User Manual

E-873.1AT DIGITAL CONTROLLER FOR Q-MOTION® AND PIEZOMIKE PIEZO INERTIA DRIVES

Contents

1 Legal Information

The following trademarks are the intellectual property of Physik Instrumente (PI) SE & Co. KG ("PI") and have been entered in the trademark register of the German Patent and Trade Mark Office and, in some cases, also in other trademark registers under the company name of Physik Instrumente (PI) GmbH & Co. KG: PI®, PIC®, PICMA®, PILine®, PIFOC®, PiezoWalk®, NEXACT®, NEXLINE®, PInano®, NanoCube®, Picoactuator®, PicoCube®, PIMikroMove®, PIMag®, PIHera®

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

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.
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 trademark of iC-Haus GmbH.
The following designations are protected company names, trademarks or registered trademarks of other owners:
Linux, MATLAB, MathWorks

Original instructions
First print: 10/24/2024
Document number: PZ273en, ASt

Subject to change. This manual is superseded by any new release. The latest respective release is available for download on our website (www.pi.ws).

Publisher:

Physik Instrumente (PI) SE & Co. KG
Auf der Römerstraße 1
76228 Karlsruhe
Germany

info@pi.de
www.pi.de

Customer service department:

Physik Instrumente (PI) SE & Co. KG
Auf der Römerstraße 1
76228 Karlsruhe
Germany

service@pi.de
www.pi.de

2 Introduction

2.1 About this Document

2.1.1 Objective and Target Group

This user manual contains the information needed for the intended use of the E-873.1AT.

Basic knowledge of closed-loop systems, motion control concepts, and applicable safety measures is assumed.

2.1.2 Other Applicable Documents

The devices and software tools that are mentioned in this documentation are described in separate manuals.

Document number	Document type	Product
SM148E	Software Manual	PIMikroMove
SM146E	Software Manual	GCS Array Data Format Description
SM151E	Software Manual	PI GCS 2.0 DLL0
SM155E	Software Manual	PI MATLAB Driver GCS 2.0
SM158E	Software Manual	PI GCS2 driver library for use with NI LabVIEW software
SM156E	Software Manual	PIStages3Editor: Software for managing the positioner database
SM164E	Software Manual	PIFirmwareManager for updating the firmware
A000T0028	User Manual	PIUpdateFinder: Updating PI Software
A000T0089	Technical Note	PI Software on ARM-Based Platforms
A000T0081	Technical Note	Downloading manuals from PI: PDF file with links to the manuals for digital electronics and software from PI. Supplied with the PI software.

The latest versions of the user manuals can be downloaded (p. 10) at www.pi.ws.

2.1.3 Explanation of Symbols

This chapter explains the symbols and markings used by PI in their user manuals.

Typographic Conventions

Symbol / label	Meaning
1. 2.	Action consisting of one or several steps with strict sequential order
►	Action consisting of one or more steps without relevant sequential order
■	Lists
p. 5	Cross-reference to page 5
RS-232	Label on the product indicating an operating element (example: RS-232 interface socket)
Start > Settings	Menu path in the PC software (example: to open the menu, *Start* and *Settings* must be clicked successively)
`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

Symbols Used

Symbol / Label	Meaning
⚠	General hazard symbol
⚡	Electrical voltage

⚠ ***DANGER***

Dangerous situation

Failure to comply could lead to death or serious injury.

► Precautionary measures for avoiding the risk.

⚠ ***WARNING***

Dangerous situation

Failure to comply could lead to serious injury.

► Precautionary measures for avoiding the risk.

⚠ ***CAUTION***

Dangerous situation

Failure to comply could lead to minor injury.

► Precautionary measures for avoiding the risk.

NOTICE

❗ **Dangerous situation**

Failure to comply could lead to material damage.

► Precautionary measures for avoiding the risk.

Information

Additional information on the E-873.1AT that can affect your application.

2.1.4 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.

2.1.5 Downloading Manuals

Information

If a manual is missing or problems occur with downloading:

► Contact our customer service department (p. 10).

Downloading Manuals

1. Open the website www.pi.ws.
2. Search the website for the product number (e.g., E-873).
3. In the search results, select the product to open the product detail page.
4. Select ***Downloads***.

➔ *The manuals are shown under* **Documentation***. Software manuals are shown under* ***General Software Documentation****.*

5. For the desired manual, select ***ADD TO LIST*** and then ***REQUEST***.
6. Fill out the request form and select ***SEND REQUEST***.

