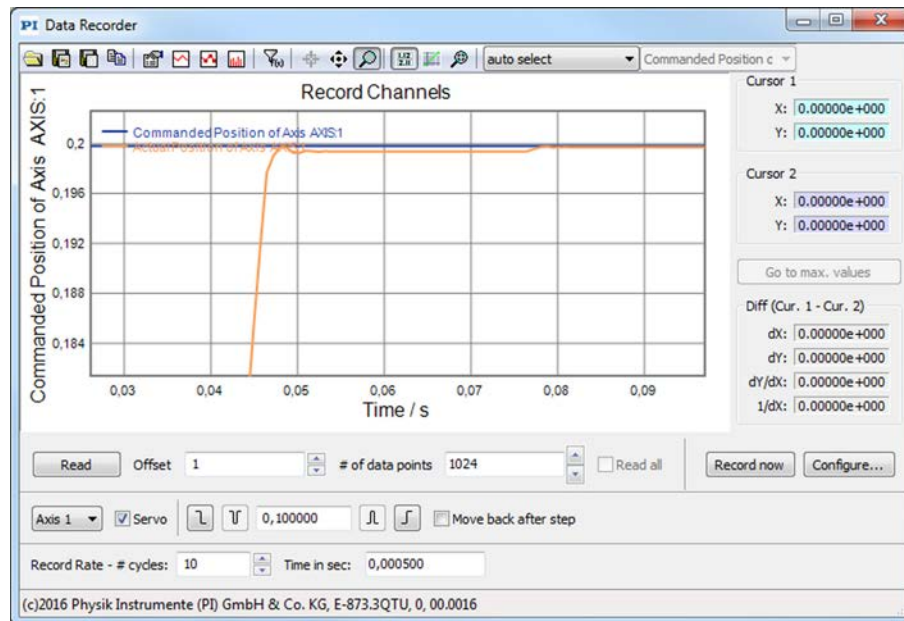
4. In the **Data Recorder** window, start the step in the positive direction as well as the recording by clicking the  button.

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



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

Example for a step response:



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

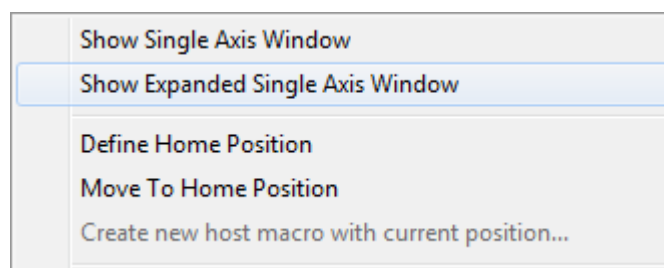
- You already have optimum parameter settings and do not have to do anything further.

If the result is not satisfactory:

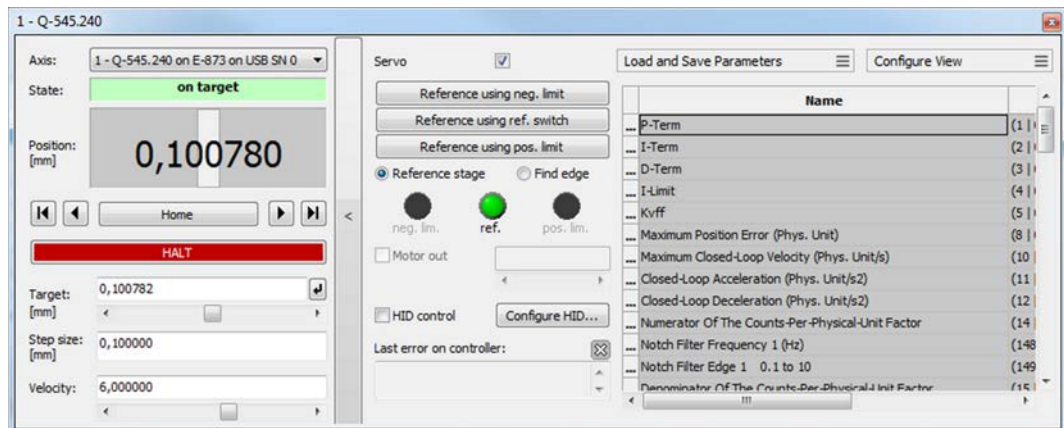
- Optimize the servo control parameters, see below.

Optimizing the servo control parameters

1. Open the expanded single axis window for the connected positioner in the main window of PIMikroMove by clicking the right mouse button on the corresponding line of the **Axes** tab and selecting **Show Expanded Single Axis Window** in the context menu.



2. Enter new values for the parameters to be adapted:



- a) If the parameter to be modified is not included in the list on the right-hand side of the window, click on **Configure view > Select parameters...** and add it to the list. You can also display certain groups of parameters or all axes-related parameters.
 - b) Type the new parameter value into the corresponding input field in the **Active Value** column of the list.
 - c) Press the **Enter** key on the PC keyboard or click outside the input field with the mouse to transfer the parameter value to the volatile memory of the controller. Note: If a parameter value in the volatile memory (**Active Value** column) is different to the parameter value in the nonvolatile memory (**Startup Value** column), the line in the list is highlighted in color.
3. In the **Data Recorder** window, record the step response of the positioner again.

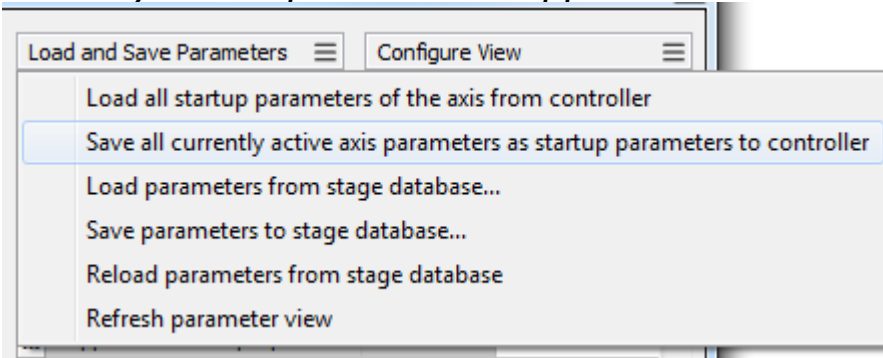
If the result is not satisfactory:

- 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 positioner database on the PC by clicking on **Load and Save Parameters -> Save parameters to stage database...**, see "Creating or Modifying a Positioner Type" (p. 234).
- Transfer the current values of the listed parameters from the volatile memory to the non-volatile memory of the E-873 by clicking **Load and Save Parameters -> Save**

all currently active axis parameters as startup parameters to controller.



7 Operation

In this Chapter

Protective Functions of the E-873	77
Data Recorder	78
Digital Output Signals.....	80
Digital Input Signals.....	87
Controlling with HID.....	91
Controller Macros	99

7.1 Protective Functions of the E-873

7.1.1 Protection Against Overheating

If a certain temperature (70°C) is reached, the E-873 reacts as follows to protect the system against damage:

- The control value is set to zero for the axis concerned.
- The servo mode is switched off for the axis concerned.
- Error code 603 is output.

Then restore the operational readiness (p. 78) for the E-873.

7.1.2 Behavior with Motion Errors

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

Motion error is possible in closed-loop or open-loop operation.

Motion errors can have the following causes, for example:

- Drive malfunction
- Position sensor malfunction
- Positioner malfunction
- Switch off the output for the piezo voltage of the E-873 because of overheating of the integrated driver electronics

If there is a motion error, the E-873 reacts as follows to protect the system against damage:

- Servo mode is switched off for the axis in question.

- Motion is stopped.
- Error code -1024 is output.

Then restore operational readiness (p. 78) for the E-873.

7.1.3 Behavior During System Errors

There is a system error when E-873 is not responsive.

For example, the cause of a system error can be a buffer overflow in the firmware of the E-873.

If a system error occurs the E-873 reacts as follows:

- After a certain delay, the safety function of the **Watchdog Timer** initiates a reboot of the E-873.

7.1.4 Re-establishing Readiness for Operation

1. Send the `ERR?` command to read out the error code.
`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-873 has **not** overheated (internal temperature is maximum 65 °C).
3. If the servo mode was switched off after an error or overheating:
 - Switch on the servo mode for the axis with the `SVO` command.

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

7.2 Data Recorder

7.2.1 Setting up the Data Recorder

The E-873 contains a real-time data recorder. The data recorder can record for example, the current position of the axis.

The recorded data is stored temporarily in 4 data recorder tables with 8192 points each. Each data recorder table contains the data of one data source.

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

INFORMATION

The settings for setting up the data recorder can only be changed in the volatile memory of the E-873. After the E-873 has been switched on or rebooted, factory settings will be active unless a configuration takes place with a startup macro.

Reading general information from the data recorder

- Send the `HDR?` command (p. 150).

The available options are displayed for recording and triggering, as well as the information on additional parameters and commands for data recording.

Configuring data to be recorded

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

- Send the `DRC?` command (p. 138) to read out the current configuration. Data recorder tables with the record option 0 are deactivated, i.e., nothing is recorded. By default, the data recorder tables of the E-873 record the following:
 - Data recorder table 1: Record option 1: Commanded position of the axis
 - Data recorder table 2: Record option 2: Current position of the axis
 - Data recorder table 3: Record option 3: Position error of the axis
 - Data recorder table 4: Record option 73: Control value of the axis
- Configure the data recorder with the `DRC` command (p. 137).

Configuring the recording trigger

You can specify how the recording is to be triggered.

- Get the current trigger option with `DRT?` (p. 142).
- Change the trigger option with the `DRT` command (p. 141). The trigger option applies to all data recorder tables whose record option is not set to 0.

Setting the record table rate

- Send the `RTR?` command (p. 186) to read out the record table rate of the data recorder.

The parameter indicates the number of servo cycles that are required for recording each data point. The default value is 10 servo cycles. The servo cycle time of the E-873 is 100 µs.

- Change the record table rate with the `RTR` command. (p. 186)

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

7.2.2 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. 193).

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.

7.2.3 Reading Recorded Data

INFORMATION

Reading the recorded data can take some time, depending on the number of data points. The data can also be read while data is being recorded.

- Read the last recorded data with the **DRR?** command (p. 139).
The data is output in the GCS array format (see the SM146E user manual on the product CD).
- Get the number of points contained in the last recording with the **DRL?** command (p. 139).

7.3 Digital Output Signals

The digital outputs of the E-873 are available at the **I/O** socket (p. 264).

- Get the number of the output lines available on the E-873 with the **TIO?** command (p. 196).

External devices can be triggered via the digital outputs of the E-873. 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. 99).

7.3.1 Commands for Digital Outputs

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

Command	Syntax	Function
CTO	CTO {<TrigOutID> <CTOPam> <Value>}	Configures the conditions for the trigger output. Couples the trigger output to the axis motion.
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 where the trigger output is activated

Command	Syntax	Function
		by TRO.
TRO	TRO {<TrigOutID> <TrigMode>}	Activates or deactivates the trigger output conditions set with CTO. Default: Trigger output deactivated.

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. 82). 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. 84).
5	Motion Error	The selected digital output line becomes active when a motion error occurs (p. 84). The line stays active until the error code is reset to 0 (by an ERR? query).
6	In Motion	The selected digital output line is active as long as the selected axis is in motion (p. 84).
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. For details, see "Setting Up "Position + Offset" Trigger Mode" (p. 85).
8	Single Position	The selected digital output line is active when the axis position has reached or exceeded a given position (p. 86).

In addition, the polarity (active high / active low) of the signal at the digital output can be set (p. 87).

INFORMATION

The settings for the configuration of the digital output lines can only be modified in the volatile memory of the E-873. After switching on or restarting the E-873 the factory default settings are active, unless a configuration has already been made using the startup macro.

7.3.2 Setting Up "Position Distance" Trigger Mode

The *Position Distance* trigger mode is suitable for 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.

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 activate 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 positioner has covered a distance of 0.1 µm.

➤ 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 activate 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 positioner has covered a distance of 0.1 μm , as long as axis 1 is moving in positive direction of motion within the range of 0.2 μm to 0.55 μm (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 μm and 0.2 μm .

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
```

7.3.3 Setting Up "On Target" Trigger Mode

The on-target state of the axis selected (p. 26) is output at the selected trigger output in *On Target* trigger mode.

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 activate 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
```

7.3.4 Setting Up "Motion Error" Trigger Mode

The *Motion Error* trigger mode lends itself to monitoring motion. 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 an `ERR?` query).

A motion error is present if the current position of the axis does not change when the control value changes. For further information, see "Motion Error".

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`.

7.3.5 Setting Up "In Motion" Trigger Mode

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

The motion state can also be read with the `#5` (p. 123), `#4` (p. 122), and `SRG?` (p. 192) 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 activate the conditions for trigger output, send TRO <TrigOutID> 1.

Example:

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

➤ Send:

```
CTO 1 2 1
CTO 1 3 6
TRO 1 1
```

7.3.6 Setting Up "Position + Offset" Trigger Mode

The *Position+Offset* trigger mode is suitable for scanning applications. The first trigger pulse is output when the axis has reached a specified position (TriggerPosition). The next trigger pulses are output respectively 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 the direction of motion in which trigger pulses are to be output.

The pulse width is one servo cycle.

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 output of the first trigger pulse.
 - Send CTO <TrigOutID> 9 Stop, where Stop indicates the stop value.

2. If you want to activate 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. A pulse should then be output on this line every time axis 1 has covered a distance of 0.1 μ m in the 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. A pulse should then be output on this line every time axis B has covered a distance of 1 μ m in the 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
```

7.3.7 Setting Up "Single Position" Trigger Mode

The selected digital output line is active in *Single Position* trigger mode, 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>) 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 activate 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
```

7.3.8 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
```

7.4 Digital Input Signals

The digital inputs of the E-873 are available on the **I/O** socket (p. 264).

- Get the number of the input lines available on the E-873 with the `TIO?` command (p. 196).
- Get the state of the input lines with the `DIO?` command (p. 136).

Potential applications:

- Use in macros (p. 89). Details and examples of macros can be found in "Controller Macros" (p. 99).
- Use as switch signals (p. 90)

7.4.1 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 the state of a digital input line to a variable when used in conjunction with the DIO? query command. Use in macros to set local variables (p. 116).
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.
FPL	FPL [{<AxisID>}]	Starts a reference move to the positive physical limit of the travel range. Limit switch. A digital input line can be used as the source of the positive limit switch signal.
FRF	FRF [{<AxisID>}]	Starts a reference move to the reference switch. A digital input line can be used as the source of the reference switch signal instead of the reference switch.
JRC	JRC <Jump> <CMD?> <OP> <Value>	Can only be used in macros. Triggers a relative jump of the macro run pointer depending on the state of a digital input line when used in conjunction with the DIO? query command.
MEX	MEX <CMD?> <OP> <Value>	Can only be used in macros. Stops running of the macro depending on the state of a digital input line when used in conjunction 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 conjunction with the DIO? query command.

Parameters

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

Parameters	Description and Possible Values
Source Of Reference Signal 0x5C	Specifies the source of the reference signal for the FRF and FED commands: 0 = reference switch 1 = Digital input 1

Parameters	Description and Possible Values
	2 = Digital input 2 3 = Digital input 3 4 = Digital input 4
Source Of Negative Limit Signal 0x5D	Specifies the source(s) of the negative limit switch signal for the FNL and FED 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)
Source Of Positive Limit Signal 0x5E	Specifies the source(s) of the positive limit switch signal for the FPL and FED 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)
Invert Digital Input Used For Negative Limit 0x5F	Inverts the polarity of the digital inputs, which 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)
Invert Digital Input Used For Positive Limit 0x60	Inverts the polarity of the digital inputs, which 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)

7.4.2 Using Digital Input Signals in Macros

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

- Conditional running 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. 99).

INFORMATION

You can connect the C-170.PB pushbutton box from PI to the **I/O** socket (p. 264) to generate the digital input signals for use in macros. It also displays the state of the digital output lines via LEDs.

7.4.3 Using Digital Input Signals as Switch Signals

The digital inputs on the **I/O** socket can be used as the source of the reference point and limit switch signals (e.g., for reference moves (p. 34)) for an axis.

Using digital input as reference signal

INFORMATION

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

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

INFORMATION

The **Has Reference?** parameter (0x14) has no influence 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 (0x5C).

Detailed information on changing parameters can be found in "Adapting Settings" (p. 229).

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 as limit switch signal may only change once over the entire travel range.

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

INFORMATION

The ***Has No Limit Switches?*** parameter (0x32) determines whether the E-873 evaluates the signals of the internal limit switches of the positioner. This parameter has no influence 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 (0x5D).
- Select the source(s) of the positive limit switch signal for the axis by changing the ***Source Of Positive Limit Signal*** parameter (0x5E).

Detailed information on changing parameters can be found in "Adapting Settings" (p. 229).

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 in the volatile memory of the E-873 only.

- 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.

7.5 Controlling with HID

7.5.1 Functionality of HID Control

Axes of human interface devices can control the following motion parameters of the positioner axes connected to the E-873:

- **Relative target position**

The displacement of the HID axis determines the frequency that moves the positioner axis controlled: The more the HID axis is displaced, the higher the frequency and therefore the velocity, with which the positioner axis is moved.

For further details, see the description of the HIA command (p. 153).

During HID control, the target position of the controlled axis of the E-873 is set to the soft limit that is specified by parameter 0x15 or 0x30 (for details, see "Travel Range and Soft Limits" (p. 29)). When the HID control is deactivated, the target position is set to the current position of the controlled axis.

INFORMATION

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

Programming the HID control

Buttons and LEDs (i.e., output units) of human interface devices can be used for example, in controller macros (p. 99), to program the HID control.

In this manual, you will find an example macro for HID control with saving of positions.

7.5.2 Commands and parameters for HIDs

Commands

The following commands are available for the use of human interface devices:

Com-mand	Syntax	Function
HDT	HDT {<HIDDeviceID> <HIDDeviceAxis> <HIDTableID>}	Assigns a lookup table to the axis of a human interface 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 human interface devices.
HIA	HIA {<AxisID> <MotionParam> <HIDDeviceID> <HIDDeviceAxis>}	Configures the control of axes of the E-873 by axes of human interface 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 human interface devices.
HIE?	HIE? [{<HIDDeviceID> <HIDDeviceAxis>}]	Gets the current displacement of the axes of human interface devices.
HIN	HIN {<AxisID> <HIDControlState>}	Enables or disables the HID control for the axes of the E-873.

Com-mand	Syntax	Function
HIN?	HIN? [{<AxisID>}]	Gets the activation state of the HID control.
HIS?	HIS? [{<HIDDeviceID> <HIDItemID> <HIDPropID>}]	Gets the properties of the operating elements of human interface devices.
HIT	HIT {<HIDTableID> <HIDTableAddr> <HIDTableValue>}	Fills lookup tables with values. The table contents can be saved in the nonvolatile memory with WPA.
HIT?	HIT? [<StartPoint> [<NumberOfPoints> [{<HIDTableID>}]]]	Gets the values of the points in the lookup tables.

Parameters

The following parameter is available for the HID control:

Parameters	Description and Possible Values
<i>Invert Direction Of Motion For Joystick-Controlled Axis?</i> 0x61	Specifies the direction of motion for the axes of the E-873 during HID control. 0 = direction of motion not inverted (default setting) 1 = direction of motion inverted

7.5.3 Testing the HID

We recommend testing the HID operating elements in PIMikroMove after connecting it to the E-873 .

INFORMATION

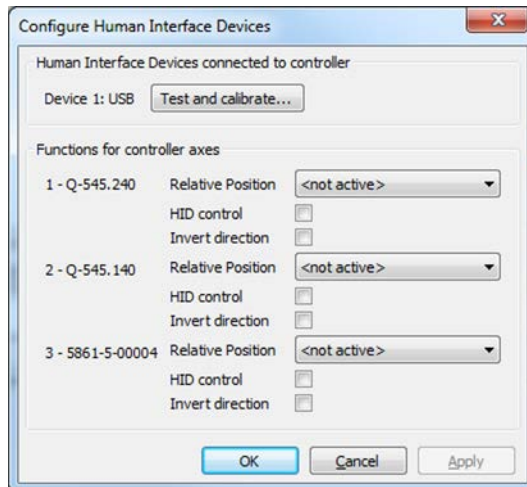
A positioner does not have to be connected to the E-873 to test HID operating elements in PIMikroMove.

Requirements

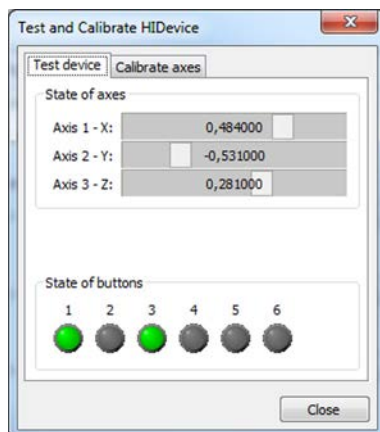
- ✓ You have read and understood the general notes on startup (p. 53).
- ✓ PIMikroMove is installed on the PC (p. 47).
- ✓ PIMikroMove has established communication between the E-873 and the PC (p. 54).
- ✓ You have connected (p. 45) the HID to the E-873.

Testing the a human interface device in PIMikroMove

1. In the main window of PIMikroMove, open the **Configure Human Interface Devices** window via the **E-873 > 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 human interface device:
 - Move the axes of the human interface device and observe the status displays in the **State of axes** area.
 - Press the buttons of the human interface device and observe the status displays in the **State of buttons** area.



In this example, a joystick with 3 axes and 6 buttons is connected to a E-873. The joystick axes are available via the identifiers 1 to 3 and the buttons via the identifiers 1 to 6. Current status in the figure: The X axis of the joystick is displaced in a positive direction, the Y axis is displaced in a negative direction, the Z axis is displaced in a positive direction, and buttons 1 and 3 are pressed.

7.5.4 Setting Up and Enabling HID Control

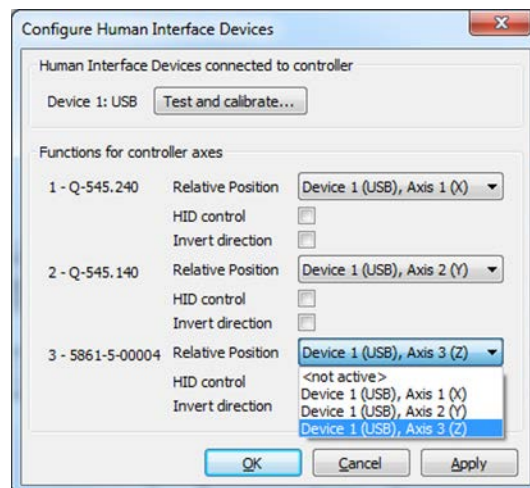
It is recommended to use PIMikroMove for setting up and enabling the HID control. Before the HID control is enabled, it is recommended to test the connected human interface device (p. 93).

Requirements

- ✓ You have carried out a successful reference move for each axis of the E-873 with PIMikroMove; see "Starting Motions" (p. 62).
- ✓ You have connected the human interface device to the E-873 .
- ✓ All devices are still ready for operation.

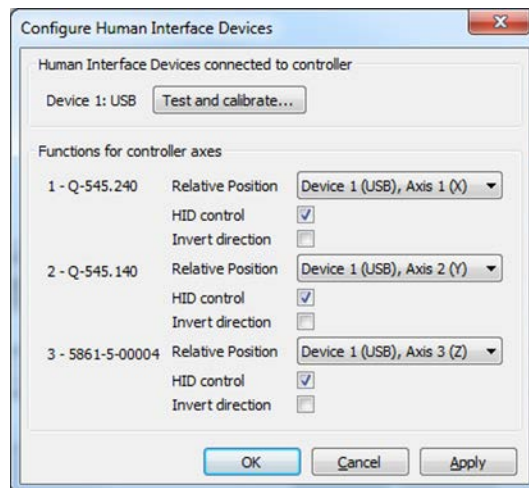
Setting Up and Enabling HID Control in PIMikroMove

1. In the main window of PIMikroMove, open the **Configure Human Interface Devices** window via the **E-873 > Configure controller HIDevice(s)...** menu item.
2. Set up the HID control for the axes of the E-873 in the **Functions for controller axes** area of the **Configure Human Interface Devices** window:
 - a) In the corresponding field, select the axis of the human interface device to be used for the **Relative Position** motion parameter to be controlled.



- b) Enable the HID control for the axes by clicking in the respective **HID Control** checkbox.
- c) If the direction of motion of an axis is to be inverted during HID control, click in the **Invert direction** check box.

In the example shown, the three axes of the controller are controlled via the X, Y, and Z axis of the digital joystick (Device 1: USB).



3. Send the settings for setting up the HID control to the E-873 by clicking the **OK** button. The **Configure Human Interface Devices** window closes.
4. Make sure that the servo mode for the HID-controlled axes of the E-873 is switched off (e.g., by clicking in the **Servo** check box on the **Axes** tab in the main window of the program).

The axes of the E-873 can now be controlled by the axes of the human interface device according to the settings made in step 2.

If HID control does not work satisfactorily:

- Follow the instructions in "Calibrating the Axes of Human Interface Devices" (p. 96).

If you want to save the assignment of the axis of the human interface device to a motion parameter in the nonvolatile memory of the E-873:

- Follow the instructions in "Saving the Configuration of the HID Control Permanently" (p. 98).

7.5.5 Calibrating HID Axes

Calibrating the axis of a human interface device in PIMikroMove

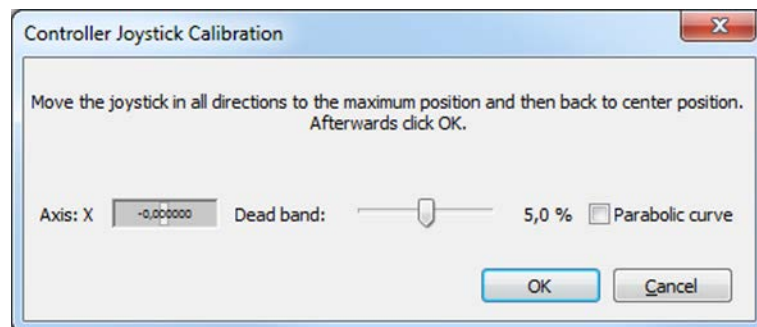
1. In the main window of PIMikroMove, open the **Configure Human Interface Devices** window via the **E-873 > Configure controller HIDevice(s)...** menu item.
2. Open the **Test and Calibrate HIDevice** window by clicking the **Test and calibrate...** button.
3. In the **Test and Calibrate HIDevice** window, select the **Calibrate axes** tab.

4. Assign a lookup table with the **User Table** designation to the axis of the human interface device to be calibrated in the corresponding selection field.



In the example shown, the X axis of the joystick connected is assigned to the user-defined "User Table 101" lookup table. The predefined parabolic lookup table is assigned to axes 2 and 3 respectively.

5. Calibrate the axis of the human interface 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 human interface device one after the other to all extreme positions. The custom lookup table values are determined in this way.
 - c) If you want to change the neutral area of the axis (i.e., the area around the middle position of the axis where no change in the controlled motion parameter is triggered), set the **Dead band** slider in the **Controller Joystick Calibration** window correspondingly.
 - d) If the values in the user-defined lookup table are to describe a parabolic curve shape, click the **Parabolic curve** check box in the **Controller Joystick Calibration** window.
 - e) Click **OK** in the **Controller Joystick Calibration** window to write the lookup table values to the volatile memory of the E-873. 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 human interface device and the contents of user-defined lookup tables in the nonvolatile memory of the E-873:
 - 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. 95).
 - 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 "Saving the Configuration of the HID Control Permanently" (p. 98).

7.5.6 Saving the Configuration of HID Control Permanently

The following settings for the configuration of HID control can be saved in the nonvolatile memory of the E-873:

- Assignment of lookup tables to the axes of the human interface device; see "Calibrating HID Axes" (p. 96)
- For contents of user-defined lookup tables, see "Calibrating HID Axes" (p. 96)
- To assign the HID axes to the motion variables to be controlled for the E-873's axis, see "Setting up and Activating HID Control" (p. 95)

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 when switching on or rebooting the E-873 and take effect immediately.

Requirements

- ✓ You have read and understood the general notes on startup (p. 53).
- ✓ PIMikroMove is installed on the PC (p. 47).
- ✓ You have read and understood the PIMikroMove manual. The manual is on the data storage device for the product.
- ✓ PIMikroMove has established communication between the E-873 and the PC (p. 54).

Saving the configuration of the HID control permanently in PIMikroMove

If you want to write the current settings for the configuration of HID control to the nonvolatile memory of the E-873:

1. Select the **E-873 > Save parameters to nonvolatile memory** menu item in the main window of PIMikroMove. The **Save Parameters to Non-Volatile Memory** dialog opens.
2. Enter either *HID* into the selection field of the **Save Parameters to Non-Volatile Memory** or select *Settings of HDT, HIA, HIT (HID)*.
3. Click **OK** to save and to close the dialog.

INFORMATION

The settings for the configuration of the HID control are also written to the nonvolatile memory of the E-873 if you select the *All Parameters, Settings of HDT, HIA, HIT (100)* or enter the password *100*. However, the entry or the password *100* also saves the current values of all parameters of the E-873, see the description for the WPA (p. 203) command and "Adapting Settings" (p. 229).

7.5.7 Available HIDs

PI offers the HIDs described in the following as optional accessories (p. 10).

C-819.JD: Digital joystick for 2 axes

Operating elements:

- Lever: Control of the X and Y axes
- Buttons (from left to right):
 - 1: X axis lock
 - 2: Y axis lock
 - 3: Free
- LEDs of buttons 1 to 3: Button status indicator (active/inactive)

7.6 Controller Macros

7.6.1 Overview: Macro Functionality and Example Macros

The E-873 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 startup macro. The startup macro runs each time the E-873 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.
- Variables (p. 116) 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.

You will find example macros in this manual for the following tasks:

- Moving an axis back and forth (p. 104)
- Moving an axis with a variable travel range back and forth (p. 105)
- Implementing multiple calls of a macro via a loop (p. 106)
- Preparing an axis via startup macro for closed-loop operation (p. 108)
- Stopping motion by pushbutton (p. 110)

7.6.2 Commands and Parameters for Macros

Commands

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

Command	Syntax	Function
ADD (p. 125)	ADD <Variable> <FLOAT1> <FLOAT2>	Adds two values and saves the result to a variable (p. 116). Can only be used for local variables in macros.
CPY (p. 128)	CPY <Variable> <CMD?>	Copies a command response to a variable (p. 116). Can only be used for local variables in macros.
DEL (p. 133)	DEL <uint>	Can only be used in macros. Delays <uint> milliseconds.
JRC (p. 168)	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. 169)	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 startup macro.
	MAC DEF?	Gets the startup macro.
	MAC DEL <macroname>	Deletes the given macro.

Command	Syntax	Function
	MAC END	Ends the macro recording.
	MAC ERR?	Reports the last error that occurred while the macro was running.
	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>]]	Runs the specified macro. The values of local variables can be set for the macro with <String1> and <String2>.
MAC? (p. 172)	MAC? [<macroname>]	Lists all macros or the content of a given macro.
MEX (p. 174)	MEX <CMD?> <OP> <Value>	Can only be used in macros. Stops the macro execution depending on a condition.
RMC? (p. 183)	RMC?	Lists macros which are currently running.
VAR (p. 200)	VAR <Variable> <String>	Sets a variable (p. 116) to a certain value or deletes it. Can only be used for local variables in macros.
VAR? (p. 201)	VAR? [{<Variable>}]	Gets variable values.
WAC (p. 202)	WAC <CMD?> <OP> <Value>	Can only be used in macros. Waits until a condition is met.
#8 (p. 123)	-	Tests if a macro is running on the controller.

Parameters

The following parameter is available for working with macros:

Parameters	Description and Possible Values
Ignore Macro Error? 0x72	Determines whether the controller macro is stopped if an error occurs when it is running. <ul style="list-style-type: none"> 0 = Stop macro when error occurs (default) 1 = Ignore error

7.6.3 Working with Macros

Work with macros comprises the following:

- Recording of macros (p. 103)
- Starting macro execution (p. 104)
- Stopping macro execution (p. 107)
- Setting up a startup macro (p. 107)
- Deleting of macros (p. 108)

INFORMATION

The E-873 can save up to 32 macros simultaneously. A maximum of 5 nesting levels are possible in macros.

INFORMATION

Basically all GCS commands (p. 113) can be included in a macro. Exceptions:

- `RBT` for rebooting the E-873
- `MAC BEG` and `MAC END` for macro recording
- `MAC DEL` for deleting a macro

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

INFORMATION

To make the use of macros more flexible, you can use local and global variables in macros. For further information, see "Variables" (p. 116).

INFORMATION

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

- Only record macros if it is necessary.
- Use variables (p. 116) in macros to make macros more flexible, and give the corresponding variable values when starting macro execution.
- Contact our customer service department (p. 257) if the E-873 shows unexpected behavior.

INFORMATION

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

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

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-873.

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-873 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-hand 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 motion in a positive direction and waits until the axis has reached the target position. Macro 2 does 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
```

Starting macro execution**INFORMATION**

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.

INFORMATION

It is not possible to run several macros simultaneously. Only one macro can be run 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 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. 229).

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 by 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 therefore 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 the E-873 is switched on or rebooted, since they are only written to the volatile memory of the E-873.

- 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*).

- 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.

- 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 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 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 macro execution, send the `MAC ERR?` command. The response shows the last error that occurred.

Setting up a startup macro

Any macro can be defined as the startup macro. The startup macro is executed each time the E-873 is switched on or rebooted.

INFORMATION

Deleting a macro does not delete its selection as a startup 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 in the PIMikroMove manual.

- Define a macro as the startup macro with the `MAC DEF macroname` command, whereby *macroname* indicates the name of the macro.

- If you want to cancel the selection of the startup macro and do not want to define another macro as the startup macro, only send `MAC DEF`.
- Get the name of the currently defined startup macro by sending the `MAC DEF?` command.

Example: Preparing an axis via a startup 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 startup macro, axis 1 is ready for closed-loop operation immediately after switch-on.

- Send:


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

INFORMATION

When using this macro, the parameter settings of the E-873 should be adapted in the nonvolatile memory to the connected positioner. Alternatively, the parameter settings can also be configured in the volatile memory via the startup macro. For further information, see "Adapting Settings" (p. 229).

Deleting a macro

INFORMATION

A macro cannot be deleted while it is running.

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 in the PIMikroMove manual.

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

7.6.4 Making Backups and Loading Controller Macros

For example, making backups of controller macros on the PC can be useful before updating the firmware (p. 247).


INFORMATION

The use of the **Controller macros** tab in PIMikroMove is recommended for backing up and loading controller macros. A detailed description of the tab can be found in the PIMikroMove manual.

Backing up controller macros onto the PC with PIMikroMove


1. Select the **Controller macros** tab in the PIMikroMove main window.
2. Select the macros in the **Macros on controller** list that you want to back up to the PC:
 - Click the desired entry in the list to select an macro.
 - To select several macros, hold down the Shift button and click the desired entries in the list.
 - To deselect, click an empty area in the list.

By selecting one or more macros, the  (Save selected macros to PC) button becomes active.

3. Save the selected macros on the PC:
 - a) Click the  button to open a directory selection window.
 - b) Select the directory on the PC where you want to save the macros.
 - c) Click **Save**.

The macros are saved as text files (<macro name>.txt) to the directory selected on the PC.

Loading controller macros from the PC to the E-873 with PIMikroMove

1. Select the **Controller macros** tab in the PIMikroMove main window.
2. Load macros from the PC to the E-873:
 - a) Click the  button to open a file selection window.

- b) Select the text files (<macro name>.txt) in the file selection window whose contents you want to load as a macro from the PC to the E-873.
- c) Click **Open**.

For each selected text file (<macro name>.txt), the content is loaded as a macro <macro name> into the E-873.

7.6.5 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 the state of the digital output lines via LEDs.

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 I/O socket to a suitable signal source.	- Pin assignment see "I/O" (p. 264).	The digital input signal can be used for a conditional jump of the macro execution pointer, for example.
Record the HALT macro on the controller.	<code>MAC BEG halt</code> <code>MVR 1 5</code> <code>JRC 2 DIO? 1 = 1</code> <code>JRC -1 ONT? 1 = 0</code> <code>HLT 1</code> <code>MAC END</code>	The macro has the following tasks: <ul style="list-style-type: none"> Start relative motion of axis 1 Set a 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. Set a 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.
Start the HALT macro on the controller.	<code>MAC START halt</code>	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.

Action	Command	Result
If error code 10 interferes: Record alternative HALTVAR macro which uses a variable. See "Variables" (p. 116) details.	<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, axis 1 is not stopped by pushbutton via the HLT command; instead the result of the POS? 1 query is copied to the TARGET variable. Then this variable is used as the target position for the MOV command. As a result, the axis stays right where it was. To clean up, TARGET is defined as empty with the VAR 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 GCS Commands

In this Chapter

Notation	113
GCS Syntax for Syntax Version 2.0	113
Target and Sender Address	116
Variables	116
Command Overview	118
Command Descriptions for GCS 2.0	122
Error Codes	205

8.1 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 element identifier or a command-specific parameter
[...]	Square brackets indicate an optional entry
{...}	Braces indicate a repetition of entries, i.e., that it is possible to access more than one element (e.g., several axes) in one command line.
	LineFeed (ASCII char #10), is the default termination character (character at the end of a command line)
	Space (ASCII char #32), indicates a space character
"..."	Quotation marks indicate that the characters enclosed are returned or to be entered.

8.2 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. However, the response to a single-character command is followed by a termination character.

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 Elements (p. 16) for the identifiers supported by the E-873.

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 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>SPLF}
```

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

 for the last line!

The arguments are listed in the response 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>SPLF
```

```
<Arg1>="<Val1>SPLF
```

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

Example 5:

Send: TSP?{"SP"2{"SP"1LF

Receive: 2=-1158.4405SPLF

1=+0000.0000LF

INFORMATION

With the E-873 only a single element per command line can be addressed (e. g. axis 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.

8.3 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.

Because the PC may only send command lines to the controller, the sender address (0) can be omitted. However, both the target and the sender addresses are part of the controller reply. Multiline responses include the target and sender address only in the first line.

Because the target address can be omitted when the target controller has address 1, the target address (1) can be omitted. 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: Query the device identification string of the E-873.3QTU with address 1

Send: `*IDN?`

The controller replies:

```
(c)2019 Physik Instrumente(PI) Karlsruhe, E-873.3QTU, 0, 1.2.0.0
```

Send: `1 *IDN?`

The same controller replies:

```
0 1 (c)2019 Physik Instrumente(PI) Karlsruhe, E-873.3QTU, 0, 1.2.0.0
```

8.4 Variables

For more flexible programming, the E-873 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 may not contain special characters (especially not "\$").
- 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 may 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 braces.

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

Note that if the braces 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. 169) 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. 125), `CPY` (p. 128) or `VAR` (p. 200), 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?`.

8.5 Command Overview

Com-mand	Arguments	Description
#4		Request Status Register (p. 122)
#5		Request Motion Status (p. 123)
#7		Request Controller Ready Status (p. 123)
#8		Query If Macro Is Running (p. 123)
#24		Stop All Axes (p. 124)
*IDN?		Get Device Identification (p. 124)
ADD	<Variable> <FLOAT1> <FLOAT2>	Add and Save To Variable (p. 125)
CCL	<Level> [<PSWD>]	Set Command Level (p. 127)
CCL?		Get Command Level (p. 128)
CPY	<Variable> <CMD?>	Copy Into Variable (p. 128)
CST?	[[<AxisID>]]	Get Assignment Of Stages To Axes (p. 129)
CSV?		Get Current Syntax Version (p. 130)
CTO	{<TrigOutID> <CTOPam> <Value>}	Set Configuration Of Trigger Output (p. 130)
CTO?	[[<TrigOutID> <CTOPam>]]	Get Configuration Of Trigger Output (p. 133)
DEL	<uint>	Delay The Command Interpreter (p. 133)
DFH	[[<AxisID>]]	Define Home Position (p. 134)
DFH?	[[<AxisID>]]	Get Home Position Definition (p. 135)
DIO	{<DIOID> <OutputOn>}	Set Digital Output Lines (p. 136)
DIO?	[[<DIOID>]]	Get Digital Input Lines (p. 136)
DRC	{<RecTableID> <Source> <RecOption>}	Set Data Recorder Configuration (p. 137)
DRC?	[[<RecTableID>]]	Get Data Recorder Configuration (p. 138)
DRL?	[[<RecTableID>]]	Get Number Of Recorded Points (p. 139)
DRR?	[<StartPoint> <NumberOfPoints> [[<RecTableID>]]]	Get Recorded Data Values (p. 139)
DRT	{<RecTableID> <TriggerSource> <Value>}	Set Data Recorder Trigger Source (p. 141)
DRT?	[[<RecTableID>]]	Get Data Recorder Trigger Source (p. 142)
ERR?		Get Error Number (p. 142)
FED	{<AxisID> <EdgeID> <Param>}	Find Edge (p. 143)

Com-mand	Arguments	Description
FNL	FNL [{<AxisID>}]	Fast Reference Move To Negative Limit (p. 145)
FPL	FPL [{<AxisID>}]	Fast Reference Move To Positive Limit (p. 146)
FRF	[{<AxisID>}]	Fast Reference Move To Reference Switch (p. 147)
FRF?	[{<AxisID>}]	Get Referencing Result (p. 148)
GOH	[{<AxisID>}]	Go To Home Position (p. 149)
HAR?	[{<AxisID>}]	Indicate Hard Stops (p. 149)
HDR?		Get All Data Recorder Options (p. 150)
HDT	{<HIDeviceID> <HIDeviceAxis> <HIDTableID>}	Set HID Default Lookup Table (p. 151)
HDT?	[{<HIDeviceID> <HIDeviceAxis>}]	Get HID Default Lookup Table (p. 152)
HIA	{<AxisID> <MotionParam> <HIDeviceID> <HIDeviceAxis>}	Configure Control Done By HID Axis (p. 153)
HIA?	[{<AxisID> <MotionParam>}]	Get Configuration Of Control Done By HID Axis (p. 154)
HIB?	[{<HIDeviceID> <HIDeviceButton>}]	Get State Of HID Button (p. 154)
HIE?	[{<HIDeviceID> <HIDeviceAxis>}]	Get Deflection Of HID Axis (p. 155)
HIN	{<AxisID> <HIDControlState>}	Set Activation State For HID Control (p. 156)
HIN?	[{<AxisID>}]	Get Activation State Of HID Control (p. 157)
HIS?	[{<HIDeviceID> <HIDItemID> <HIDPropID>}]	Get Configuration Of HI Device (p. 157)
HIT	{<HIDTableID> <HIDTableAddr> <HIDTableValue>}	Fill HID Lookup Table (p. 160)
HIT?	[<StartPoint> [<NumberOfPoints> [{<HIDTableID>}]]]	Get HID Lookup Table Values (p. 161)
HLP?		Get List of Available Commands (p. 163)
HLT	[{<AxisID>}]	Halt Motion Smoothly (p. 164)
HPA?		Get List Of Available Parameters (p. 164)
IFS	<Pswd> {<InterfacePam> <PamValue>}	Set Interface Parameters As Default Values (p. 166)
IFS?	[{<InterfacePam>}]	Get Interface Parameters As Default Values (p. 167)
JRC	<Jump> <CMD?> <OP> <Value>	Jump Relatively Depending On Condition (p. 168)
LIM?	[{<AxisID>}]	Indicate Limit Switches (p. 169)