➔ *The download link will be sent to the email address entered in the form.*

2.2 European Declarations of Conformity

For the E-873.1AT, declarations of conformity were issued according to the following European statutory requirements:

- Low Voltage Directive
- EMC Directive
- RoHS Directive

The standards applied for certifying conformity are listed below.

- Safety (Low Voltage Directive): EN 61010-1
- EMC: EN 61326-1
- RoHS: EN IEC 63000

2.3 Customer Service Department

For enquiries and orders, contact your PI representative or send us an email.

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.

Customer service address:

Physik Instrumente (PI) SE & Co. KG
Auf der Römerstraße 1
76228 Karlsruhe
Germany

service@pi.de
www.pi.de

2.4 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 fulfill the responsibility as the product manufacturer, PI undertakes environmentally correct disposal of all PI equipment free of charge, if it was made available to the market after August 13, 2005.

Any old PI equipment can be sent free of charge to the following address:

Physik Instrumente (PI) SE & Co. KG
Auf der Römerstraße 1
76228 Karlsruhe
Germany

info@pi.de
www.pi.de

3 Safety

3.1 Intended Use

The E-873.1AT 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-873.1AT is intended for operating positioners with piezo inertia drives (Q-Motion®, PiezoMike). The E-873.1AT is intended for closed-loop operation with incremental or absolute-measuring position sensors. In addition, it can read and process the reference and limit switch signals from the connected positioner.

The E-873.1AT may not be used for purposes other than those stated in this user manual. In particular, the E-873.1AT may not be used to drive ohmic or inductive loads.

3.2 General Safety Instructions

The E-873.1AT is built according to state-of-the-art technology and recognized safety standards. Improper use of the E-873.1AT may result in personal injury and/or damage to the E-873.1AT.

- ► Use the E-873.1AT only for its intended purpose and when it is in perfect condition.
- ► Read the user manual.
- ► Eliminate any malfunctions that may affect safety immediately.

The operator is responsible for installing and operating the E-873.1AT correctly.

- ► Install the E-873.1AT near the power adapter so that the power plug can be quickly and easily disconnected from the mains.
- ► Use sufficiently dimensioned components to connect the E-873.1AT to the power supply.
- ► Only use cables and connections that comply with local safety regulations.

3.3 Organizational Measures

3.3.1 User Manual

- ► Always keep this user manual available with the E-873.1AT. The latest versions of the user manuals can be downloaded (p. 10) at www.pi.ws.
- ► Add all information from the manufacturer such as supplements or technical notes to the user manual.
- ► If you give the E-873.1AT to a third party, also include this user manual as well as other relevant information provided by the manufacturer.
- ► Only use the device on the basis of the complete user manual. Missing information due to an incomplete user manual can result in minor injury and damage to equipment.
- ► Only install and operate the E-873.1AT after you have read and understood this user manual.

3.3.2 General Personnel Qualification

The E-873.1AT may only be installed, started up, operated, maintained, and cleaned by authorized and appropriately qualified personnel.

4 Product Description

4.1 Front Panel

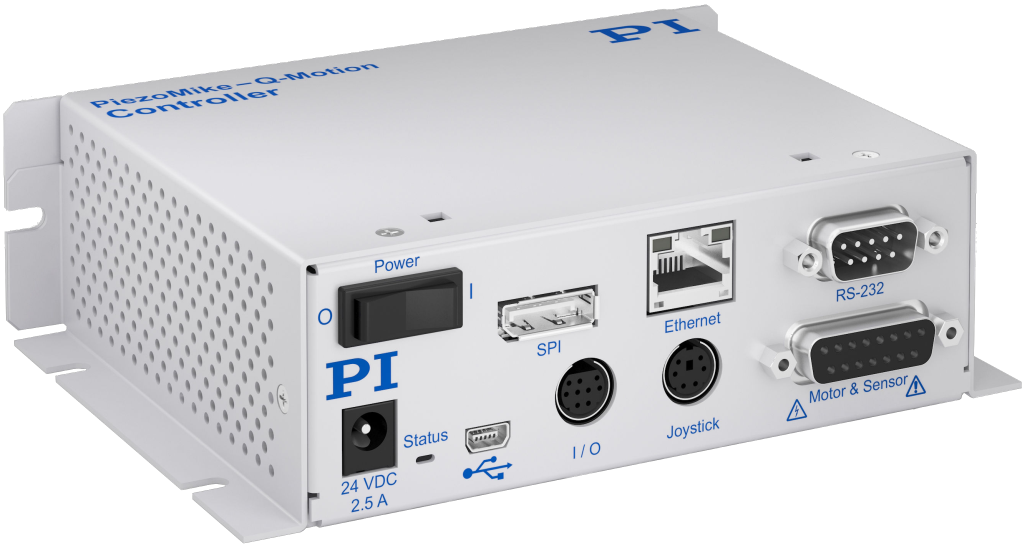
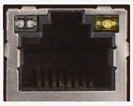
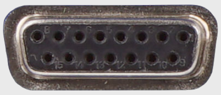


Figure 1: Front view of the E-873.1AT

Element	Labeling	Type	Function
	Power	Toggle switch	On/Off switch O: E-873.1AT switched off —: E-873.1AT switched on
	SPI	Display port	Serial connection to a serial peripheral interface (SPI) master unit
	Ethernet	RJ45	Ethernet interface for communication via TCP/IP
	RS-232	D-sub 9 (m) (p. 243)	Serial connection to PC
	24 VDC 2.5 A	Barrel connector-socket (p. 244)	Connector for the supply voltage
	Status	LED	Controller state: Green - lights up continuously: E-873.1AT ready for normal operation Green - flashing: E-873.1AT in firmware update mode Red: Error Off: E-873.1AT not connected to the supply voltage

Element	Labeling	Type	Function
		Mini-B USB	Universal serial bus for connection to the PC
	I / O	Mini-DIN 9 (f) (p. 242)	Digital inputs/outputs, analog inputs
	Joystick	Mini-DIN 6 (f) (p. 243)	Connector for analog HID (Human Interface Device), e.g., joystick
	Motor & Sensor	D-sub 15 (f) (p. 241)	Connector for the positioning axis

The protective earth connector (p. 49) is on the right-hand rear mounting rail:

Element	Labeling	Type	Function
	⏚	Threaded pin	Threaded pin with mounting hardware for protective earth conductors The threaded pin must be connected to a protective earth conductor because the E-873.1AT is not grounded via the power adapter connector.

4.2 Type Plate

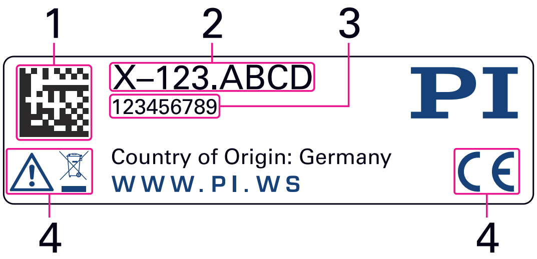


Figure 2: Type plate of the E-873.1AT

1. Data matrix code (example; contains the serial number)
2. Product number (example)
3. Serial number (example), individual for each E-873.1AT
4. Warning and conformity symbols (old equipment disposal (p. 11), CE mark (p. 10))

4.3 Scope of Delivery

Product number	Description
E-873.1AT	Digital controller according to the order
C-501.24050H	Wide range input power supply 24 V, 50 W
3763	Power cord
MS242EK	Short instructions for digital motor controllers and drivers
000036360	USB cable (type A to Mini-B) for connection to the PC
C-990.CD1	Data storage medium with PC software from PI

4.4 Optional Accessories

The following articles are not in the scope of delivery of the E-873.1AT and must be ordered separately, if required.