Com-mand	Arguments	Description
MAC	<keyword> {<parameter>} BEG <macro> DEF <macro> DEF? DEL <macro> END ERR? NSTART <macro> <uint> [<String1> [<String2>]] START <macro> [<String1> [<String2>]]	Call Macro Function (p. 169)
MAC?	[<macroname>]	List Macros (p. 172)
MAN?	<CMD>	Get Help String For Command (p. 173)
MEX	<CMD?> <OP> <Value>	Stop Macro Execution Due To Condition (p. 174)
MOV	{<AxisID> <Position>}	Set Target Position (p. 175)
MOV?	[{<AxisID>}]	Get Target Position (p. 176)
MVR	{<AxisID> <Distance>}	Set Target Relative To Current Position (p. 177)
OMA	{<AxisID> <Position>}	Set Open-Loop Target Position (p. 178)
OMA?	[{<AxisID>}]	Get Open-Loop Target Position (p. 179)
OMR	{<AxisID> <Distance>}	Set Open-Loop Target Relative To Current Position (p. 179)
ONT?	[{<AxisID>}]	Get On-Target State (p. 180)
OSM	{<AxisID> <Value>}	Open-Loop Step Moving (p. 181)
OSN?	[{<AxisID>}]	Read Left Steps (p. 181)
POS	{<AxisID> <Position>}	Set Real Position (p. 182)
POS?	[{<AxisID>}]	Get Real Position (p. 182)
RBT		Reboot System (p. 183)
RMC?		List Running Macros (p. 183)
RON	{<AxisID> <ReferenceOn>}	Set Reference Mode (p. 184)
RON?	[{<AxisID>}]	Get Reference Mode (p. 184)
RPA	[{<ItemID> <PamID>}]	Reset Volatile Memory Parameters (p. 185)
RTR	<RecordTableRate>	Set Record Table Rate (p. 186)
RTR?		Get Record Table Rate (p. 186)
SAI	{<AxisID> <NewIdentifier>}	Set Current Axis Identifiers (p. 187)
SAI?	[ALL]	Get List Of Current Axis Identifiers (p. 187)

Com-mand	Arguments	Description
SEP	<Pswd> {<ItemID> <PamID> <PamValue>}	Set Nonvolatile Memory Parameters (p. 188)
SEP?	[[<ItemID> <PamID>]]	Get Nonvolatile Memory Parameters (p. 189)
SPA	{<ItemID> <PamID> <PamValue>}	Set Volatile Memory Parameters (p. 190)
SPA?	[[<ItemID> <PamID>]]	Get Volatile Memory Parameters (p. 191)
SRG?	{<AxisID> <RegisterID>}	Query Status Register Value (p. 192)
STE	<AxisID> <Amplitude>	Start Step And Response Measurement (p. 193)
STP		Stop All Axes (p. 194)
SVO	{<AxisID> <ServoState>}	Set Servo Mode (p. 195)
SVO?	[[<AxisID>]]	Get Servo Mode (p. 196)
TIO?		Tell Number Of Digital I/O Lines (p. 196)
TMN?	[[<AxisID>]]	Get Minimum Commandable Position (p. 197)
TMX?	[[<AxisID>]]	Get Maximum Commandable Position (p. 197)
TNR?		Get Number Of Record Tables (p. 198)
TRO	{<TrigOutID> <TrigMode>}	Set Trigger Output State (p. 198)
TRO?	[[<TrigOutID>]]	Get Trigger Output State (p. 198)
TRS?	[[<AxisID>]]	Indicate Reference Switch (p. 199)
TVI?		Tell Valid Character Set For Axis Identifiers (p. 200)
VAR	<Variable> <String>	Set Variable Value (p. 200)
VAR?	[[<Variable>]]	Get Variable Value (p. 201)
VER?		Get Versions Of Firmware And Drivers (p. 202)
WAC	<CMD?> <OP> <Value>	Wait For Condition (p. 202)
WPA	<Pswd> [[<ItemID> <PamID>]]	Save Parameters To Non-Volatile Memory (p. 203)

8.6 Command Descriptions for GCS 2.0

#4 (Request Status Register)

Description: Requests system status information.

Format: #4

Arguments: None

Response: The response is bit-encoded. See below for the individual codes.

Notes: This command is identical in function to SRG? (p. 192), 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-873, the response is the sum of the following codes, in hexadecimal format:

Bit	15	14	13	12	11	10	9	8
Description	On-target state	Is referencing	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 switch	Negative limit switch

Example:

Send: #4

Receive: 0x9005

Note: The response is 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 positioner is on the positive side of the reference 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 status of the axes.
Format:	#5
Arguments:	None
Response:	The response <uint> is bit-encoded and returned as the hexadecimal sum of the following codes: 1=First axis in motion 2=Second axis in motion 4=Third axis in motion ...
Examples:	0 indicates motion of all axes complete 3 indicates that the first and the second axis are in motion

#7 (Request Controller Ready Status)

Description:	Asks controller for ready status (tests if controller is ready to run a new command). Note: Use #5 (p. 123) instead of #7 to verify if motion has ended.
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., doing 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 further details, see the notes below.

Sets error code to 10.

This command is identical in function to STP (p. 194), 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. 175), MVR (p. 177), GOH (p. 149), STE (p. 193), OSM (p. 181), OMA (p. 178), OMR (p. 179)), commands for referencing (FNL (p. 145), FPL (p. 146), FRF (p. 147)) and macros (MAC (p. 169)). Also stops macro execution.

After the axis has been stopped, its target position is set to its current position.

HLT (p. 164) in contrast to #24 stops motion with given deceleration with regard to system inertia.

*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: With E-873, *IDN? responds something like:

```
(c)2019 Physik Instrumente (PI) GmbH &
Co. KG, E-873, 113059603, 1.005
```

ADD (Add and Save to Variable)

Description: Adds two values and saves the result to a variable (p. 116).

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 where the result is to be copied is specified 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", do 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 therefore determines the availability of commands and 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 apply:

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 password required is "advanced".

Level > 1 is provided for PI service personnel only. Users cannot change to a level > 1. Contact the customer service department if you have problems with parameters for command level 2 or higher (p. 257).

Response: None

Troubleshooting: Invalid password

Notes: With the E-873, the command levels only determine the write permission for the parameters. The availability of the commands of the E-873 is independent of the active command level.

HPA? (p. 164) lists the parameters including the

information on which command level allows write access to them. For further information on using parameters, see "Adapting Settings" (p. 229).

After controller switch-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:	<p><Level> should be 0 or 1.</p> <p><Level> = 0 is the default setting, write access is given for level 0 parameters, read access is given for all parameters</p> <p><Level> = 1 allows write access for level 1 parameters (parameters from level 0 are included).</p>

CPY (Copy Into Variable)

Description:	<p>Copies a command response to a variable (p. 116).</p> <p>The variable is present in volatile memory (RAM) only.</p>
Format:	CPY <Variable> <CMD?>
Arguments:	<p><Variable> is the name of the variable to which the command response is to be copied.</p> <p><CMD?> 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: 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.

Write the "connect" macro:

```
MAC BEG connect
CPY 1 DIO? 0
DIO 0 $1
MAC START CONNECT
MAC END
```

Example 2: It is possible to copy the value of one variable (e.g., SOURCE) to another variable (e.g., TARGET):

```
CPY TARGET VAR? SOURCE
```

CST? (Get Assignment Of Stages To Axes)

Description: Returns the name of the connected positioner type for the queried axis.

Format: CST? [{<AxisID>}]

Arguments: <AxisID> is one axis of the controller

Response: {<AxisID>=" "<string> LF}

where

<string> is the name of the positioner type assigned to the axis.

Notes: The positioner name is read from the **Stage Name** parameter (ID 0x3C). If 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 positioner with SPA (p. 190) or SEP (p. 188). Because the PC software from PI uses the parameter value to configure the E-873 for the connected positioner (p. 62), it is not recommended to change manually with

SPA or SEP.

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. 198). Do not use DIO (p. 136) on digital output lines where the trigger output is activated by TRO.

The CTO settings are lost when you power down or reboot the E-873. They can be easily maintained by saving them in a macro.

Output lines and trigger conditions available: <TrigOutID> corresponds to digital output lines 1 to 4, IDs = 1 to 4; see "I/O".

<CTOPam> parameter IDs available for E-873:

- 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 activate the trigger output for a limited position range and a certain direction of motion only (negative or positive; Note: If 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 motion state can also be read with commands, e.g. SRG? or #5).
- 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 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 the direction of motion, for which trigger pulses are to be output. Trigger processing is done by the DSP of the E-873.

- 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 is active when this position is reached or exceeded

For application examples and further details see "Digital Output Signals" (p. 80) 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 μm . 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, digital output line 1 is to 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).

Further information can be found in the description of the MAC command (p. 169) and in the "Controller Macros" (p. 99) section.

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. 182) (Get the current position)
- TMN? (p. 197) (Get the minimum commandable position)
- TMX? (p. 197) (Get the maximum commandable position)

DFH does **not** change the values of the parameters for the definition of travel range and soft limits (p. 29).

The offset is reset to zero in the following cases:

- When switching on and rebooting the E-873: For all axes
- During referencing: For the affected axis

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. Finally, 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 new zero position using DFH. Therefore, the offset value of axis 1 is 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:	<p>Queries the position value that is currently used as the offset for the specified axis to move the zero position.</p> <p>If all arguments are omitted, queries the position value of all axes.</p>
Format:	DFH? [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller
Response:	{<AxisID>="<PositionOffset> LF}
	<p>where</p> <p><PositionOffset> is the axis position that was valid at the time the last DFH command was processed. This position value is used internally as offset for the calculation of the current axis position.</p>
Troubleshooting:	Illegal axis identifier
Notes:	<p>The axis position that 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:</p> <ul style="list-style-type: none"> ▪ When switching on and rebooting the E-873: For all axes ▪ During referencing: For the affected axis

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. 196) to get the number of installed digital I/O lines.
Format:	DIO {<DIOID> <OutputOn>}
Arguments:	<p><DIOID> is one digital output line of the controller, see below for details.</p> <p><OutputOn> is the state of the digital output line, see below for details.</p>
Response:	none
Notes:	<p>You can use the DIO command to activate/deactivate the Output 1 to Output 4 lines on the I/O socket (p. 264). The E-873 allows you to either set a single line per DIO command, or all lines at once.</p> <p>The <DIOID> 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 <OutputOn>.</p> <p>If <OutputOn>=1 the line is set to HIGH/ON, if <OutputOn>=0 it is set to LOW/OFF.</p> <p>Do not use DIO on output lines for which the trigger output is activated with TRO (p. 198).</p>

DIO? (Get Digital Input Lines)

Description:	Gets the states of the specified digital input lines.
	Use TIO? (p. 196) to get the number of available digital I/O lines.

Format:	DIO? [{<DIOID>}]
Arguments:	<DIOID> is the identifier of the digital input line, see below for details.
Response:	{<DIOID>=" "<InputOn> LF}
	where
	<InputOn> gives the state of the digital input line, see below for details.
Notes:	<p>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. 264).</p> <p>The <DIOID> identifiers to use for the lines are 1 to 4. If the identifier is omitted or 0, all lines are queried.</p> <p>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.</p>

DRC (Set Data Recorder Configuration)

Description:	Determines the data source to be used and the type of data to be recorded (record option) for the data recorder table given.
Format:	DRC {<RecTableID> <Source> <RecOption>}
Arguments:	<p><RecTableID> is one data recorder table of the controller, see below.</p> <p><Source> is the ID of the data source, for example, an axis or channel of the controller. The required source depends on the selected record option.</p> <p><RecOption> is the type of data to be recorded (record option).</p> <p>See the following list of the available record options and the corresponding data sources for details.</p>
Response:	none

Notes: The E-873 has 4 data recorder tables with 8192 points per table.

With HDR? (p. 150), you will obtain a list of all available record and trigger options and additional information on the data recording. The number of available data recorder tables can be read with TNR? (p. 198).

For further information, see "Data Recorder" (p. 78).

Recording options available with the corresponding data sources:

- 0=Nothing is recorded
- Data source is the axis:
- 1=Commanded position of axis
 - 2=Actual position of the 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

DRC? (Get Data Recorder Configuration)

Description: Gets the settings for the data to be recorded.

Format: DRC? [{<RecTableID>}]

Arguments: <RecTableID>: is a data recorder table of the controller; if this entry 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 type of data to be recorded (record option).

The available record options can be queried with HDR? (p. 150).

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. 137).

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:	<p><StartPoint> is the first point to be read from the data recorder table; it starts with index 1.</p> <p><NumberOfPoints> is the number of points to be read per table.</p> <p><RecTableID> is one data recorder table of the controller.</p>
Response:	For the recorded data in GCS array format, see the separate manual for the GCS array, SM146E, and the example below.
Notes:	<p>If <RecTableID> is not specified, the data is read from all tables with a record option not equal to zero.</p> <p>With HDR? (p. 150), you will obtain a list of all available recording and triggering options as well as additional information on data recording.</p> <p>For further information, see the description of the DRC command (p. 137) as well as "Data Recorder" (p. 78).</p>

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><RecTableID> is one data recorder table of the controller. See below for details.</p> <p><TriggerSource> ID of the trigger source, see below for a list of available options.</p> <p><Value> depends on the trigger source, can be a dummy, see below.</p>
Response:	none
Notes:	<p>At present, only 0 is valid for <RecTableID>; this means that the specified trigger source is set for all data recorder tables that have a record option that is not zero.</p> <p>Irrespective of the trigger option set, data recording is always triggered when step response measuring is done with STE (p. 193).</p> <p>With HDR? (p. 150), you will obtain a list of all available record and trigger options and additional information on the data recording.</p> <p>For further information, see the description of the DRC command (p. 137) as well as "Data Recorder" (p. 78).</p>
Available trigger options:	<p>0 = default setting Data recording is triggered by STE; <Value> must be a dummy.</p> <p>1 = any command changing target position e.g., MVR (p. 177), MOV (p. 175); <Value> must be a dummy.</p> <p>2 = next command resets trigger after execution; <Value> must be a dummy.</p> <p>6 = any command changing target position, reset trigger after execution e.g., MVR, MOV; resets trigger after execution; <Value> must be a dummy.</p>

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.

Further information can be found in the description of the DRT command (p. 141).

Notes: Since all data recorder tables of the E-873 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: Get error code <int> of the last occurred error and reset 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. 205).

Format: ERR?

Arguments: None

Response: The error code of the last error that occurred (integer).

Troubleshooting: Communication breakdown

- Notes:
- In the case of simultaneous access to the controller by several instances, the error code is only returned to the first instance that sent the ERR? command. Because the error is reset to 0 by the query, the error is not visible for any further querying instance.
- If possible, access the controller with one instance only.
 - If incorrect system behavior does not cause the controller to send an error code, check whether the error code is queried regularly in the background by a macro, script or the PC software (e.g., PIMikroMove).
- If the cause of an error continues, the corresponding error code is immediately set again after a query with ERR?.

FED (Find Edge)

- Description:
- Moves the 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. 145), FPL (p. 146), and FRF (p. 147) commands for referencing), 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 switch are disabled (see below); SVO? (p. 196) responds with the value 0.

Notes:	<p>Servo mode must be switched on with SVO (p. 195) for the commanded axis prior to using this command (closed-loop operation).</p>
	<p>The firmware of the E-873 determines the following based on parameters:</p>
	<ul style="list-style-type: none"> ▪ Is a reference switch present (parameter 0x14)? ▪ Are limit switches present (parameter 0x32)? ▪ Should the hard stops be used for reference moves (parameter 0x7A)? ▪ If the reference switch is represented by an index pulse: How should the move to the index pulse take place (parameters 0x70, 0x78, 0x79)?
	<p>According to the values of those parameters, the E-873 activates or deactivates FED motion to the corresponding signal edges. Adapt the parameter values to your hardware using SPA (p. 190) or SEP (p. 188). For further information, see "Adapting Settings" .</p>
	<p>You can use the digital input lines instead of the switches as source of the switch signals for FED. Refer to "Digital Input Signals" (p. 87) for further information.</p>
	<p>FED can be used to measure the physical travel range of new mechanics and therefore determine the values for the corresponding parameters:</p>
	<ul style="list-style-type: none"> ▪ Distance from the negative to the positive end of the travel range ▪ Gap between the negative end of the travel range and the reference switch (parameter 0x17) ▪ Gap between the reference switch and the positive end of the travel range (parameter 0x2F)
	<p>For further information, see "Travel Range and Soft Limits" (p. 29).</p>
	<p>Motion can be stopped with #24 (p. 124), STP (p. 194), and HLT (p. 164).</p>
	<p>Motion commands such as FED are not permitted when HID control is activated for the axis. See "Controlling with a Human Interface Device" (p. 91) for further information.</p>
Available edge types and parameters:	<p>The following edge types with their parameter settings are available:</p> <p>1 = negative travel range limit, <Param> must be 0 2 = positive travel range limit, <Param> must be 0</p>

3 = reference switch, <Param> must be 0

FNL (Fast Reference Move To Negative Limit)

Description:	<p>Does a reference move.</p> <p>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.</p> <p>If multiple axes are specified in the command, they are moved synchronously.</p>
Format:	FNL [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller, if omitted, all axes are affected.
Response:	None
Troubleshooting:	Illegal axis identifier
Notes:	<p>Servo mode must be switched on with SVO (p. 195) for the commanded axis prior to using this command (closed-loop operation).</p> <p>If the reference move was successful, absolute motion will then be possible in closed-loop operation.</p> <p>The negative physical limit of the travel range can be represented by the following items of the positioner:</p> <ul style="list-style-type: none"> ▪ Negative limit switch ▪ If the positioner does not have integrated limit switches: Negative hard stop <p>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).</p> <p>You can use a digital input instead of the negative limit switch as source of the negative limit switch signal for FNL. Refer to "Digital Input Signals" (p. 87) for further information.</p> <p>Motion can be stopped with #24 (p. 124), STP (p. 194), and HLT (p. 164).</p> <p>Use FRF? (p. 148) to check whether the reference move</p>

was successful.

For best repeatability, referencing must always be done 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.

For further information, see "Referencing" (p. 34) and "Travel Range and Soft Limits" (p. 29).

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:	<p>Servo mode must be switched on with SVO (p. 195) for the commanded axis prior to using this command (closed-loop operation).</p> <p>If the reference move was successful, absolute motion will then be possible in closed-loop operation.</p> <p>The positive physical limit of the travel range can be represented by the following elements of the positioner:</p> <ul style="list-style-type: none"> ▪ Positive limit switch ▪ If the positioner does not have integrated limit switches: Positive hard stop <p>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.</p> <p>You can use a digital input instead of the positive limit switch as source of the positive limit switch signal for FPL.</p>

Refer to "Digital Input Signals" (p. 87) for further information.

Motion can be stopped with #24 (p. 124), STP (p. 194), and HLT (p. 164).

Use FRF? (p. 148) to check whether the reference move was successful.

For best repeatability, referencing must always be done 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.

For further information, see "Referencing" (p. 34) and "Travel Range and Soft Limits" (p. 29).

FRF (Fast Reference Move To Reference Switch)

Description:	Starts a reference move.
	Moves the specified axis to the reference switch and sets the current position to a defined value. See below for details.
	If multiple axes are given in the command, they are started simultaneously.
Format:	FRF [{<AxisID>}]
Arguments:	<AxisID> is one axis of the controller, if omitted, all axes are affected.
Response:	None
Troubleshooting:	Illegal axis identifier
Notes:	<p>Servo mode must be switched on with SVO (p. 195) for the commanded axis prior to using this command (closed-loop operation).</p> <p>If the reference move was successful, absolute motion will then 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 switch.</p>

You can use a digital input instead of the reference switch as source of the reference signal for the FRF command. Refer to "Digital Input Signals" (p. 87) for further information.

Motion can be stopped with #24 (p. 124), STP (p. 194), and HLT (p. 164).

Use FRF? (p. 148) to check whether the reference move was successful.

Use FNL (p. 145) or FPL (p. 146) instead of FRF to do a reference move for an axis which has no reference switch.

For best repeatability, referencing must always be done in the same way.