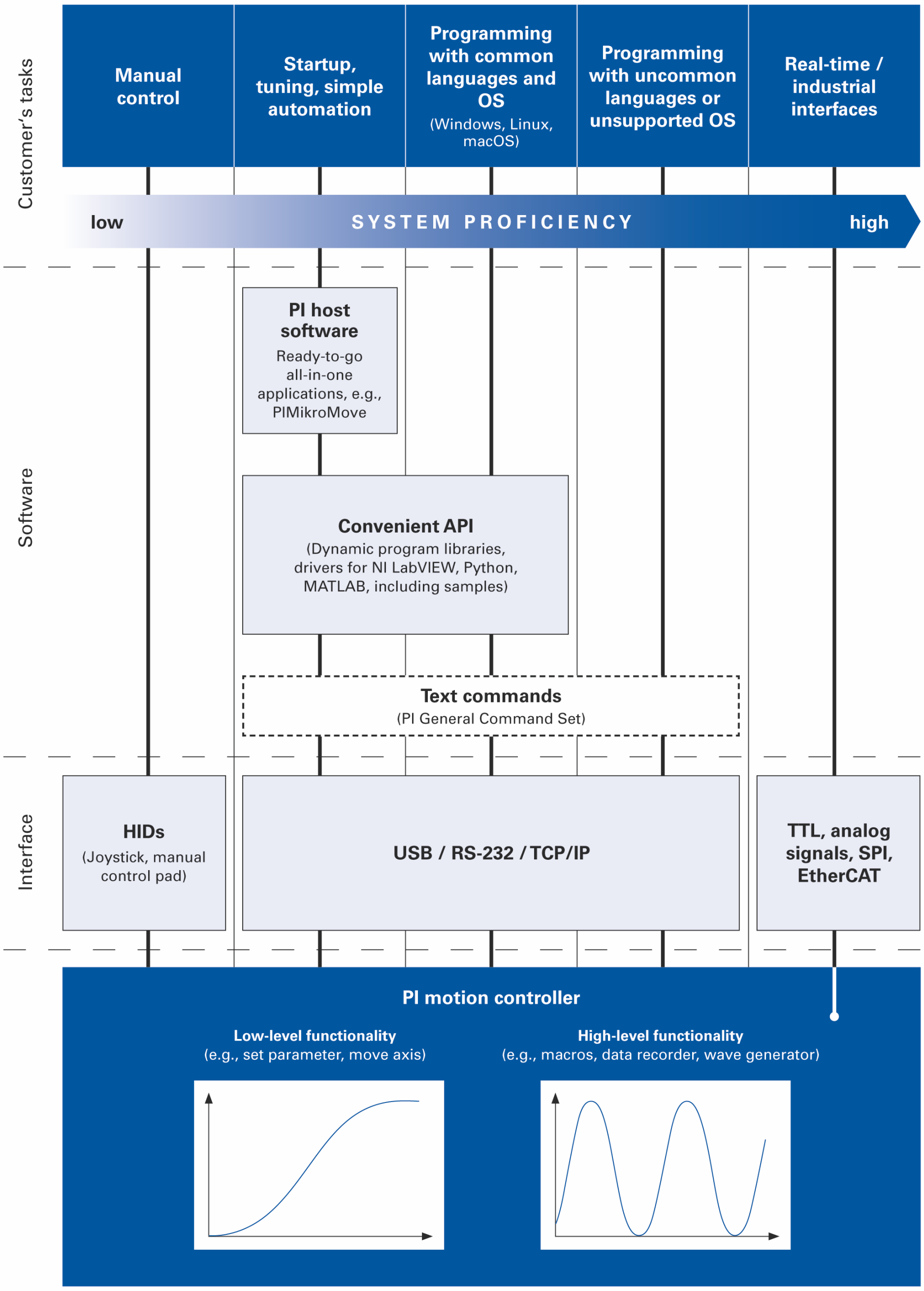
Product number	Description
C-815.553	Straight-through network cable for connecting the PC via a TCP/IP network
C-815.563	Crossover network cable for connecting directly to the PC via TCP/IP
C-815.34	RS-232 null modem cable, 3 m, 9/9-pole
C-819.20	Analog joystick for 2 axes, 2 programmable buttons
C-819.20Y	Y cable for 2 controllers to C-819.20 joystick
C-819.30	Analog joystick for 3 axes, 3 programmable buttons
C-170.PB	Pushbutton box with 4 buttons and 4 LEDs
C-170.IO	I/O cable, 2 m, open end (p. 242)

To order, contact our customer service department (p. 10).

4.5 Communication Interfaces

4.5.1 Controlling PI Systems

Basically, PI systems can be controlled as follows:



Customer's tasks
Manual control
Startup, tuning, simple automation
Programming with common languages and OS (Windows, Linux, macOS)
Programming with uncommon languages or unsupported OS
Real-time / industrial interfaces
low
SYSTEM PROFICIENCY
high
Software
PI host software
Ready-to-go all-in-one applications, e.g., PIMikroMove
Convenient API
(Dynamic program libraries, drivers for NI LabVIEW, Python, MATLAB, including samples)
Text commands
(PI General Command Set)
Interface
HIDs
(Joystick, manual control pad)
USB / RS-232 / TCP/IP
TTL, analog signals, SPI, EtherCAT
PI motion controller
Low-level functionality
(e.g., set parameter, move axis)
High-level functionality
(e.g., macros, data recorder, wave generator)

4.5.2 E-873.1AT Interfaces

The E-873.1AT can be controlled via the following communication interfaces:

- PC interfaces, via software (p. 17) or PI General Command Set:
  - TCP/IP
  - USB
  - RS-232
- SPI (if you wish to use the SPI interface, contact the customer service department (p. 10).)
- Analog signals
- Digital Signals: TTL

The interface parameters in the E-873.1AT's volatile memory can be queried with the IFC? command and changed with IFC command. The IFS? and IFS commands are available for querying and changing the interface parameters in the E-873.1AT's nonvolatile memory.

The number of input lines available on the E-873.1AT can be queried with the TAC? (p. 173) command. The voltage at the analog inputs can be queried with the TAV? (p. 173) command.

The number of available digital input and output lines to and from the E-873.1AT can be queried with the TIO? (p. 173) command. The status of the digital input lines can be queried with the DIO? (p. 132) command.

TCP/IP

Interface parameters of the E-873.1AT for TCP/IP communication:

Interface property	Factory setting	Note
IP address format: <IP-Adresse>:<Portnummer> (IPADR)	000.000.000.000: 50000	Is not used when an IP address is assigned to the E-873.1AT by a DHCP server (IPSTART = 1). <Portnummer> cannot be changed.
Subnet mask (IPMASK)	24	Bit mask for subnet (24 = subnet mask 255.255.255.0
Startup behavior for configuring the IP address for TCP/IP communication (IPSTART)	1	0: The address defined by IPADR is used. 1: DHCP is used to get the E-873.1AT's address. The default setting of the startup behavior only needs to be changed if the network devices are to use static addresses instead.
MAC level address (MACADR)		Invariable, unique network hardware address

RS-232

E-873.1AT's interface settings for RS-232 communication:

Interface setting	Factory setting	Note
Baud rate (RSBAUD)	115200	Possible values: 9600, 19200, 38400, 115200 To establish communication successfully, the baud rates of the E-873.1AT and PC must match.

4.6 Software Overview

A data storage device with the PI Software Suite is included in the E-873.1AT's scope of delivery (p. 15). Some components of the PI Software Suite are described in the table below. For information on the compatibility of the software with PC operating systems see the C-990.CD1 Release News in the root directory of the data storage device.

Libraries, drivers

PC software	Operating system	Short description	Recommended use
Dynamic program-libraries for GCS	Windows,Linux	Allows software programming of the E-873.1AT with programming languages such as C++. The functions in dynamic program libraries are based on the PI General Command Set (GCS).	For users who would like to use a dynamic program library for their application. Is required for PIMikroMove. Is required for NI LabVIEW drivers if communication is to be established via USB (with Linux only via virtual COM port) or a daisy chain network.
Drivers for use with NI LabVIEW software	Windows,Linux	NI LabVIEW is a software for data acquisition and process control (must be ordered separately from National Instruments). The NI LabVIEW software from PI is a collection of virtual instrument drivers (VI drivers) for PI controllers. These drivers support GCS commands.	For users who wish to use NI LabVIEW to program their application.
MATLAB drivers	Windows	MATLAB is a development environment and programming language for numerical calculations (must be ordered separately from MathWorks). The PIMATLAB 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 wish to use MATLAB to program their application.
USB driver	Windows	Driver for the USB interface	For users who want to connect the controller to the PC via the USB interface.

User software

PC software	Operating system	Short description	Recommended use
PIMikroMove	Windows	Graphical user interface for Windows, which can be used for controllers from PI: ■ Start the system without programming effort ■ Graphic representation of the motion ■ 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 perform simple automation tasks or test their equipment before or instead of programming an application. No command knowledge is necessary to operate PIMikroMove. A log window showing the commands sent makes it possible to learn how to use the commands.
PIStages3Editor	Windows	Program opening and editing positioner databases in .db format.	For users who want to deal with the contents of positioner databases more intensively.
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.
PIFirmwareManager	Windows	Program for updating the firmware of the E-873.1AT	For users who want to update the firmware
PIUpdateFinder	Windows	Checks the PI software installed on the PC. If newer versions of the PC software are available on the PI server, they are offered for download.	For users who want to update the PC software.