For further information, see "Referencing" (p. 34) and "Travel Range and Soft Limits" (p. 29).

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 done with FNL (p. 145), FPL (p. 146) or FRF (p. 147) or when the position was set directly with POS (p. 182) (depending on the referencing method selected with RON (p. 184)).

GOH (Go To Home Position)

Description: Moves the given axis to the zero position.

GOH [{<AxisID>}]
is the same as
MOV {<AxisID> 0}

The motion can be stopped by #24 (p. 124), STP (p. 194),
and HLT (p. 164).

Format: GOH [{<AxisID>}]

Arguments: <AxisID>: Is one axis of the controller; if omitted, all axes
are affected.

Response: None

Troubleshooting: Illegal axis identifier

Notes: Servo mode must be switched on for the commanded axis
prior to using this command (closed-loop operation).

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-873 uses a parameter (ID 0x7A) to determine whether the hard stops of the axis can be used for reference moves. The E-873 activates or deactivates reference moves that use the hard stops according to the value of this parameter.

Adapt the parameter value to your hardware using SPA (p. 190) or SEP (p. 188). Further information, see "Referencing" (p. 34).

HDR? (Get All Data Recorder Options)

Description: Lists a help string which contains all information available on data recording (record options and trigger options, information on 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-873, the response to HDR? reads as follows:

```
#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
#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
#Additional information
4 record tables
8192 datapoints per table
end of help

```

Note: TriggerOptions = 0 (default) means that recording is triggered by the STE command (p. 193).

HDT (Set HID Default Lookup Table)

Description:	<p>Assigns a lookup table to the specified axis of the specified HID.</p> <p>Lookup tables are used while HID is controlling several motion variables of the E-873' axes, see HIA (p. 153) for details. A lookup table maps the displacement of an HID axis to the controlled motion variable (see HIE? (p. 155) for further details).</p>
Format:	HDT {<HIDDeviceID> <HIDDeviceAxis> <HIDTableID>}
Arguments:	<p><HIDDeviceID> is an HID connected to the controller; see below for more details.</p> <p><HIDDeviceAxis> is an HID axis; see below for more details.</p> <p><HIDTableID> is a controller's lookup table; see below for more details.</p>
Response:	None

Notes: Lookup tables are only assigned with HDT in the volatile memory (RAM) of the E-873. With the WPA command (p. 203), the currently valid assignment can be saved in the nonvolatile memory of the E-873.

 The E-873 supports one human interface device (identifier: 1). Information on the supported axes of the human interface device can be queried with the HIS? command (p. 157). For further information, see "Commandable Items" (p. 16).

Available lookup tables: The E-873 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. 160), user-defined lookup tables can be filled with values.

HDT? (Get HID Default Lookup Table)

Description: Queries the currently assigned lookup table for the specified axis of the specified HID.

Format: HDT? [{<HIDDeviceID> <HIDDeviceAxis>}]

Arguments: <HIDDeviceID> is an HID connected to the controller; see HDT for more details.

 <HIDDeviceAxis> is an HID axis; see HDT for more details.

Response: {<HIDDeviceID> <HIDDeviceAxis>="<HIDTableID>LF}

 where

 <HIDTableID> is a controller's lookup table; see HDT for more details.

HIA (Configure Control Done By HID Axis)

Description:	<p>Configures control of the E-873's axis by HID axes ("HID Control"):</p> <p>Assigns a specified motion variable of the specified E-873 axis to an HID axis.</p> <p>The HID control is activated or deactivated with the HIN command (p. 156). HIA can only be used when HID control is deactivated for the affected axis of the E-873.</p>
Format:	HIA {<AxisID> <MotionParam> <HIDDeviceID> <HIDDeviceAxis>}
Arguments:	<p><AxisID> is one axis of the controller.</p> <p><MotionParam> is a motion parameter of the controller's axis; see below for more information.</p> <p><HIDDeviceID> is an HID connected to the controller, see below for more details.</p> <p><HIDDeviceAxis> is an HID axis; see below for more details.</p>
Response:	None
Notes:	<p>HID control is only configured with HIA in the volatile memory (RAM) of the E-873. With the WPA command (p. 203), the currently valid configuration can be saved in the nonvolatile memory of the E-873.</p> <p>The E-873 supports one human interface device (identifier: 1). Information on the supported human interface device axes can be queried with the HIS? command (p. 157). For further information, see "Commandable Items" (p. 16).</p> <p><MotionParam> specifies the motion parameter to be controlled and can take on the following values:</p> <p>0 - delete configuration Deletes the current configuration of the HID control. Can be sent without specifying the <HIDDeviceID> and <HIDDeviceAxis> in abbreviated notation as: HIA <AxisID> 0</p> <p>2 - Relative target position The displacement of the HID axis determines the frequency for moving the controlled positioner axis.</p>

If an axis of the E-873 is configured for HID control (p. 91) of the relative target position: The current configuration must be deleted by sending the HIA command with a value of zero for <MotionParam> before a new configuration can be set.

If HID control is deactivated with the HIN command, it will have no effect in the following cases:

- <MotionParam> has the value zero, i.e., a function to be controlled has not been selected for the E-873's axis
- <HIDeviceID> has the value zero, i.e., human interface device is not selected for HID control
- <HIDeviceAxis> has the value zero, i.e., axis of a human interface device is not selected for HID control

HIA? (Get Configuration Of Control Done By HID Axis)

Description: Queries the current motion variable for the specified motion parameter of the specified E-873 axis, i.e., the currently assigned HID axis.

Format: HIA? [{<AxisID> <MotionParam>}]

Arguments: <AxisID> is one axis of the controller.

<MotionParam> is a motion parameter of the controller axis; see HIA for more details.

Response: {<AxisID> <MotionParam>="<HIDeviceID>
<HIDeviceAxis>LF}

where

<HIDeviceID> is an HID connected to the controller; see HIA for more details.

<HIDeviceAxis> is an HID axis; see HIA for more details.

HIB? (Get State Of HID Button)

Description: Queries the current state of the specified button of the specified HID.

Format: HIB? [{<HIDeviceID> <HIDeviceButton>}]

Arguments: <HIDeviceID> is an HID connected to the controller; see below for more details.

<HIDeviceButton> is an HID button; see below for more details.

Response: {<HIDeviceID> <HIDeviceButton> "="<HIDButtonState>}

where

<HIDButtonState> specifies the status of the button as 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. 157). 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.

HIE? (Get Deflection Of HID Axis)

Description: Queries the current displacement of the specified axis of the specified HID.

Format: HIE? [{<HIDeviceID> <HIDeviceAxis>}]

Arguments: <HIDeviceID> is an HID connected to the controller; see below for more details.

<HIDeviceAxis> is an HID axis; see below for more details.

Response: {<HIDeviceID> <HIDeviceAxis> "="<HIDDeflection>}

where

<HIDDeflection> specifies the current HID axis displacement, see below for more details.
 <HIDDeflection> specifies the current HID axis displacement as floating point number in a range from -1.0 to 1.0 an.

For HID 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-873 prepares the information received by the HID so that 256 different displacement values can be shown. When HID control for a motion variable is based on lookup tables, each of the displacement values is assigned to exactly one point in the currently assigned lookup table (see HDT (p. 151) and HIT (p. 160) for more details).

Example:

Send:	<code>HIE? 1 1 1 2</code>
Receive:	<code>1 1=0.02</code> <code>1 2=-0.7</code>
Note:	Displacement of axes 1 and 2 of HID 1: Axis 1 has the value 0.02, which corresponds to approximately the center 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: Activates or deactivates control of the specified E-873 axis by HIDs ("HID control") connected to the controller.

The HID control is configured with the HIA command (p. 153).

Format: HIN {<AxisID> <HIDControlState>}

Arguments: <AxisID> is one axis of the controller.

<HIDControlState> is the HID control activation status:

0 = control by HID is deactivated

1 = control by HID is activated

Response: None

- Notes:
- The E-873 supports an HID (identifier 1). See "Connecting an HID" (p. 45) for information on the connection options.
 - The deactivated HID control will not have any effect if it has not been configured suitably with HIA.
 - When HID control is deactivated, the target position is set to the current position of the controlled axis.
 - No HID control is possible in open-loop operation (servo mode switched off).
 - Motion commands such as MOV (p. 175) are not permitted when HID control is activated for the axis. See "Controlling with an HID" (p. 91) for more information.

HIN? (Get Activation State Of HID Control)

- Description: Queries the activation status of the control by HID ("HID control") connected to the controller for the specified axis of the E-873.
- Format: HIN? [{<AxisID>}]
- Arguments: <AxisID> is one axis of the controller.
- Response: {<AxisID>=" "<HIDControlState>LF}
- where
- <HIDControlState> is the HID control activation status:
 0 = control by HID is deactivated
 1 = control by HID is activated

HIS? (Get Configuration Of HI Device)

- Description: Queries the specified property for the specified operating element of an HID.
- Format: HIS? [{<HIDDeviceID> <HIDItemID> <HIDPropID>}]

Arguments:	<p><HIDDeviceID> is an HID connected to the controller; see below for more details.</p> <p><HIDItemID> is an HID operating element, see below for more details.</p> <p><HIDPropID> is a property of an HID operating element; see below for more details.</p>
Response:	<pre>{<HIDDeviceID> <HIDItemID> <HIDPropID>="<HIDPropValue>LF}</pre> <p>where</p> <p><HIDPropValue> is a string with the value that the property of the operating element is set to; see below for more details.</p>
Supported operating elements and their properties	<p>The E-873 supports one human interface device (identifier 1). All supported operating elements of the human interface device are numbered consecutively for <HIDItemID> starting with 1, irrespective of their type.</p> <p>The following properties of the human interface device are shown with HIS?:</p> <p>For <HIDPropID> = 1:</p> <p><HIDPropValue> indicates the type and identifier of the operating element. Possible types:</p> <ul style="list-style-type: none"> ▪ "Axis" = axis of the human interface device ▪ "Button" = button of the human interface device <p>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.</p> <p>For <HIDPropID> = 2:</p> <p><HIDPropValue> is the value for the current status of the operating element. The meaning of the value depends on the type of operating element:</p> <ul style="list-style-type: none"> ▪ "Axis": Current displacement of the axis; for more details, see HIE? (p. 155) ▪ "Button": Current state of the button; for more details, see HIB? (p. 154) <p>For <HIDPropID> = 3:</p>

<HIDPropValue> is the name of an axis of the human interface device

For <HIDPropID> = 4:

<HIDPropValue> is the name of the human interface device

For <HIDPropID> = 5:

<HIDPropValue> indicates the smallest possible value for the status of an operating element of the "Button" type

For <HIDPropID> = 6:

<HIDPropValue> indicates the largest possible value for the status of an operating element of the "Button" type

Example:

```
HIS?
1 1 1=Axis_1
1 1 2=0.031
1 1 3=X
1 1 4=USB
1 2 1=Axis_2
1 2 2=0.023
1 2 3=Y
1 2 4=USB
1 3 1=Axis_3
1 3 2=0.031
1 3 3=Z
1 3 4=USB
1 4 1=Button_1
1 4 2=1
1 4 4=USB
1 4 5=0
1 4 6=1
1 5 1=Button_2
1 5 2=0
1 5 4=USB
1 5 5=0
1 5 6=1
1 6 1=Button_1
1 6 2=1
1 6 4=USB
1 6 5=0
1 6 6=1
```

HIT (Fill HID Lookup Table)

Description: Fills the specified lookup table with values.

Lookup tables are used while HID is controlling several motion variables of the E-873' axes, see HIA (p. 153) for details. A lookup table maps the displacement of an HID axis to the controlled motion variable (see HIE? (p. 155)) for further details.

The HDT (p. 151) command assigns the lookup tables to HID axes.

Format: HIT {<HIDTableID> <HIDTableAddr> <HIDTableValue>}

Arguments: <HIDTableID> is a controller's lookup table; see below for more details.

<HIDTableAddr> is the index of a point in the lookup table, begins with 1, see below for number of points per table.

<HIDTableValue> is the value of point n as a floating point number in the a from -1.0 to 1.0; see below for further information.

Response: None

Notes: HIT only fills the lookup tables in the volatile memory (RAM) of the E-873. With the WPA command (p. 203), the currently valid table contents can be saved in the nonvolatile memory of the E-873.

The value of one point can be sent to the E-873 per HIT command.

Available lookup tables: The E-873 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 identifier δ 100 identifier are predefined and write-protected.

The first point of a lookup table corresponds to the maximum axis displacement of the HID 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.

When HID control is activated, the direction of motion for the E-873's axis can be reversed with the ***Invert Direction Of Motion For Joystick-Controlled Axis?*** parameter (ID 0x61).

HIT? (Get HID Lookup Table Values)

Description: Queries the values of the specified points in the specified 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> specifies the number of the points to be queried per lookup table; see HIT for more details.

<HIDTableID> is a controller's lookup table; see HIT for more details.

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 further details, see the notes below.

Error code 10 is set.

#24 (p. 124) and STP (p. 194) 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: HLT stops motion with given system deceleration with regard to system inertia.

HLT stops all motion caused by motion commands (e.g., MOV (p. 175), MVR (p. 177), GOH (p. 149), STE (p. 193), OSM (p. 181), OMA (p. 178), OMR (p. 179)), commands for referencing (FNL (p. 145), FPL (p. 146), FRF (p. 147)) and macros (MAC (p. 169)).

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 that contains all available parameters with short descriptions. Refer to "Parameter Overview" (p. 239) for further information.

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 the following format:

```
<CmdLevel>TAB<MaxItem>TAB<DataType>TAB<FunctionGroupDescription>TAB<ParameterDescription>[<PossibleValue>=" "<ValueDescription>]
```

where

<CmdLevel> is the command level that 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-873, an "item" is an axis or the entire system.

<DataType> is the data type of the parameter value; it can be INT, FLOAT, or CHAR.

<FunctionGroupDescription> is the name of the function group which the parameter belongs to.

<ParameterDescription> is the parameter name.

<PossibleValue> is one value from the permissible 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. 190) influences the parameter settings in volatile memory (RAM).

WPA (p. 203) copies parameter settings from volatile to nonvolatile memory.

SEP (p. 188) writes parameter settings directly into nonvolatile memory (without changing settings in volatile memory).

RPA (p. 185) resets volatile memory to the values from nonvolatile memory.

IFS (Set Interface Parameters as Default Values)

Description:	Stores interface parameters. The default parameters for the interface are changed in the nonvolatile memory, but the currently active parameters are not. Settings made with IFS become active with the next switching on or reboot.
Format:	IFS <Pswd> {<InterfacePam> <PamValue>}
Arguments:	<p><Pswd> is the password for writing to the nonvolatile memory, default is "100"</p> <p><InterfacePam> is the interface parameter to be changed, see below</p> <p><PamValue> gives the value of the interface parameter, see below</p>
Response:	None
Parameters and values:	<p><InterfacePam>: Interface parameters</p> <p>IPSTART: Startup behavior for configuring the IP address for TCP/IP communication:</p> <ul style="list-style-type: none"> ▪ 0 = The address is used that is defined by IPADR. ▪ 1 = DHCP is used to get the IP address. <p>IPADR: IP address with port for TCP/IP communication Format: <IP address>:<port number></p> <p>IPMASK: Subnet mask for TCP/IP communication Formats:</p> <ul style="list-style-type: none"> ▪ uint.uint.uint.uint e.g.: 255.255.0.0 ▪ <decimal number> = Number of bits of the subnet mask (specifies how

many bits make up the beginning of the IP address in the network component)
e.g.: 16

IPGTWAY: Standard gateway for TCP/IP communication
Format: uint.uint.uint.uint

<PamValue>: Possible and default values for the interface parameters are:

IPADR
Default port: 50000, cannot be changed
Note: The E-873 will use the address specified by IPADR only if IPSTART = 0.

IPMASK:
Default: 24 (= subnet mask 255.255.255.0)

IFS? (Get Interface Parameters as Default Values)

Description: Gets the parameter values of the interface configuration stored in the nonvolatile memory (i.e. default settings)

Format: IFS? [{<InterfacePam>}]

Arguments: <InterfacePam> is the interface parameter to be queried. See below for possible values.

Response: {<InterfacePam>="<PamValue> LF}

where

<PamValue> is the value of the interface parameter in the nonvolatile memory.

Parameters and values: <InterfacePam>: Interface parameters

IPSTART: Startup behavior for configuring the IP address for TCP/IP communication:

- 0 = The address is used that is defined by IPADR.
- 1 = DHCP is used to get the IP address.

IPADR: IP address with port for TCP/IP communication
Format: <IP address>:<port number>

IPMASK: Subnet mask for TCP/IP communication

Formats:

- uint.uint.uint.uint
e.g.: 255.255.0.0
- <decimal number>
= Number of bits of the subnet mask (specifies how many bits make up the beginning of the IP address in the network component)
e.g.: 16

IPGTWAY: Standard gateway for TCP/IP communication

Format: uint.uint.uint.uint

MACADR: Ethernet address (= unchangeable, unique address of the network hardware) in the E-873

JRC (Jump Relatively Depending On Condition)

Description: Jumps relatively depending on a specified condition of the following type: one specified value is compared with a queried value according to a specified 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. 202). 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 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

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 (query ONT?). When the stop button is pressed before the target position has been reached: The response to the POS? query is copied into the TARGET variable. This variable is then used as second argument for the MOV command. Therefore, the positioner just stays where it 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

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 <i>macroname</i> 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 startup macro. This macro will be automatically executed after the next switch-on or reboot of the controller. If <macroname> is omitted, the current startup macro selection is canceled.
	MAC DEF?
	Asks for the startup macro
	Response: <macroname>
	If no startup 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. 116) 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. For further information, see "Working with Macros" (p. 102) and "Target and Sender Address" (p. 116).

The **MAC BEG** and **MAC END** commands may not be specified when macros are recorded in the **Controller macros** tab in PIMikroMove.

A macro can be overwritten by a macro with the same name.

Macros can contain local and global variables. For further information, see "Variables" (p. 116).

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-873 can only be used in macros:

DEL (p. 133), JRC (p. 168), MEX (p. 174) and WAC (p. 202).

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 while 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. 124) and STP (p. 194).

It is not possible to run several macros simultaneously. Only one macro can be run at a time.

A macro cannot be deleted while it is running.

You can query with #8 (p. 123) 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: A detailed help text can be displayed for the following GCS commands:
CTO, CTO?, HIA, HIA?, HIS, HIS?, HIT, HIT?, WPA

Example: Send: MAN? CTO?
Receive:
CTO {<TrigOutID> <CTOPam> <Value>} Set Configuration Of Trigger Output
#AvailableCTOparameters
<CTOPam> <Description>
1 Trigger Step
2 Axis
3 Trigger Mode
7 Polarity
8 Start Threshold
9 Stop Threshold
10 Trigger Position
#AvailableTriggerModes
<Value> <Description>
0 Position Distance
2 On Target
5 Motion Error
6 In Motion

```

7 Position+Offset
8 Single Position
#AvailablePolarities
<Value> <Description>
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. 202).

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:

LOOP macro contains the following:

```
MAC START KEY1
MAC START KEY2
MEX DIO? 4 = 1
MAC START LOOP
```

KEY1 macro contains the following:

```
MEX DIO? 4 = 1
MEX DIO? 1 = 0
MVR 1 1.0
DEL 100
```

KEY2 macro contains the following:

```
MEX DIO? 4 = 1
MEX DIO? 2 = 0
MVR 1 -1.0
DEL 100
```

LOOP macro forms an infinite loop by permanently calling KEY1, KEY2 and itself.

KEY1 checks the state of the digital input channel 1 (is on the I/O socket). 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.

By connecting the digital input channels 1, 2 and 4 with pushbuttons, e.g., with the C-170.PB pushbutton box, it is possible to realize interactive control of an axis without any software assistance. The delay (DEL 100) is required to avoid generating multiple MVR commands while pressing the pushbutton for a short time.

Channel 4 is used as a global exit. Since MEX only stops execution of the current macro, it must also be included in the calling macro, which would otherwise continue.

MOV (Set Target Position)

Description:	Sets an absolute target position for the given axis.
Format:	MOV {<AxisID> <Position>}
Arguments:	<p><AxisID> is one axis of the controller.</p> <p><Position> is the absolute target position in physical units.</p>
Response:	none
Notes:	<p>The servo mode must be switched on when this command is used (closed-loop operation).</p> <p>The target position must be inside the soft limits. Use TMN? (p. 197) and TMX? (p. 197) to query the current valid soft limits.</p> <p>The motion can be stopped by #24 (p. 124), STP (p. 194), and HLT (p. 164).</p> <p>During 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 permitted when HID control is activated for the axis. For further information, see "Controlling with an HID" (p. 91).</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:	<p>Send: <code>MOV 1 243</code></p> <p>Send: <code>ERR?</code></p> <p>Receive: <code>7</code></p> <p>Note: The axis does not move. The error code "7" in the reply to the ERR? command (p. 142) indicates that the target position given in the motion command is out of limits.</p>

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:	<p>The target position can be changed by commands that cause motion (e.g., MOV (p. 175), MVR (p. 177), MVE, GOH (p. 193), STE (p. 149)) or by HID control (when HID control is disabled, the target position is set to the current position for HID-controlled axes in closed-loop operation).</p> <p>MOV? gets the commanded positions. Use POS? (p. 182) to get the current positions.</p>

MVR (Set Target Relative To Current Position)

Description:	Moves the given axis relative to the last commanded target position.
Format:	MVR {<AxisID> <Distance>}
Arguments:	<p><AxisID> is one axis of the controller.</p> <p><Distance> gives the distance that the axis is to move; the sum of the distance and the last commanded target position is set as the new target position (in physical units).</p>
Response:	none
Notes:	<p>The servo mode must be switched on when this command is used (closed-loop operation).</p> <p>The target position must be inside the soft limits. Use TMN? (p. 197) and TMX? (p. 197) to get the currently valid soft limits, and MOV? (p. 176) to get the current target.</p> <p>Motion can be stopped by #24 (p. 124), STP (p. 194), and HLT (p. 164).</p>

During 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.

Motion commands such as MVR are not permitted when HID control is activated for the axis. For further information, see "Controlling with an HID" (p. 91).

Example:

Send: `MOV 1 0.5`

Note: This is an absolute motion.

Send: `POS? 1`

Receive: `1=0.500000`

Send: `MOV? 1`

Receive: `1=0.500000`

Send: `MVR 1 2`

Note: This is a relative motion.

Send: `POS? 1`

Receive: `1=2.500000`

Send: `MVR 1 2000`

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`

OMA (Absolute Open-Loop Motion)

Description: Moves the specified axis to the specified absolute position. Motion is performed in open-loop nanostepping mode.

Servo mode must be switched off for the commanded axis when using this command (open-loop operation).

Position control is not done with OMA (i.e., the target position is not held in the servo loop).

Depending on the drive type of the positioner, axis overshoot is possible due to the dynamics profile of the E-873 (velocity, acceleration). The controller compensates this by moving the axis back the corresponding number of steps.

Format:	OMA {<AxisID> <Position>}
Arguments:	<AxisID>: Axis of the controller
	<Position>: New absolute target position; in physical units, format: Floating point number
Response:	None
Troubleshooting:	Invalid axis identifier; servo mode for the axis is not switched off; HID control (joystick) is active for the axis
Notes:	<p>The velocity for open-loop nanostepping mode is controlled by the step frequency. The step frequency is determined by the amplitude of the input voltage.</p> <p>Motion commands such as OMA, OMR, OSM are not permitted when HID control is active for the axis. See "Controlling with HID" (p. 91) for further information.</p> <p>The motion can be stopped by #24 (p. 124), STP (p. 194), and HLT (p. 164).</p>

OMA? (Get Open-Loop Target Position)

Description:	<p>Queries the last valid commanded target position for open-loop operation.</p> <p>The target position for open-loop operation is changed with OMA (p. 178) and OMR (p. 179).</p>
Format:	OMA? {[<AxisID>]}
Arguments:	<AxisID>: Axis of the controller
Response:	{<AxisID>="<Position> LF}
	<p>where</p> <p><Position> is the last valid commanded target position for open-loop operation; in physical units, format: Floating point number.</p>

OMR (Relative Open-Loop Motion)

Description:	<p>Moves the specified axis relative to the last commanded target position for open-loop operation. Motion is performed in open-loop nanostepping mode.</p> <p>Servo mode must be switched off for the commanded axis when using this command (open-loop operation).</p> <p>Position control is not done with OMR (i.e., the target position is not held in the servo loop).</p> <p>Depending on the drive type of the positioner, axis overshoot is possible due to the dynamics profile of the E-873 (velocity, acceleration). The controller compensates this by moving the axis back the corresponding number of steps.</p>
Format:	OMR {<AxisID> <Distance>}
Arguments:	<p><AxisID>: Axis of the controller</p> <p><Distance> specifies the distance that the axis is to move; the sum of the distance and the last commanded target position is set as the new target position (in physical units, format: Floating point number).</p>
Response:	None
Troubleshooting:	Invalid axis identifier; servo mode for the axis is not switched off; HID control (joystick) is active for the axis
Notes:	<p>The velocity for open-loop nanostepping mode is controlled by the step frequency. The step frequency is determined by the amplitude of the input voltage.</p> <p>Motion commands such as OMA, OMR, OSM are not permitted when HID control is active for the axis. See "Controlling with HID" (p. 91) for further information.</p> <p>The motion can be stopped by #24 (p. 124), STP (p. 194), and HLT (p. 164).</p>

ONT? (Get On-Target State)

Description:	<p>Gets the on-target state of the given axis.</p> <p>If all arguments are omitted, gets state of all axes.</p>
Format:	ONT? [{<AxisID>}]

Arguments:	<AxisID> is one axis of the controller.
Response:	{<AxisID>=" "<uint> LF} where <uint> = "1" when the given axis has reached the target value, otherwise "0".
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. 26).

OSM (Open-Loop Step Moving)

Description:	Moves the specified axis by the specified number of steps. Servo mode must be switched off when using this command for the commanded axis (open-loop operation).
Format:	OSM {<AxisID> <Value>}
Arguments:	<AxisID>: Axis of the controller <Value>: Number of steps to be performed
Response:	None
Troubleshooting:	Invalid axis identifier; servo mode for the axis is not switched off; HID control (joystick) is active for the axis
Notes:	The velocity for open-loop nanostepping mode is controlled by the step frequency. The step frequency is determined by the amplitude of the input voltage. Motion commands such as OMA, OMR, OSM are not permitted when HID control is active for the axis. See "Controlling with HID" (p. 91) for further information. The motion can be stopped by #24 (p. 124), STP (p. 194), and HLT (p. 164).

OSN? (Read Left Steps)

Description:	Queries the number of steps that still have to be performed by the specified axis. With this command, it is possible to determine how many steps the commanded axis still has to perform for motion in open-loop operation before it arrives at its target position.
Format:	OSN? {<AxisID>}
Arguments:	<AxisID>: Axis of the controller
Response:	{<AxisID>=" "<uint> LF} where <uint> is the number of steps that still have to be performed.

POS (Set Real Position)

Description:	Sets the current position of the axis (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 referencing method "0" is selected, see RON (p. 184).
 - An axis is considered to be "referenced" when the position has been set with POS (for more information, see "Referencing" (p. 34)).
 - The minimum and maximum commandable positions (TMN? (p. 197), TMX? (p. 197)) are not adapted when a position is set with POS. This can result in target positions which are allowed by the E-873 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-873. 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. The controller behaves the same as after switching on.
- Format: RBT
- Arguments: none

Response: none

Notes: RBT cannot be used in macros. This is to avoid problems with startup 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: Selects the referencing method for the specified axes

Format: RON {<AxisID> <ReferenceOn>}

Arguments: <AxisID> is one axis of the controller.

<ReferenceOn> is the referencing method. Can be 0 or 1. 1 is default. See below for details.

Response: None

Troubleshooting: Illegal axis identifier

Notes:

<ReferenceOn> = 0: An absolute position value can be assigned with POS (p. 182) or a reference move can be started with FRF (p. 147), FNL (p. 145) or FPL (p. 146). Relative motion with MVR is possible, even when referencing has not been done for the axis.

<ReferenceOn> = 1: A reference move for the axis must be started with FRF, FNL or FPL. Using POS is not allowed. Motion in closed-loop operation is only possible when the axis has been referenced.

For further information, see "Referencing" (p. 34) and "Travel Range and Soft Limits" (p. 29).

RON? (Get Reference Mode)

Description: Gets referencing method of specified axes.

Format: RON? [{ <AxisID>}]

Arguments: <AxisID> is one axis of the controller.

Response: {<AxisID>=" "<ReferenceOn> LF}

where

<ReferenceOn> is the currently selected referencing method for the axis

Troubleshooting: Illegal axis identifier

Note: Further information can be found in the description of the RON command (p. 184).

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. 164) you can obtain a list of the available parameters. SPA (p. 190) influences the parameter settings in volatile memory, WPA (p. 203) writes parameter settings from volatile to nonvolatile memory, and SEP (p. 188) 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 positioner's ID chip and the positioner databases are loaded into the volatile memory of the E-873. The loaded data is overwritten by RPA. Only use RPA if you are sure that the E-873 functions correctly with the parameter values from the nonvolatile memory.
	With the E-873, 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. 187). For further information, see "Commandable Items" (p. 16).
	Valid parameter IDs are specified in "Parameter Overview" (p. 239).

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:	<p>The duration of the recording can be calculated as follows:</p> $\text{Rec. duration} = \text{cycle time of the servo loop} * \text{RTR value} * \text{number of points}$ <p>where</p> <p>the cycle time of the servo loop for the E-873 is 100 μs</p> <p>the number of points for the E-873 is 8192 (length of data recorder table)</p> <p>For further information, see "Data Recorder" (p. 78).</p> <p>The record table rate set with RTR is saved in volatile memory (RAM) only.</p>

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:	<p>Sets the axis identifiers for the given axes.</p> <p>After it was set with SAI, the new axis identifier must be used as <AxisID> in all axis-related commands.</p>
Format:	SAI {<AxisID> <NewIdentifier>}
Arguments:	<p><AxisID> is one axis of the controller</p> <p><NewIdentifier> 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. 200) to ask for valid characters.</p> <p>The new axis identifier is saved automatically and is therefore still available after rebooting or switching on the next time.</p>

SAI? (Get List Of Current Axis Identifiers)

Description:	<p>Queries the axis identifiers.</p> <p>See also "Commandable Items" (p. 16).</p>
Format:	SAI? [ALL]
Arguments:	[ALL] is optional. For controllers that allow deactivating the axis, [ALL] ensures that the response also includes the axes that are "deactivated".
Response:	{<AxisID> LF}
Notes:	<p><AxisID> is one axis of the controller.</p> <p>If the Stage Name 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.</p>

SEP (Set Non-Volatile 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. 185) 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. 164) returns a list of the available parameters.

SPA (p. 190) writes parameter settings into volatile memory (without changing the settings in nonvolatile memory).

WPA (p. 203) writes parameter settings from volatile to nonvolatile memory.

Format: SEP <Pswd> {<ItemID> <PamID> <PamValue>}

Arguments

<Pswd> is the password for writing to the nonvolatile memory; the default value is "100".

<ItemID> is the item for which a parameter is to be changed in the nonvolatile memory. See below for details.

<PamID> is the parameter identifier, 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:	<p>Note that the number of write cycles in the nonvolatile memory is limited. Write default settings only if necessary.</p> <p>With the E-873 you can only write one parameter per SEP command.</p>
Available item IDs and parameter IDs:	<p>An item is an axis (the identifier can be changed with SAI (p. 187)) or the entire system. For further information, see "Commandable Items" (p. 16).</p> <p>Valid parameter IDs are given in "Parameter Overview" (p. 239).</p>
SEP? (Get Nonvolatile Memory Parameters)	
Description:	<p>Gets the value of a parameter of a given item from nonvolatile memory.</p> <p>With HPA? (p. 164) you can obtain a list of the available parameters and their IDs.</p>
Format:	SEP? [{<ItemID> <PamID>}]
Arguments:	<p><ItemID> is the element for which a parameter value from nonvolatile memory is to be queried. See below for details.</p> <p><PamID> is the parameter ID, can be written in hexadecimal or decimal format. See below for details.</p>
Response:	<p>{<ItemID> <PamID>="<PamValue> LF}</p> <p>where</p> <p><PamValue> is the value of the specified parameter for the specified element</p>
Troubleshooting:	Illegal item identifier, wrong parameter ID
Notes:	With the E-873, 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. 187)) or the entire system. For further information, see "Commandable Items" (p. 16).

Valid parameter IDs are given in "Parameter Overview" (p. 239).

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: <ItemID> is the element for which a parameter is changed in volatile 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 specified parameter of the specified element is set.

Response: None

Parameter changes are also lost when the parameters are reset to their default values with RPA (p. 185).

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. 164) returns a list of the available parameters.

SEP (p. 188) writes parameter settings directly into nonvolatile memory (without changing the settings in volatile memory).

WPA (p. 203) writes parameter settings from volatile to nonvolatile memory.

RPA resets volatile memory to the value in nonvolatile memory.

Troubleshooting: Illegal element identifier, wrong parameter ID, value out of range

Notes: With the E-873, you can write only one parameter per SPA command.

Available item IDs and parameter IDs: An item is an axis (the identifier can be changed with SAI (p. 187)) or the entire system. For further information, see "Commandable Items" (p. 16).

Valid parameter IDs are specified in the parameter overview (p. 239).

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 mechanics.

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. 66)). 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. 203) for details on the extent of the saved settings.

SPA? (Get Volatile Memory Parameters)

Description: Queries the value of a parameter of a specified item from volatile memory (RAM).

	You can obtain a list of the available parameters with HPA? (p. 164).
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 specified parameter for the specified item
Troubleshooting:	Illegal item identifier, wrong parameter ID
Notes:	With the E-873, you can query either all parameters or specific individual parameters for each SPA? command.
Available item IDs and parameter IDs:	An item is an axis (the identifier can be changed with SAI (p. 187)) or the entire system. For further information, see "Commandable Items" (p. 16).
	Valid parameter IDs can be found in the parameter overview (p. 239).

SRG? (Query Status Register Value)

Description:	Returns register values for queried items and registers.
Format:	SRG? [{<ItemID> <RegisterID>}]
Arguments:	<ItemID> is the item for which a register is to be queried. See below for details.
	<RegisterID> is the ID of the specified register; for available registers, see below.
Response:	{<ItemID><RegisterID>="<Value> LF}
	where
	<Value> is the value of the register; for more details, see below.

Note: This command is identical in function to #4 (p. 122) which should be preferred when the controller is performing time-consuming tasks.

Possible register IDs and response values: <ItemID> is one axis of the controller.
<RegisterID> can be 1.

<Value> is the bit-encoded response and is returned as the sum of the following individual codes in hexadecimal format:

Bit	15	14	13	12	11	10	9	8
Description	On-target state	Is referencing	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 switch	Negative limit switch

Example: Send: SRG? 1 1
Receive: 1 1=0x9002

Note: The response is in hexadecimal format. It means that axis 1 is on target (on-target state = true), the servo mode is ON for that axis, no error has occurred, the states of digital input lines 1 to 4 are low, and axis 1 is on the positive side of the reference switch.

STE (Start Step And Response Measurement)

Description: Starts a step and records the step response for the specified axis.

The data recorder configuration, i.e., the assignment of data sources and record options to the recorder tables, can be set with DRC (p. 137).

The recorded data can be read with the DRR? command (p. 139).

Format:	STE <AxisID> <Amplitude>
Arguments:	<p><AxisID> is one axis of the controller</p> <p><Amplitude> is the size of the step. See below for details.</p>
Response:	None
Troubleshooting:	In closed-loop operation, the target position must be inside the soft limits. Use TMN? (p. 197) and TMX? (p. 197) to get the currently valid soft limits, and MOV? (p. 176) to get the current target.
Notes:	<p>Motion commands such as STE are not permitted when HID control is activated for the axis. See "Controlling with a Human Interface Device" (p. 91) for further information.</p> <p>A "step" consists of motion with the specified amplitude which is performed relative to the current position. How the value for <Amplitude> is interpreted depends on the current servo mode:</p> <ul style="list-style-type: none"> ▪ Closed-loop operation: <Amplitude> specifies the distance for the step (in physical units). Floating point numbers are allowed. ▪ Open loop operation: <Amplitude> specifies the number of steps to be moved (execution in step mode (p. 19)) and must be an integer value. The step frequency is specified by the value of parameter 0x1F000400.

STP (Stop All Axes)

Description:	<p>Stops all axes abruptly. For further details, see the notes below.</p> <p>Sets error code to 10.</p> <p>This command is identical in function to #24 (p. 124).</p>
Format:	STP
Arguments:	None
Response:	None
Troubleshooting:	Communication breakdown

Notes: STP stops all motion caused by motion commands (e.g., MOV (p. 175), MVR (p. 177), GOH (p. 149), STE (p. 193), OSM (p. 181), OMA (p. 178), OMR (p. 179)), commands for referencing (FNL (p. 145), FPL (p. 146), FRF (p. 147)) and macros (MAC (p. 169)). 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 actual servo mode state determines the usable motion commands:

Servo mode on: Use the MOV (p. 175), MVR (p. 177), GOH (p. 149), STE (p. 193) commands or HID control (p. 91).
 Servo mode off: Use STE (p. 193), OSM (p. 181), OMA (p. 178) or OMR (p. 179).

The servo mode must be switched on before reference moves can be started with FRF (p. 147), FNL (p. 145) or FPL (p. 146).

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 when switching on or rebooting. For more information, see "Setting up a startup macro" (p. 107).

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

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. 136). Furthermore, you can program the Output 1 to Output 4 lines using the CTO command (p. 130) (trigger configuration) and the TRO command (p. 198) (trigger enabling/disabling). The digital input lines reported by TIO? are Input 1 to Input 4. They can be read with DIO? (p. 136), #4 (p. 122) and SRG? (p. 192). In addition, you can use the Input 1 and 2 or

Input 3 and 4 lines for HID control of the E-873's axis. See HIA (p. 153) and "Connecting an HID" (p. 45) for details.

All line are on the E-873's **I/O** socket.

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. When redefining the zero position with the DFH (p. 134) command, the minimum commandable position is automatically adapted to the new zero position.

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. When redefining the zero position with the DFH (p. 134) command, the maximum commandable position is automatically adapted to the new zero position.

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-873 has 2 data recorder tables with 1024 data points per table.

For more information see "Data Recorder".

TRO (Set Trigger Output State)

Description: Activates or deactivates the trigger output conditions set with CTO (p. 130) 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 deactivated
1 = Trigger output activated

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".

Do not use DIO (p. 136) on digital output lines where the trigger output is activated by TRO.

TRO? (Get Trigger Output State)

Description: Queries the activation status of the trigger output configuration made with CTO (p. 130) for the specified digital output line.

If arguments are not specified, gets state of all digital

output lines.

Format:	TRO? [{<TrigOutID>}]
Arguments:	<TrigOutID> is one digital output line of the controller, see TRO (p. 198) for more details.
Response:	{<TrigOutID>="<TrigMode> LF}
	where
	<TrigMode> is the current state of the digital output line: 0 = Trigger output deactivated 1 = Trigger output activated
Troubleshooting:	Illegal identifier of the digital output line

TRS? (Indicate Reference Switch)

Description:	Indicates whether axes have a reference 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 switch (=1) or not (=0).
Troubleshooting:	Illegal axis identifier
Notes:	<p>The E-873 firmware detects the presence or absence of a reference switch via a parameter (ID 0x14). The E-873 activates or deactivates reference moves to the reference switch (FRF command (p. 147)) according to the value of this parameter. Adapt the parameter value to your hardware using SPA (p. 190) or SEP (p. 188). For further information, see "Reference Switch Detection" (p. 27).</p> <p>You can use a digital input line instead of the reference switch as source of the reference point signal for the FRF command. For further information, see "Digital Input Signals" (p. 87).</p>

TVI? (Tell Valid Character Set For Axis Identifiers)

Description: Returns a string with characters which can be used for axis identifiers.

Use SAI (p. 187) to change the axis identifiers and SAI? (p. 187) to ask for the current valid axis identifiers.

Format: TVI?

Arguments: None

Response: <string> is a list of characters

Notes: With the E-873, 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. 116) for more details on 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. 116) for more details on 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 braces 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. 125) for another example.

VAR? (Get Variable Values)

Description:	Gets values of variables.
	If VAR? is combined with CPY (p. 128), JRC (p. 168), MEX (p. 174) or WAC (p. 202), the response to VAR? has to be a single value and not more.
	See "Variables" (p. 116) for more details on local and global variables.
Format:	VAR? [{<Variable>}]
Arguments:	<p><Variable> is the name of the variable to be queried. See "Variables" (p. 116) for more details on name conventions.</p> <p>If <Variable> is omitted, all global variables present in the RAM are listed.</p>
Response:	<p>{<Variable>}"="<String>LF}</p> <p>where</p> <p><String> gives the value to which the variable is set.</p>
Notes:	Local variables can be queried using VAR? only when a macro with local variables is running. See "Variables" (p. 116) for details regarding local and global variables.
Example:	See ADD (p. 125) for an example.

VER? (Get Versions Of Firmware And Drivers)

Description: Gets the versions of the firmware of the E-873 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. 174).

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. 185) is used to restore the parameters.

With HPA? (p. 164) you can obtain a list of all available parameters.

Use SPA? (p. 190) to check the current parameter settings in volatile memory.

See SPA (p. 190) for an example.

Format: WPA <Pswd> [{<ItemID> <PamID>}]

Arguments: <Pswd> is the password for writing to the nonvolatile memory. See below for details.

<ItemID> is the item for which a parameter is to be saved from the volatile to the nonvolatile memory. See below for details.

<PamID> is the parameter ID, can be written in hexadecimal or decimal format. See below for details.

Response:	None						
Troubleshooting:	<p>Illegal element identifier, wrong parameter ID, invalid password</p> <p>Note that the number of write cycles in the nonvolatile memory is limited. Write default settings only if necessary.</p>						
Notes:	<p>Parameters can be changed in the volatile memory with SPA (p. 190). Some of the parameters are loaded into the volatile memory of the E-873 from the connected positioner's ID chip (p. 13) when the E-873 is switched on or rebooted. When you use the PC software from PI, further information is loaded as parameter values from a positioner database (p. 12) into the volatile memory of the E-873.</p> <p>WPA can also save parameter-independent settings set with the following commands: HDT (p. 151) assigns an HID axis to a lookup table HIA (p. 153), configures HID control HIT (p. 160) fills lookup tables with values</p> <p>The used password determines what is saved with WPA:</p>						
Passwords	<p>Valid passwords for writing to the nonvolatile memory:</p> <table> <tr> <td>100</td><td>Saves the currently valid values of all parameters and the currently valid settings for HDT, HIA and HIT</td></tr> <tr> <td>101</td><td>Saves the currently valid values of all parameters</td></tr> <tr> <td>HID</td><td>Saves the currently valid settings for HDT, HIA and HIT</td></tr> </table>	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
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-873; i. e., <ItemID> and <PamID> are ignored.						

8.7 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 permitted 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_PROGRESS	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 permitted when analog target is active
73	PI_CNTR_WAVE_GENERATOR_ACTIVE	Motion commands are not permitted 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 overrun 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 permitted 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 permitted 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 permitted 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 activated.
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
97	PI_CNTR_INVALID_VALUE_LENGTH	Invalid length of value (too much characters)
98	PI_CNTR_AUTOZERO_FAILED	Autozero procedure was not successful
99	PI_CNTR_SENSOR_VOLTAGE_OFF	Sensor voltage is off
100	PI_LABVIEW_ERROR	PI driver for use with NI LabVIEW 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 permitted 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
220	PI_CNTR_INTERPOLATION_FIFO_UNDERRUN	FIFO buffer underrun during interpolation
221	PI_CNTR_INTERPOLATION_FIFO_OVERFLOW	FIFO buffer underrun during interpolation
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.
507	PI_CNTR_LOST_SPI_SLAVE_CONNECTION	Digital connection to external SPI slave device is lost
508	PI_CNTR_MOVE_ATTEMPT_NOT_PERMITTED	Move attempt not permitted due to customer or limit settings
509	PI_CNTR_TRIGGER_EMERGENCY_STOP	Emergency stop caused by trigger input
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

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
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)
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
545	PI_CNTR_SOFTLIMITS_INVALID	Soft limits invalid due to changes in coordinate system
546	PI_CNTR_CS_WRITE_PROTECTED	Coordinate system is write protected
547	PI_CNTR_CS_CONTENT_FROM_CONFIG_FILE	Coordinate system cannot be changed because its content is loaded from a configuration file
548	PI_CNTR_CS_CANNOT_BE_LINKED	Coordinate system may not be linked
549	PI_CNTR_KSB_CS_ROTATION_ONLY	A KSB-type coordinate system can only be rotated by multiples of 90 degrees
551	PI_CNTR_CS_DATA_CANNOT_BE_QUERIED	This query is not supported for this coordinate system type
552	PI_CNTR_CS_COMBINATION_DOES_NOT_EXIST	This combination of work-and-tool coordinate systems does not exist
553	PI_CNTR_CS_COMBINATION_INVALID	The combination must consist of one work and one tool coordinate system
554	PI_CNTR_CS_TYPE_DOES_NOT_EXIST	This coordinate system type does not exist
555	PI_CNTR_UNKNOWN_ERROR	BasMac: unknown controller error

556	PI_CNTR_CS_TYPE_NOT_ACTIVATED	No coordinate system of this type is activated
557	PI_CNTR_CS_NAME_INVALID	Name of coordinate system is invalid
558	PI_CNTR_CS_GENERAL_FILE_MISSING	File with stored CS systems is missing or erroneous
559	PI_CNTR_CS_LEVELING_FILE_MISSING	File with leveling CS is missing or erroneous
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
607	PI_CNTR_NO_INTEGER	Value is not integer
608	PI_CNTR_FAST_ALIGNMENT_PROCESS_IS_NOT_RUNNING	Fast alignment process cannot be paused because it is not running
609	PI_CNTR_FAST_ALIGNMENT_PROCESS_IS_NOT_PAUSED	Fast alignment process cannot be restarted/resumed because it is not paused
650	PI_CNTR_UNABLE_TO_SET_PARAM_WITH_SPA	Parameter could not be set with SPA - SEP needed?
651	PI_CNTR_PHASE_FINDING_ERROR	Phase finding error
652	PI_CNTR_SENSOR_SETUP_ERROR	Sensor setup error
653	PI_CNTR_SENSOR_COMM_ERROR	Sensor communication error
654	PI_CNTR_MOTOR_AMPLIFIER_ERROR	Motor amplifier error
655	PI_CNTR_OVER_CURR_PROTEC_TRIGGERED_BY_I2T	Overcurrent protection triggered by I2T-module
656	PI_CNTR_OVER_CURR_PROTEC_TRIGGERED_BY_AMP_MODULE	Overcurrent protection triggered by amplifier module
657	PI_CNTR_SAFETY_STOP_TRIGGERED	Safety stop triggered
658	PI_SENSOR_OFF	Sensor off?

700	PI_CNTR_COMMAND_NOT_ALLOWED_IN_EXTERNAL_MODE	Command not permitted in external mode
710	PI_CNTR_EXTERNAL_MODE_ERROR	External mode communication error
715	PI_CNTR_INVALID_MODE_OF_OPERATION	Invalid mode of operation
716	PI_CNTR_FIRMWARE_STOPPED_BY_CMD	Firmware stopped by command (#27)
717	PI_CNTR_EXTERNAL_MODE_DRIVER_MISSING	External mode driver missing
718	PI_CNTR_CONFIGURATION_FAILURE_EXTERNAL_MODE	Missing or incorrect configuration of external mode
719	PI_CNTR_EXTERNAL_MODE_CYCLETIME_INVALID	External mode cycle time invalid
720	PI_CNTR_BRAKE_ACTIVATED	Brake is activated
731	PI_CNTR_SURFACEDETECTION_RUNNING	Command not permitted while surface detection is running
732	PI_CNTR_SURFACEDETECTION_FAILED	Last surface detection failed
733	PI_CNTR_FIELDBUS_IS_ACTIVE	Fieldbus is active and is blocking GCS control commands
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 response 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
1025	PI_CNTR_MAX_MOTOR_OUTPUT_REACHED	Maximum motor output reached
1063	PI_CNTR_EXT_PROFILE_UNALLOWED_COMMAND	User profile mode: command is not permitted, 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
4008	PI_CNTR_FW_CRC_PAR_ERROR	FW Parameter CRC wrong
4009	PI_CNTR_FW_CRC_FW_ERROR	FW CRC wrong
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 performed 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
5300	PI_CNTR_FREQU_ANALYSIS_FAILED	Frequency analysis failed
5301	PI_CNTR_FREQU_ANALYSIS_RUNNING	Another frequency analysis is running

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!)
-62	COM_SOCKET_HOST_NOT_FOUND	Host not found
-63	COM_DEVICE_CONNECTED	Device already connected

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 www.pi.ws .
-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.
-1083	PI_WRONG_DATA_TYPE	A function argument has an unexpected data type.
-1084	PI_DIFFERENT_ARRAY_SIZES	Length of data arrays is different.
-1085	PI_PARAM_NOT_FOUND_IN_PARAM_DAT_FILE	Parameter value not found in parameter DAT file.
-1086	PI_MACRO_RECORDING_NOT_ALLOWED_IN_THIS_MODE	Macro recording is not allowed in this mode of operation.
-1087	PI_USER_CANCELLED_COMMAND	Command cancelled by user input.
-1088	PI_TOO_FEW_GCS_DATA	Controller sent too few GCS data sets
-1089	PI_TOO_MANY_GCS_DATA	Controller sent too many GCS data sets

-1090	PI_GCS_DATA_READ_ERROR	Communication error while reading GCS data
-1091	PI_WRONG_NUMBER_OF_INPUT_ARGUMENTS	Wrong number of input arguments.
-1092	PI_FAILED_TO_CHANGE_CCL_LEVEL	Change of command level has failed.
-1093	PI_FAILED_TO_SWITCH_OFF_SERVO	Switching off the servo mode has failed.
-1094	PI_FAILED_TO_SET_SINGLE_PARAMETER_WHILE_PERFORMING_CST	A parameter could not be set while performing CST: CST was not performed (parameters remain unchanged).
-1095	PI_ERROR_CONTROLLER_REBOOT	Connection could not be reestablished after reboot.
-1096	PI_ERROR_AT_QHPA	Sending HPA? or receiving the response has failed.
-1097	PI_QHPA_NONCOMPLIANT_WITH_GCS	HPA? response does not comply with GCS2 syntax.
-1098	PI_FAILED_TO_READ_QSPA	Response to SPA? could not be received. Response to SPA? could not be received.
-1099	PI_PAM_FILE_WRONG_VERSION	Version of PAM file cannot be handled (too old or too new)
-1100	PI_PAM_FILE_INVALID_FORMAT	PAM file does not contain required data in PAM-file format
-1101	PI_INCOMPLETE_INFORMATION	Information does not contain all required data
-1102	PI_NO_VALUE_AVAILABLE	No value for parameter available
-1103	PI_NO_PAM_FILE_OPEN	No PAM file is open
-1104	PI_INVALID_VALUE	Invalid value
-1105	PI_UNKNOWN_PARAMETER	Unknown parameter
-1106	PI_RESPONSE_TO_QSEP_FAILED	Response to SEP? could not be received.
-1107	PI_RESPONSE_TO_QSPA_FAILED	Response to SPA? could not be received.
-1108	PI_ERROR_IN_CST_VALIDATION	Error while performing CST: One or more parameters were not set correctly.
-1109	PI_ERROR_PAM_FILE_HAS_DUPLICATE_ENTRY_WITH_DIFFERENT_VALUES	PAM file has duplicate entry with different values.

-1110	PI_ERROR_FILE_NO_SIGNATURE	File has no signature
-1111	PI_ERROR_FILE_INVALID_SIGNATURE	File has invalid signature
-10000	PI_PARAMETER_DB_INVALID_STAGE_TYPE_FORMAT	PI stage database: String containing stage type and description has invalid format.
-10001	PI_PARAMETER_DB_SYSTEM_NOT_AVAILABLE	PI stage database: Database does not contain the selected stage type for the connected controller.
-10002	PI_PARAMETER_DB_FAILED_TO_ESTABLISH_CONNECTION	PI stage database: Establishing the connection has failed.
-10003	PI_PARAMETER_DB_COMMUNICATION_ERROR	PI stage database: Communication was interrupted (e.g. because database was deleted).
-10004	PI_PARAMETER_DB_ERROR_WHILE_QUERYING_PARAMETERS	PI stage database: Querying data failed.
-10005	PI_PARAMETER_DB_SYSTEM_ALREADY_EXISTS	PI stage database: System already exists. Rename stage and try again.
-10006	PI_PARAMETER_DB_QHPA_CONTAINS_UNKNOWN_PARAMETER_IDS	PI stage database: Response to HPA? contains unknown parameter IDs.
-10007	PI_PARAMETER_DB_AND_QHPA_ARE_INCONSISTENT	PI stage database: Inconsistency between database and response to HPA?.
-10008	PI_PARAMETER_DB_SYSTEM_COULD_NOT_BE_ADDED	PI stage database: Stage has not been added.
-10009	PI_PARAMETER_DB_SYSTEM_COULD_NOT_BE_REMOVED	PI stage database: Stage has not been removed.
-10010	PI_PARAMETER_DB_CONTROLLER_DB_PARAMETERS_MISMATCH	Controller does not support all stage parameters stored in PI stage database. No parameters were set.
-10011	PI_PARAMETER_DB_DATABASE_IS_OUTDATED	The version of PISTAGES3.DB stage database is out of date. Please update via PIUpdateFinder. No parameters were set.

-10012	PI_PARAMETER_DB_AND_HPA_MISMATCH_STRICT	Mismatch between number of parameters present in stage database and available in controller interface. No parameters were set.
-10013	PI_PARAMETER_DB_AND_HPA_MISMATCH_LOOSE	Mismatch between number of parameters present in stage database and available in controller interface. Some parameters were ignored.
-10014	PI_PARAMETER_DB_FAILED_TO_SET_PARAMETERS_CORRECTLY	One or more parameters could not be set correctly on the controller.
-10015	PI_PARAMETER_DB_MISSING_PARAMETER_DEFINITIONS_IN_DATABASE	One or more parameter definitions are not present in stage database. Please update PISTAGES3.DB via PIUpdateFinder. Missing parameters were ignored.

9 Adapting Settings

In this Chapter

Settings of the E-873.....	229
Changing Parameter Values in the E-873	229
Creating or Changing a Positioner Type.....	234
Parameter Overview	239

9.1 Settings of the E-873

The properties of the E-873 and the connected positioner are stored in the E-873 as parameter values (e.g., settings for the servo algorithm (p. 23)).

The parameters can be divided into the following categories:

- Protected parameters whose default settings cannot be changed
- Parameters that can be set by the user to adapt to the application

The write permission for the parameters is determined by command levels.

Every parameter is in the volatile as well as in the nonvolatile memory of the E-873. The values in the nonvolatile memory are loaded to the volatile memory as default values when switching on or rebooting the E-873. The values in the volatile memory determine the current behavior of the system.

The designation "Active Values" is used for the parameter values in the volatile memory and "Startup Values" is used for the parameter values in the nonvolatile memory in the PC software from PI.

9.2 Changing Parameter Values in the E-873

NOTICE



Unsuitable parameter settings!

The values in the nonvolatile memory are loaded to the volatile memory as default values when switching on or rebooting the E-873 and take effect immediately. Unsuitable parameter settings can cause damage to the connected mechanics.

- Change parameter values only after careful consideration.
- Save the current parameter values to the PC (p. 231) before you make changes in the nonvolatile memory.

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. 231) before you make changes in the nonvolatile memory.
- Contact our customer service department (p. 257), if the E-873 exhibits unexpected behavior.

INFORMATION

If the connected positioner has an ID chip (p. 13), the data is loaded from the ID chip into the volatile memory of the E-873 after switching on or rebooting the E-873.

The ID chip only contains some of the information that is required to operate the positioner with the E-873. When you use the PC software from PI, further information is loaded as parameter values from a positioner database (p. 12) into the volatile memory of the E-873. Parameters that are loaded from the ID chip or from a positioner database are marked in color in the parameter overview (p. 239).

9.2.1 General Commands for Parameters

The following general commands are available for parameters:

Command	Function
CCL	Change to a higher command level, e.g., to obtain write permission for particular parameters.
CCL?	Get active command level.
HPA?	Responds with a help string that contains all available parameters with short descriptions.
RPA	Copy a parameter value from the nonvolatile to the volatile memory.
SEP	Change parameters in the nonvolatile memory.
SEP?	Get parameter values from the nonvolatile memory.
SPA	Change parameters in the volatile memory.
SPA?	Get parameter values from the volatile memory.
WPA	Copy a current parameter value from the volatile to the nonvolatile memory. Here it is used as a default value.

You can find details in the command descriptions (p. 122).

9.2.2 Saving Parameter Values in a Text File

INFORMATION

The E-873 is configured via parameters, e.g., to adapt the mechanics connected. Changing parameter values can cause undesirable results.

- Create a backup copy on the PC before changing the parameter settings of the E-873. You can then restore the original settings at any time.
- Create an additional backup copy with a new file name each time after optimizing the parameter values or adapting the E-873 to specific mechanics.

INFORMATION

Parameter values saved in a text file on the PC can be loaded back to the E-873 in PIMikroMove or PITerminal. The **Send file...** button is available for this purpose in the send command window. Before loading into the E-873, the individual lines of the text files must be converted into command lines that contain the corresponding SPA or SEP commands.

Requirements

- ✓ You have established communication with PIMikroMove or PITerminal between the E-873 and the PC (p. 54).

Saving parameter values in a text file

1. If you use PIMikroMove, open the window for sending commands:
 - Select the **Tools > Command entry** menu item in the main window 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-873: Send the **SPA?** command.
 - If you want to save the parameter values from the nonvolatile memory of the E-873: Send the **SEP?** command.
3. Click on the **Save...** button.

The **Save content of terminal as textfile** window opens.
4. Save the queried parameter values in a text file to your PC in the **Save content of terminal as textfile** window.

9.2.3 Changing Parameter Values: General Procedure

For working with parameters, you can use the general commands (p. 230) and the commands for quick access.

For simpler access to parameters, PIMikroMove is used in the following, so you do not have to deal with the corresponding commands.

NOTICE



Unsuitable parameter settings!

The values in the nonvolatile memory are loaded to the volatile memory as default values when switching on or rebooting the E-873 and take effect immediately. Unsuitable parameter settings can cause damage to the connected mechanics.

- Change parameter values only after careful consideration.
- Save the current parameter values to the PC (p. 231) before you make changes in the nonvolatile memory.

INFORMATION

The following procedure is generally recommended for changing parameter values:

1. Change the parameter values in the volatile memory.
2. Check whether the E-873 works correctly with the changed parameter values.

If so:

- Write the changed parameter values into the nonvolatile memory.

If not:

- Change and check the parameter values in the volatile memory again.

INFORMATION

The write access for the parameters of the E-873 is defined by command levels. After the controller is switched on or rebooted, the active command level is always 0. On command levels > 1, write access is only available to PI service personnel.

- Contact the customer service department if there seem to be problems with parameters of command level 2 or higher (p. 257).

Requirements

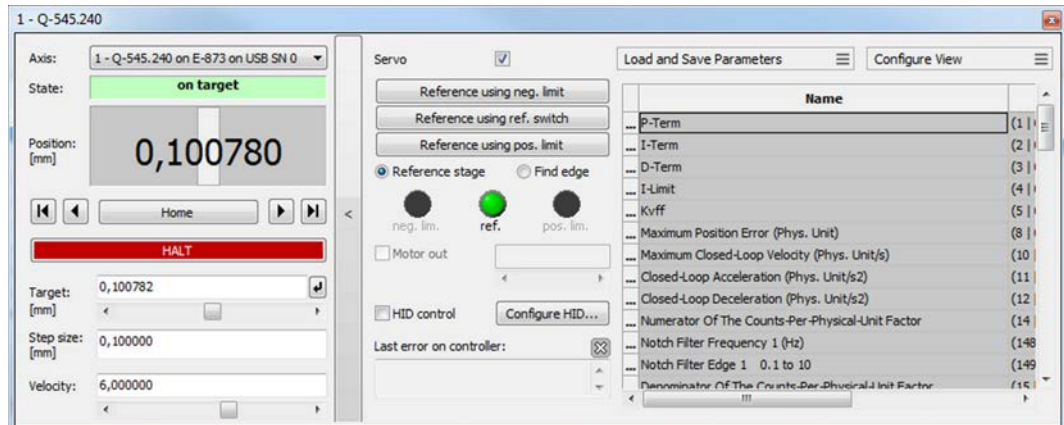
- ✓ If you want to change parameter values in the nonvolatile memory of the E-873: You have saved the parameter values of the E-873 in a text file on the PC (p. 231).
- ✓ You have established communication between the E-873 and the PC with PIMikroMove (p. 54).

Changing parameter values: General procedure

1. Display the parameter list in PIMikroMove.

If you want to change the axis-related parameters of the E-873:

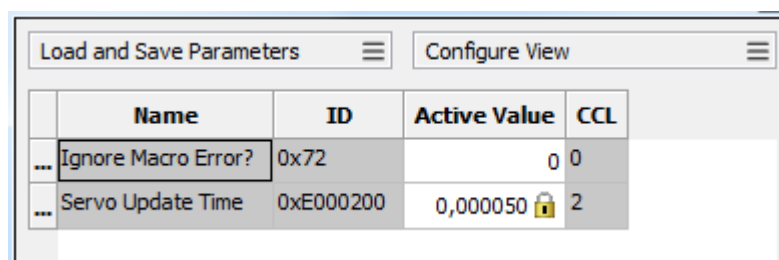
- a) Open the expanded single axis window for the connected positioner in the main window of PIMikroMove by clicking the right mouse button on the corresponding line of the **Axes** tab and selecting **Show Expanded Single Axis Window** in the context menu.



- b) If the parameter to be modified is not included in the list on the right-hand side of the window, click **Configure View > Select parameters...** and add it to the list. You can also display certain groups of parameters or all axis-related parameters.

If you want to change the system-related parameters of the E-873:

- Open the window for the system-related parameters of the E-873 in the main window of PIMikroMove by selecting **E-873 > Show system parameters** in the menu.



2. Change the desired parameter values in the volatile or nonvolatile memory of the E-873 in the corresponding parameter list.

If you want to change parameter values in the volatile memory, you have the following options:

- Type the new parameter value into the corresponding input field in the **Active Value** column of the list. Press the **Enter** key on the PC keyboard or click with the mouse outside the input field to transfer the parameter value to the volatile memory of the E-873.

- Click **Load and Save Parameters -> Load all startup parameters of the axis / system from controller** in order to load the values of all axis-related / system-related parameters from the nonvolatile memory of the E-873.
- Click **Load and Save Parameters > Load parameters from stage database...** in the extended single-axis window to load a selected parameter set for the axis from the positioner database. You can use **Load and Save Parameters > Reload parameters from stage database...** to reload the currently loaded parameter set.

If you want to change parameter values in the nonvolatile memory, you have the following options:

- Type the new parameter value into the appropriate input field in the **Startup Value** column in the list. Press the **Enter** key on the PC keyboard or click with the mouse outside the input field to transfer the parameter value to the nonvolatile memory of the E-873.
- Click **Load and Save Parameters -> Save all currently active axis / system parameters as startup parameters to controller** to write the values of all axis-related / system-related parameters from the volatile to the nonvolatile memory of the E-873. You can skip parameters that do not have write access on the current command level.

If a parameter value in the volatile memory (**Active Value** column) is different from the parameter value in the nonvolatile memory (**Startup Value** column), the line in the list is highlighted in color.

9.3 Creating or Changing a Positioner Type

You can select a suitable parameter set for your positioner from a positioner database in the PC software from PI. The software transfers the values of the selected parameter set to the volatile or nonvolatile memory of the controller. For further information, see "Positioner Databases" (p. 12).

You can create and edit new parameter records in the PI_UserStages2.dat positioner database. This can be required in the following cases, for example:

- You would like to operate a positioner with different servo control parameter settings than those in the default parameter set.
- You would like to adapt the soft limits of the positioner to your application.
- You have a custom positioner.

INFORMATION

You can create a new positioner type easily by modifying an existing positioner type in PIMikroMove and saving it under a new name.

INFORMATION

If a positioner type with the same name is present in the standard positioner 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 positioner type is selected in the PC software. The parameter settings from PI_UserStages2.dat are not used in this case.

- When saving positioner types, only assign names that are **not** already used in the PIStages2.dat or PIMicosStages2.dat positioner database.

In the following, PIMikroMove is used for creating a new positioner type and for modifying an existing positioner type.

Requirements

- ✓ You have installed the latest version of the pistages2.dat and pimicosstages2.dat positioner databases onto your PC (p. 48).
- ✓ If PI has provided you with a custom database for your positioner, then you have installed this database on your PC (p. 50).
- ✓ You have established communication with PIMikroMove between the E-873 and the PC (p. 54).

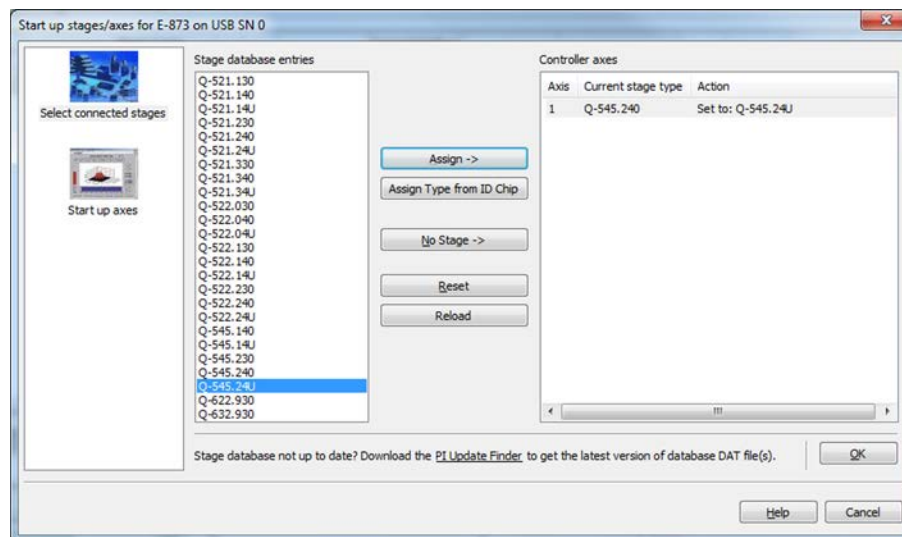
Create a positioner type in the positioner database

1. In the main window of PIMikroMove, select the **E-873 > Select connected stages...** menu item.

The **Start up stages/axes for E-873** window opens, the step **Select connected stages** is active.

2. Select an appropriate type of positioner during the **Select connected stages** step:
 - Click on **Assign Type from ID Chip**.
 - or
 - a) Mark the positioner type in the **Stage database entries** list.

b) Click **Assign**.

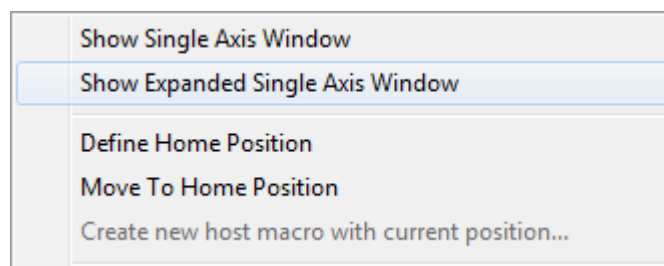


c) Confirm the selection with **OK**.

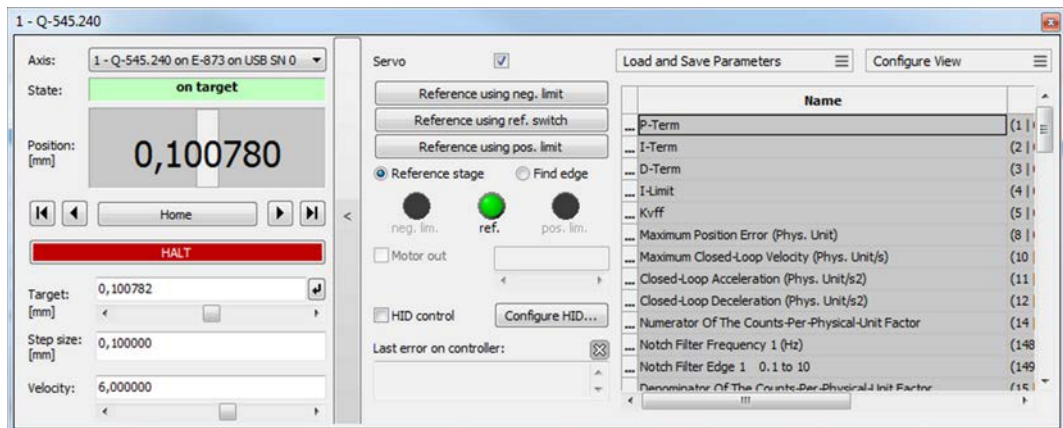
3. In the **Save all changes permanently** dialog, click **Keep the changes temporarily** to load the parameter settings into the volatile memory of the E-873.

The **Start up stages/axes** window changes to the step **Start up axes**.

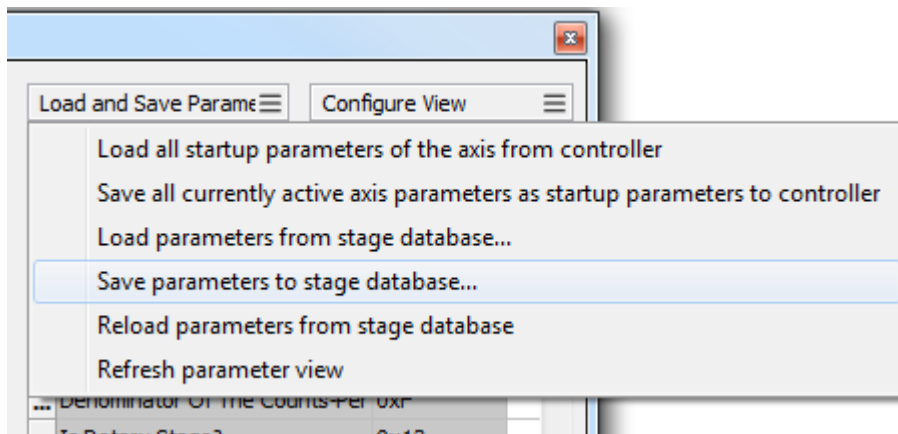
4. In the step **Start up axes** click on **Close** to close the **Start up stages/axes** window.
5. Open the expanded single axis window for the selected positioner in the main window of PIMikroMove by clicking the right mouse button on the corresponding line of the **Axes** tab and selecting **Show Expanded Single Axis Window** in the context menu.



6. Enter new values for the parameters to be changed:



- If the parameter to be modified is not included in the list on the right-hand side of the window, click on **Configure view > Select parameters...** and add it to the list. You can also display certain groups of parameters or all axes-related parameters.
 - Type the new parameter value into the corresponding input field in the **Active Value** column of the list.
 - Press the **Enter** key on the PC keyboard or click outside the input field with the mouse to transfer the parameter value to the volatile memory of the controller. Note: If a parameter value in the volatile memory (**Active Value** column) is different to the parameter value in the nonvolatile memory (**Startup Value** column), the line in the list is highlighted in color.
7. Click on **Load and Save Parameters -> Save parameters to stage database...**



The **Save Parameters as User Stage Type** dialog opens.

- Save the changed parameter values as new positioner type in the **Save Parameters as User Stage Type** dialog:
 - Leave the entry in the **Parameters of axis** field unchanged.
 - Enter the name for the new positioner type into the **Save as** field.
 - Click **OK**.

The new positioner type was saved to the PI_UserStages2.dat positioner database. The display of the connected positioner type was updated in the single axis window and in the main window of PIMikroMove. The new positioner type is also available immediately for selection in the **Select connected stages** step.

Changing a positioner type in the positioner database

1. Select the **E-873 > Select connected stages...** menu item in the main window of PIMikroMove.

The **Start up stages/axes for E-873** window opens, the **Select connected stages** step is active.

2. Select one of the positioners you created as described above (p. 235): Proceed with the selection as described in step 2 of the **Creating a positioner type in the positioner database** instruction.
3. Proceed with steps 3 to 7 in **Creating a positioner type in the positioner database**.
4. Save the modified parameter values of the positioner type in the **Save Parameters as User Stage Type** dialog:
 - a) Leave the entry in the **Parameters of axis** field unchanged.
 - b) Leave the entry in the **Save as** field unchanged.
 - c) Click **OK**.
 - d) Click **Change settings** in the **Stage type already defined** dialog. The **Save Parameters as User Stage Type** dialog closes automatically after a short time.

The parameter values of the positioner type have been updated in the PI_UserStages2.dat positioner database and in the main window of PIMikroMove.

9.4 Parameter Overview

INFORMATION

The write access for the parameters of the E-873 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-873 ignores the active command level in the following cases:

- The E-873 reads parameter values from the ID chip of the positioner.
 - The positioner 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. 257).

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 highlight in the parameter table:

Dark gray:	The value of the parameter is loaded from the ID chip of the positioner (p. 13).
Light gray:	The value of the parameter can be loaded from a positioner database (p. 12).
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. 232).

Designations in the header of the following table:

- ID = Parameter ID, hexadecimal format
- Type = Data type:
 - INT = integer value, including Boolean values

- FLOAT = floating point number
- CHAR = String format
- CL = Command Level for write access
- Element = Element type to which the parameter refers, for more information, see "Commandable Items" (p. 16)
- Parameter name = Name of the parameter
- Description = Explanation of the parameter

ID	Type	CL	Element	Parameter name	Description
0x1	INT	0	Axis	P term	Proportional constant of the PID servo algorithm For details, see "Servo Algorithm and Other Control Value Corrections" (p. 23).
0x2	INT	0	Axis	I term	Integration constant of the PID servo algorithm For details, see "Servo Algorithm and Other Control Value Corrections" (p. 23).
0x3	INT	0	Axis	D term	Differential constant of the PID servo algorithm For details, see "Servo Algorithm and Other Control Value Corrections" (p. 23).
0x4	INT	0	Axis	I limit	Limiting the integration constant of the PID servo algorithm For details, see "Servo Algorithm and Other Control Value Corrections" (p. 23).
0x5	INT	0	Axis	Kvff	Available for compatibility reasons only.
0x8	FLOAT	0	Axis	Maximum Position Error (Phys. Unit)	Maximum position error The E-873 does not evaluate this parameter.
0x9	INT	0	Axis	Maximum Motor Output	Maximum permissible absolute measure of the control value (dimensionless)
0xE	INT	0	Axis	Numerator Of The Counts-Per-Physical-Unit Factor	Numerator and denominator of the factor for counts per physical length

ID	Type	CL	Element	Parameter name	Description
0xF	INT	0	Axis	Denominator Of The Counts-Per-Physical-Unit Factor	unit For details, see "Physical Units" (p. 21).
0x13	INT	0	Axis	Is Rotary Stage?	Is this a rotation stage? 0 = Not a rotation stage 1 = Rotation stage No evaluation by the E-873, but only by the PC software: PIMikroMove determines which motion is permissible on the basis of this value.
0x14	INT	0	Axis	Has Reference?	Does the positioner have a reference switch? For details, see "Reference Switch Detection" (p. 27).
0x15	FLOAT	0	Axis	Maximum Travel In Positive Direction (Phys. Unit)	Soft limit in positive direction See examples in "Travel Range and Soft Limits" (p. 31).
0x16	FLOAT	0	Axis	Value At Reference Position (Phys. Unit)	Position value at the reference switch See examples in "Travel Range and Soft Limits" (p. 31).
0x17	FLOAT	0	Axis	Distance From Negative Limit To Reference Position (Phys. Unit)	Gap between the reference switch and the negative end of the travel range See examples in "Travel Range and Soft Limits" (p. 31).
0x18	INT	0	Axis	Limit Mode	Signal logic of the limit switches The E-873 can only receive limit switch signals via digital input lines. For details, see "Limit Switch Detection" (p. 28).
0x1B	INT	0	Axis	Profile mode	Type of dynamics profile 5 = Without dynamics profile
0x2F	FLOAT	0	Axis	Distance From Reference Position To Positive Limit (Phys. Unit)	Gap between the reference switch and the positive end of the travel range See examples in "Travel Range and Soft Limits" (p. 31).
0x30	FLOAT	0	Axis	Maximum Travel In Negative Direction (Phys. Unit)	Soft limit in a negative direction See examples in "Travel Range and Soft Limits" (p. 31).

ID	Type	CL	Element	Parameter name	Description
0x31	INT	0	Axis	Invert Reference?	Should the reference signal be inverted? For details, see "Reference Switch Detection" (p. 27).
0x32	INT	0	Axis	Has No Limit Switches?	Does the positioner have limit switches? The E-873 does not receive limit switch signals at the connections for positioners. It can only receive limit switch signals via the digital input lines. See "Limit Switch Detection". (p. 28)
0x33	INT	0	Axis	Motor Offset Positive	Available for compatibility reasons only.
0x34	INT	0	Axis	Motor Offset Negative	Available for compatibility reasons only.
0x36	INT	0	Axis	Settling Window (encoder counts)	Settling window around the target position For details, see "On-Target State" (p. 26).
0x3C	CHAR	0	Axis	Stage Name	Positioner name Maximum of 20 characters; default value: NOSTAGE The value NOSTAGE "deactivates" the axis. A deactivated axis is not accessible for axis-related commands (e.g., motion commands or position queries).
0x3F	FLOAT	0	Axis	Settling Time (s)	Delay time for setting the on-target state. For details, see "On-Target State" (p. 26).
0x47	INT	0	Axis	Reference Travel Direction	Default direction for the reference move For details, see "Referencing" (p. 34).
0x48	INT	0	Axis	Motor Drive Offset	Available for compatibility reasons only.
0x5A	INT	0	Axis	Numerator Of The Servo Loop Input Factor	Numerator and denominator of the servo-loop input factor For details, see "Servo Algorithm and Other Control Value Corrections" (p. 23).
0x5B	INT	0	Axis	Denominator Of The Servo Loop Input Factor	

ID	Type	CL	Element	Parameter name	Description
0x5C	INT	0	Axis	Source Of Reference Signal	Reference signal source for the FRF or FED commands For details, see "Commands and Parameters for Digital Inputs" (p. 88) and "Using Digital Input Signals as Switch Signals" (p. 90).
0x5D	INT	0	Axis	Source Of Negative Limit Signal	Reference signal source for the FNL or FED commands For details, see "Commands and Parameters for Digital Inputs" (p. 88) and "Using Digital Input Signals as Switch Signals" (p. 90).
0x5E	INT	0	Axis	Source Of Positive Limit Signal	Reference signal source for the FPL or FED commands For details, see "Commands and Parameters for Digital Inputs" (p. 88) and "Using Digital Input Signals as Switch Signals" (p. 90).
0x5F	INT	0	Axis	Invert Digital Input Used For Negative Limit	Inverts the polarity of the digital inputs that are used as the source of the negative limit switch signal. For details, see "Commands and Parameters for Digital Inputs" (p. 88) and "Using Digital Input Signals as Switch Signals" (p. 90).
0x60	INT	0	Axis	Invert Digital Input Used For Positive Limit	Inverts the polarity of the digital inputs that are used as the source of the positive limit switch signal. For details, see "Commands and Parameters for Digital Inputs" (p. 88) and "Using Digital Input Signals as Switch Signals" (p. 90).
0x61	INT	0	Axis	Invert Direction Of Motion For Joystick-Controlled Axis?	Should the direction of motion for HID-controlled axes be inverted? 0 = Direction of motion not inverted 1 = direction of motion inverted For details, see "Commands and Parameters for HID Control" (p. 92).
0x63	FLOAT	0	Axis	Distance Between Limit And Hard Stop (Phys. Unit)	Distance between the built-in limit switch and the hard stop For details, see "Referencing" (p. 34).

ID	Type	CL	Element	Parameter name	Description
0x70	INT	0	Axis	Reference Signal Mode	Reference signal type For details, see "Reference Switch Detection" (p. 27).
0x71	INT	0	Axis	D Term Delay (No. Of Servo Cycles)	D term delay For details, see "Servo Algorithm and Other Control Value Corrections" (p. 23).
0x72	INT	0	System	Ignore Macro Error?	Ignore macro error? For details, see "Commands and Parameters for Macros" (p. 100).
0x77	INT	0	Axis	Use Limit Switches Only For Reference Moves?	Should the limit switches only be used for reference moves? For details, see "Limit Switch Detection" (p. 28).
0x78	FLOAT	0	Axis	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 For details, see "Referencing" (p. 34).
0x79	FLOAT	0	Axis	Distance For Reference Search (Phys. Unit)	Maximum distance for the reference move to the index pulse For details, see "Referencing" (p. 34).
0x7A	INT	0	Axis	Use Hard Stops For Referencing?	Should the hard stops be used for reference moves? For details, see "Referencing" (p. 34).
0x94	FLOAT	0	Axis	Notch Filter Frequency 1 (Hz)	Frequency of the first notch filter For details, see "Servo Algorithm and Other Control Value Corrections" (p. 23).
0x95	FLOAT	0	Axis	Notch Filter Edge 1	Rise of the edge of the first notch filter For details, see "Servo Algorithm and Other Control Value Corrections" (p. 23).
0x7000000	FLOAT	0	Axis	Range Limit Min	Additional soft limit for the negative direction of motion (physical unit) For details, see "Travel Range and Soft Limits" (p. 29).

ID	Type	CL	Element	Parameter name	Description
0x7000001	FLOAT	0	Axis	Range Limit Max	Additional soft limit for the positive direction of motion (physical unit) For details, see "Travel Range and Soft Limits" (p. 29).
0x07000601	CHAR	0	Axis	Axis Unit	Unit symbol For details, see "Physical Units" (p. 21).
0xD000000	CHAR	2	System	Device S/N	Serial number of the E-873
0x0F000100	CHAR	2	Axis	Stage Type	Positioner type Format for standard positioners: x-xxx Format for customized positioners: x-xxxKxxx
0x0F000200	CHAR	2	Axis	Stage Serial Number	Serial number of the positioner 9-digit number
0x0F000300	CHAR	2	Axis	Stage Assembly Date	Date of manufacture of the positioner Date format: DDMMYY
0x0F000400	INT	2	Axis	Stage HW Version	Version number of the positioner hardware
0x1F000000	FLOAT	1	Axis	PIShift Upper Supply Voltage (V)	Maximum value of the piezo voltage for the Q-Motion® inertia drive For details see "Q-Motion® drive modes" (p. 19).
0x1F000100	FLOAT	1	Axis	PIShift Lower Supply Voltage (V)	Minimum value of the piezo voltage for the Q-Motion® inertia drive For details see "Q-Motion® drive modes" (p. 19).
0x1F000200	FLOAT	1	Axis	PIShift Forward Current (A)	Maximum current consumption of the Q-Motion® inertia drive during forward motion For details see "Q-Motion® drive modes" (p. 19).
0x1F000300	FLOAT	1	Axis	PIShift Backward Current (A)	Maximum current consumption of the Q-Motion® inertia drive during backward motion For details see "Q-Motion® drive modes" (p. 19).

ID	Type	CL	Element	Parameter name	Description
0x1F000400	FLOAT	1	Axis	PIShift Frequency (Hz)	Frequency of the piezo voltage for the step mode of the Q-Motion® inertia drive (= frequency of the modified sawtooth signal) For details see "Q-Motion® drive modes" (p. 19).
0x1F000500	FLOAT	1	Axis	PIShift Charge Cycle	Duty cycle of the current source during the output of one period of the modified sawtooth signal For details see "Q-Motion® drive modes" (p. 19).
0x1F000700	FLOAT	1	Axis	PIShift Step Size (Phys. Unit)	Size of the slow individual steps in closed-loop operation For details, see "Servo Algorithm and Other Control Value Corrections" (p. 23).
0x1F000701	FLOAT	1	Axis	PIShift Delay (ms)	Delay time for closed-loop operation For details, see "Servo Algorithm and Other Control Value Corrections" (p. 23).

10 Maintenance

In this Chapter

Cleaning the E-873	247
Updating Firmware	249

10.1 Cleaning the E-873

NOTICE



Short circuits or flashovers!

The E-873 contains electrostatic-sensitive devices that can be damaged by short-circuiting or flashovers when cleaning fluids penetrate the housing.

- Before cleaning, disconnect the E-873 from the power source by removing the mains plug.
- Prevent cleaning fluid from penetrating the housing.

- When necessary, clean the surfaces of the E-873's housing using a cloth dampened with a mild cleanser or disinfectant.

10.2 Updating Firmware

INFORMATION

The `*IDN?` command reads the version number of the firmware among other things.

Example of a E-873 response:

```
(c)2019 Physik Instrumente (PI) GmbH & Co. KG, E-873.3QTU,
117048994, 01.400
```

- E-873.3QTU: Device name
- 117048994: Serial number of the device.
- 01.400: Firmware version

INFORMATION

If the E-873 is in firmware update mode, the **STA** LED is off. The E-873 does not leave the firmware update mode until it is **rebooted** after a **successful** firmware update. If the firmware update was unsuccessful or aborted, the E-873 remains in the firmware update mode after a reboot.

If the **STA** LED is still off, even though the E-873 has been rebooted after the firmware update:

- Repeat the firmware update.
- If the update of the firmware fails, contact our customer service department (p. 257).

INFORMATION

If new parameters are introduced with the firmware update or the E-873 memory management is changed, an initialization of the E-873 is required after updating the firmware.

Requirements

- ✓ You have connected the E-873 to the PC via the TC/IP or USB interface (p. 51).
- ✓ The **PI Firmware Updater** program is installed on the PC (p. 47).
- ✓ You have copied the new firmware file, which you have received from our customer service department, 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. You have learned from the documentation whether new parameters are introduced with the firmware update or the memory management of the E-873 changes.
- ✓ You have saved the parameter values of the E-873 to a text file on the PC (p. 231)
- ✓ You have saved the E-873 controller macros to files on the PC (p. 109).
- ✓ You have established communication with PIMikroMove or PITerminal between the E-873 and the PC (p. 54).

Updating the firmware of the E-873

1. Activate the firmware update mode in PIMikroMove or PITerminal:
 - a) If the window for sending commands is not already open in PIMikroMove, select the menu item **Tools> Command entry** in the main window or press the **F4** key on the keyboard.
 - b) Send the following commands one by one:

```
ZZZ 100 Flash
```

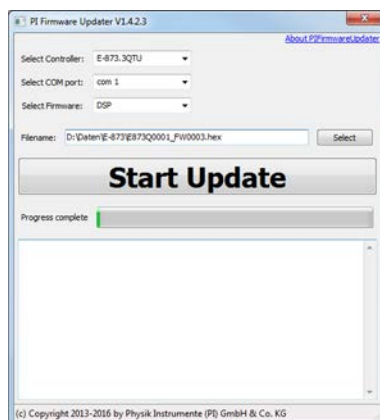
```
rbt
```

The E-873 reboots. If the E-873 is in firmware update mode after rebooting, the **STA** LED is off.

2. Close PIMikroMove® or PITerminal.
3. Start the **PI Firmware Updater** program on the PC.

The **PI Firmware Updater** window opens.

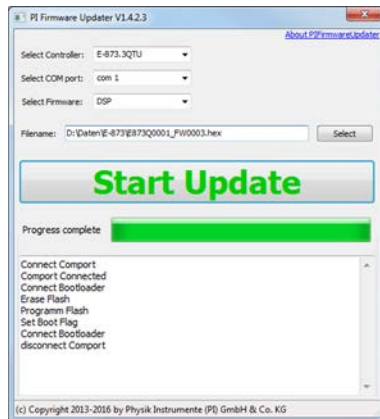
4. Set the following in the selection boxes:
 - In the **Select Controller:** field, select the entry for your controller model: *E-873.3QTU*.
 - Do not change the **Select COM port:** field. This specification is not relevant for connecting the E-873 to the PC via TCP/IP or USB.
 - Do not change the **Select Firmware:** field. The "DSP" (DSP = Digital Signal Processor) is automatically entered by selecting the controller.
5. Select the new firmware file:
 - a) Click the **Select** button.
 - b) In the file selection window, go to the directory in which you have stored the firmware file.
 - c) Double-click on the new firmware file (.hex file extension) to enter the file path in the **Filename** field.



6. Start the firmware update by clicking on the **Start Update** button.

The firmware of the E-873 is updated. The progress of the update is displayed in the message list and by the progress bar.

The update was successful when the `disconnect Comport` message appears as the last entry in the message list.



7. Close the **PI Firmware Updater** program by clicking the cross in the top right corner of the window.
8. Switch the E-873 off and on again.

If the firmware update was successful, the E-873 exits the firmware update mode and the **STA** LED lights up green.

Have new parameters been added by the firmware update, or has the memory management of the E-873 been changed?

- If no: Firmware update is finished.
- If yes: An initialization of the E-873 is required, see below.

Initializing the E-873 after a firmware update

The initialization of the E-873 resets **all** parameters to their factory settings and deletes all controller macros. Consequently, parameter values and controller macros that are not saved are lost during the initialization process.

1. Make sure that the current parameter values and controller macros of the E-873 have been saved on the PC.
2. On the PC, start PITerminal or PIMikroMove, connect to the E-873, and, if necessary, open the window to send commands.
3. Initialize the E-873, by sending the following commands one by one:

```
ZZZ 100 parameter
```

```
ZZZ 100 macros
```

After successful initialization, the controller issues a corresponding message.

4. Adapt the parameter values of the E-873.

For instructions on the general procedure, see "Changing Parameter Values: General Procedure" (p. 232).

- 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.
5. If you have saved controller macros on the PC: Load the controller macros back to the E-873, see "Making Backups and Loading Controller Macros" (p. 109).

11 Troubleshooting

Fault: Positioner does not move	
Possible causes	Remedial measures
Cable not connected correctly	➤ Check the cable connections.
Positioner or cable is defective	➤ If available, replace the defective positioner with another positioner and test the new combination.
Incorrect configuration	➤ Check the parameter settings of the E-873 with the <code>SPA?</code> command (volatile memory) and <code>SEP?</code> command (nonvolatile memory); see "Adapting Settings" (p. 229).
Incorrect command or incorrect syntax	➤ Send the <code>ERR?</code> command and check the error code that is returned.
Incorrect axis commanded	➤ Make sure that the correct axis identifier is used and that the commanded axis belongs to the correct positioner.
HID control is enabled	<p>Motion commands are not permitted when the HID control is enabled for the axis.</p> <p>➤ Disable the HID control with the <code>HIN</code> command.</p>

Fault: Positioner performs unintentional motion	
Possible causes	Remedial measures
HID is not connected but HID control is activated in the E-873	➤ Activate HID control only when there an HID is actually connected to the E-873.
HID axis/axes not calibrated	➤ Calibrate the HID (p. 96) axis/axes.
Startup macro is run	➤ Check whether a macro is specified as the startup macro and cancel the selection of the startup macro if necessary (p. 102).

Fault: Positioner is oscillating or positions inaccurately	
Possible causes	Remedial measures
The load was changed.	➤ Reset the notch filter (p. 66) and the servo control parameters (p. 71) corresponding to the load change.

Fault: Positioner is already oscillating during the reference move	
Possible causes	Remedial measures
Very high load on the positioner	<p>In case of a very high load, proceed with PIMikroMove during the reference move as follows:</p> <ol style="list-style-type: none"> 1. Do not start the reference move in the Start up axes step, but click on Close to close the Start up controller window instead. 2. Open the single axis window for the connected positioner in the main window by selecting the positioner in the View > Single Axis Window menu. 3. Expand the view of the single axis window by clicking on the > button at the right edge of the window. 4. With the Servo check box, make sure that the servo mode is switched on. 5. Start the reference move by clicking on one of the Reference... buttons. 6. If the positioner is oscillating: Stop the reference move immediately in the Reference Axes dialog, close the dialog and switch off the servo mode by unchecking the respective check box in the single axis window. 7. Enter suitable values for the settings of the notch filter, see "Setting the Notch Filter" (p. 66). 8. Restart the reference move. 9. If the positioner continues to oscillate, repeat steps 6 to 8 until the reference move has completed successfully without oscillation.

Fault: There is no communication between the controller and the PC	
Possible causes	Remedial measures
The wrong communication cable is used or it is defective	<ul style="list-style-type: none"> ➤ Use the following cables for TCP/IP connections (if available): <ul style="list-style-type: none"> – TCP/IP direct connection with the PC: crossover network cable – TCP/IP network: Straight-through network cable ➤ Use a null-modem cable for the RS-232 connection (if available). ➤ If necessary, check whether the cable works on a fault-free system.
Communication interface not correctly configured	<p>When using the TCP/IP connection:</p> <ul style="list-style-type: none"> ➤ Connect the controller to the network before you switch it on. Otherwise, you will have to switch the controller off and then on again. ➤ Check the network settings (p. 56). ➤ Make sure that the network is not blocked for unknown devices. ➤ Make sure that several PC software applications cannot access the E-873 at the same time. ➤ Make sure that you have selected the correct E-873 when establishing communication. ➤ If you cannot solve the problems, consult your network administrator if necessary.
Another program is accessing the interface.	<ul style="list-style-type: none"> ➤ Close the other program.
Problems with special software	<ul style="list-style-type: none"> ➤ Check whether the system works with other software, such as a terminal program or a development environment. You can test the communication by starting a terminal program (such as for example, PI Terminal) and entering <code>*IDN?</code> or <code>HLP?</code>. ➤ Make sure that you end the commands with an LF (line feed). <p>A command is only executed when the LF has been received.</p>

Fault: The customer software does not function with the PI drivers	
Possible causes	Remedial measures
Incorrect combination of driver routines/VIs	<ul style="list-style-type: none">➤ Check whether the system functions with a terminal program (e.g., PITerminal). If so: <ul style="list-style-type: none">➤ Read the information in the corresponding software manual and compare the sample code on the product CD with your program code.

If the problem that occurred with your system is not in the list above or cannot be solved as described, contact our customer service department (p. 257).

12 Customer Service

For inquiries and orders, contact your PI sales engineer or send us an email (<mailto:service@pi.de>).

- If you have any questions concerning your system, provide the following information:
 - Product and serial numbers of all products in the system
 - Firmware version of the controller (if applicable)
 - Version of the driver or the software (if applicable)
 - Operating system on the PC (if applicable)
- If possible: Take photographs or make videos of your system that can be sent to our customer service department if requested.

The latest versions of the user manuals are available for download (p. 4) on our website.

13 Technical Data

In this Chapter

Specifications	259
System Requirements	261
Dimensions	262
Pin Assignment	263

13.1 Specifications

13.1.1 Data Table




	E-873.3QTU
Function	Q-Motion® controller for positioning systems with piezo inertia drives, benchtop device with option for control cabinet mounting
Axes	3
Supported functions	Point-to-point motion. Startup macro. Data recorder for recording operating data such as motor voltage, position or position error. Internal safety circuitry: Watchdog timer. ID chip detection.
Motion and control	E-873.3QTU
Controller type	PID controller, parameter changing during operation
Encoder input	Analog encoder inputs sine-cosine, interpolation selectable to 20000. Interpolation electronics preset for differential transmission, 1 V _{pp} and 2.5 V encoder offset signal.
Stall detection	Automatic motor stop
Input reference switch	1 × TTL for integrated reference in the encoder
Electrical properties	E-873.3QTU
Max. output power	30 W per axis
Output voltage	0 to 100 V, drive-dependent selection




Interfaces and operation	E-873.3QTU
Communication interfaces	TCP/IP, USB
Motor / sensor connection	3 × Sub-D 15 (f)
I/O lines	4 digital inputs, 4 digital outputs
Command set	PI General Command Set (GCS)
User software	PIMikroMove
Application programming interfaces	API for C / C++ / C# / VB.NET / MATLAB / Python, drivers for NI LabVIEW
Manual control (optional)	USB joystick

Miscellaneous	E-873.3QTU
Operating voltage	24 V from external power adapter (in the scope of delivery)
Max. current consumption	5 A
Operating temperature range	0 to 50 °C
Mass	1.7 kg
Dimensions	312 mm x 153.4 mm x 59.2 mm (incl. mounting rails)

13.1.2 Maximum Ratings

The E-873 is designed for the following operating data:

Input on:	Maximum operating voltage	Operating frequency	Maximum current consumption
			
Barrel connector socket (m)	24 V	—	5 A

Output on:	Maximum Output Voltage	Maximum output current	Maximum Output Frequency
			
Sub-D 15 (f):	100 V	± 650 mA	25 kHz

13.1.3 Ambient Conditions and Classifications

The following ambient conditions and classifications for the E-873 must be observed:

Area of application	For indoor use only
Maximum altitude	2000 m
Air pressure	1100 hPa to 0.1 hPa (corresponds roughly to 825 torr to 0.075 torr)
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

13.2 System Requirements

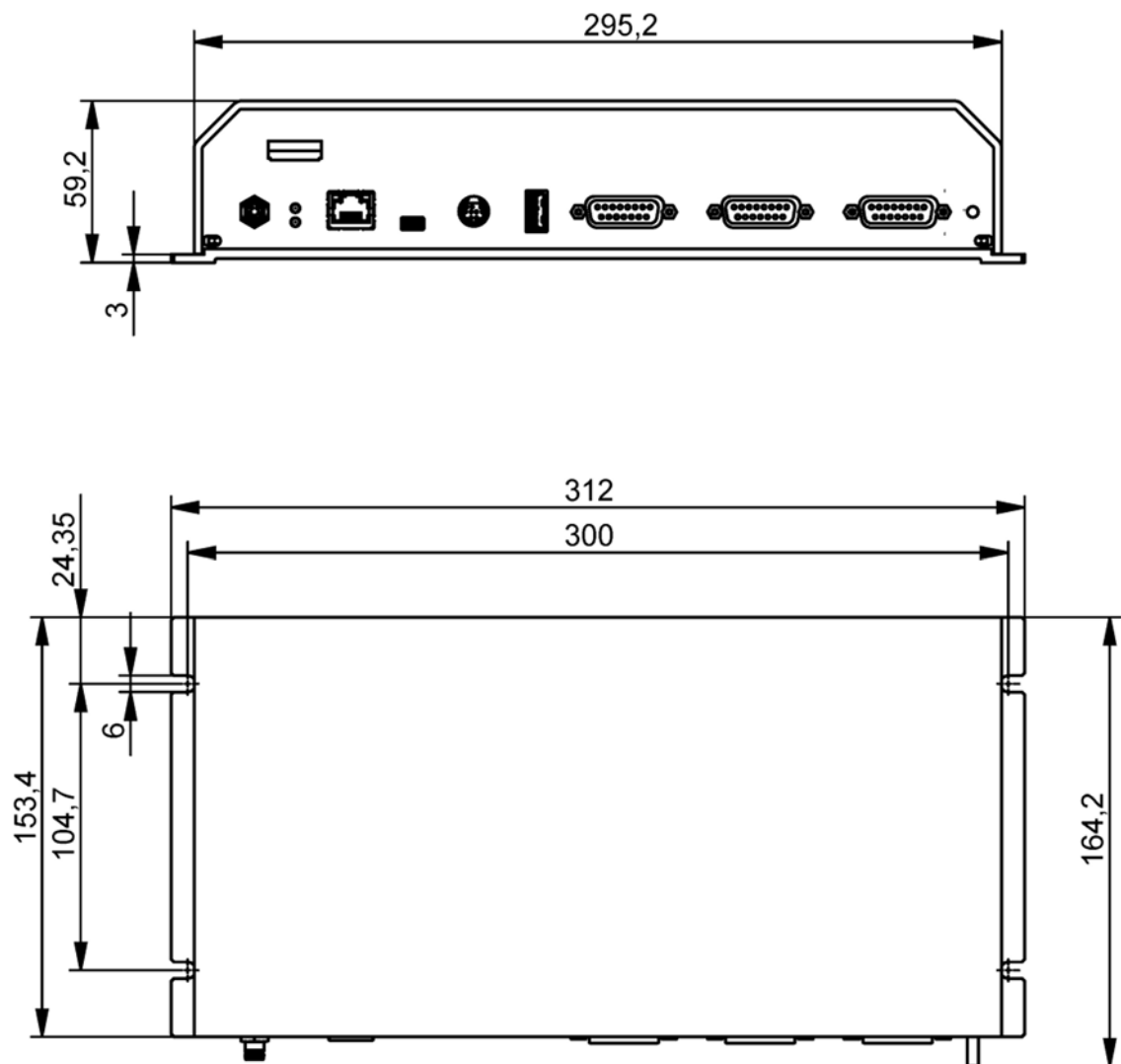
The following system requirements must be met to operate the E-873:

- E-873.3QTU with power supply and adapter for power supply connection (included in the scope of delivery (p. 9))
- Mechanics (positioner) with Q-Motion® inertia drive
- PC with Windows (7, 8, 10; 32 bit, 64 bit) or Linux operating system and at least 30 MB free memory
- For communication of the E-873 with the PC:
 - USB socket on the PC (type A for using the USB cable supplied)
 - USB cable (type A to mini-B in the scope of delivery)
 or
 - RJ45 Ethernet socket on the PC
 - Crossover network cable for direction connection via TCP/IP (in the scope of delivery)
 or
 - Free access point to the network, to which the PC is connected

- Straight-through network cable for connection via a TCP/IP network (in the scope of delivery)
- Product CD with the PC software (in the scope of delivery)

13.3 Dimensions

Dimensions in mm. Note that the decimal points are separated by a comma in the drawings.



13.4 Pin Assignment

13.4.1 Motor and Sensor

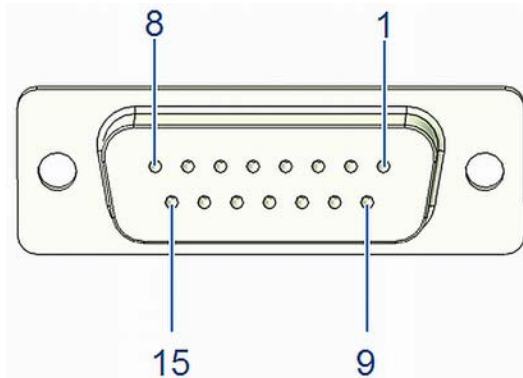


Figure 9: D-sub 15 socket

Pin	Signal	Function
1	REF-	Reference switch, differential (-)
2	PIEZO-	Motor signal (-)
3	PIEZO+	Motor signal (+)
4	5 V	Supply voltage +5 V
5	n.a.	Not connected
6	ID_CHIP	ID chip data
7	ENCA-	Encoder channel A, differential (-)
8	ENCB-	Encoder channel B, differential (-)
9	PIEZO-	Motor signal (-)
10	GND	GND
11	PIEZO+	Motor signal (+)
12	n.a.	Not connected
13	REF+	Reference switch, differential (+)
14	ENCA+	Encoder channel A, differential (+)
15	ENCB+	Encoder channel B, differential (+)

13.4.2 I/O

Mini-DIN socket, 9-pin, female

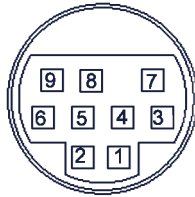


Figure 10: Front view of the mini-DIN socket

Pin	Function
1	Input 1 (digital: TTL)
2	Input 2 (digital: TTL)
3	Input 3 (digital: TTL)
4	Input 4 (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)
Shield	GND

13.4.3 C-170.IO Cable for Connecting to the I/O Socket

Mini-DIN connector, 9-pin, male, open end

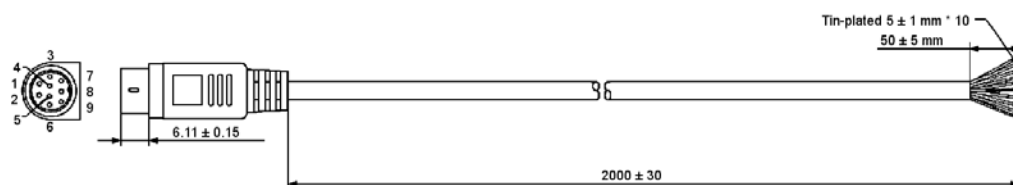


Figure 11: C-170.IO cable

Specifications

Temperature range: -25 °C to +85 °C

Nominal current: 1 A AC/DC

Insulation resistance: 50 MΩ 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-873
1	Black	Input 1 (digital: TTL)
2	white	Input 2 (digital: TTL)
3	Red	Input 3 (digital: TTL)
4	Yellow	Input 4 (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

13.4.4 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

14 Old Equipment Disposal

In accordance with EU law, electrical and electronic equipment may not be disposed of in EU member states via the municipal residual waste.

Dispose of your old equipment according to international, national, and local rules and regulations.

In order to fulfil its responsibility as the product manufacturer, Physik Instrumente (PI) GmbH & Co. KG undertakes environmentally correct disposal of all old PI equipment made available on the market after 13 August 2005 without charge.

Any old PI equipment can be sent free of charge to the following address:

Physik Instrumente (PI) GmbH & Co. KG
Auf der Roemerstr. 1
D-76228 Karlsruhe, Germany



15 EU Declaration of Conformity

For the E-873, an EU Declaration of Conformity has been issued in accordance with the following European directives:

Low Voltage Directive

EMC Directive

RoHS Directive

The applied standards certifying the conformity are listed below.

Safety (Low Voltage Directive): EN 61010-1

EMC: EN 61326-1

RoHS: EN 50581