4.7 Positioner database

You can select a parameter set appropriate 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 controller's volatile or nonvolatile memory.

Database file name	Description
PISTAGES3.DB	Delivery includes parameter sets for all standard positioners from PI and PI miCos, and is saved to the PC automatically during installation of the PC software. New parameter sets can be created, edited, and saved.
Product code.db	Includes the parameter set for the custom positioner "product code". In order for the parameter set to be selectable in the PC software, it must be imported (p. 51) into PISTAGES3.DB first.

The positioner database only contains some of the information that is required to operate a positioner with the E-873.1AT. When the positioner connected to the E-873.1AT is equipped with an ID chip: Further information is loaded as parameter values from the ID chip to the volatile memory of the E-873.1AT when the E-873.1AT is switched on or rebooted.

4.8 ID Chip Detection

PI positioners with piezo inertia drives have an ID chip in the connector where the following data is stored as parameters:

- Information on the positioner: Type, serial number, date of manufacture, hardware version
- Signal type output by the position sensor

The data of the connected positioner is loaded from the ID chip into the volatile memory of the E-873.1AT when the E-873.1AT is switched on or rebooted.

The ID chip only contains some of the information that is required to operate the positioner with the E-873.1AT. When you use the PC software from PI, further information is loaded as parameter values from a positioner database into the volatile memory of the E-873.1AT.

The parameter values in the E-873.1AT's volatile memory can be queried and written to the nonvolatile memory (refer to Adapting Settings (p. 108)).

4.9 Functional Principles of the E-873.1AT

4.9.1 Block Diagram

The E-873.1AT controls the motion of a positioner's logical axis. The following block diagram shows how the E-873.1AT generates the piezo voltage for the axis connected:

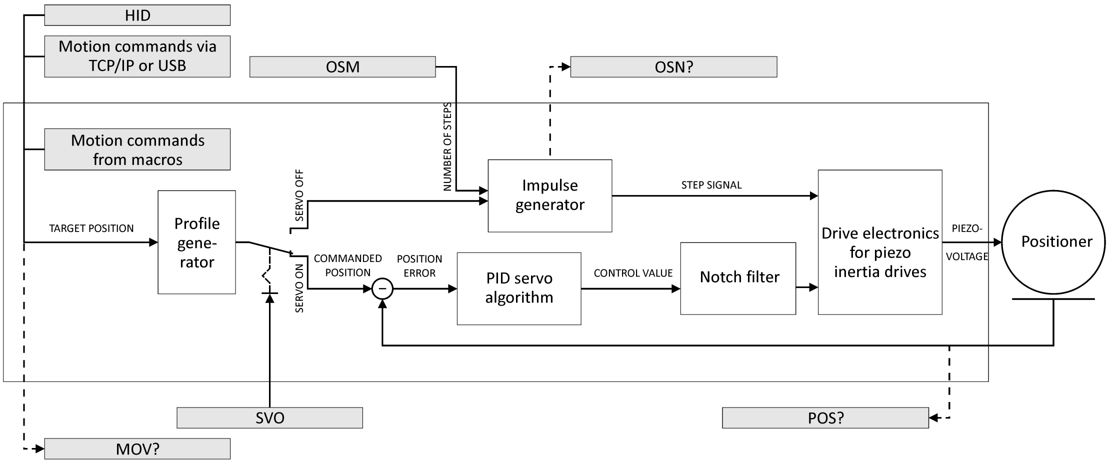


Figure 3: E-873.1AT: Generating the piezo voltage

The E-873.1AT supports positioners with piezo inertia drives and incremental or absolute-measuring positioning sensors.

4.9.2 Important Firmware Components

The functional units of the E-873.1AT's firmware are described in the following.

The firmware can be updated with a tool. The current firmware version can be ascertained with the VER? command.

Component	Description
Parameters	Parameters reflect the properties of the positioner connected (e.g., travel range) and specify the behavior of the E-873.1AT (e.g., settings for the servo algorithm). The parameters can be divided into the following categories: ■ Protected parameters whose default settings cannot be changed ■ Parameters that must be set by the user to adapt to the application 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-873.1AT. Command levels determine the write permission for the parameters. The current command level can be queried with the CCL? command and changed with the CCL command. This may require entering a password. The list of parameters available in the E-873.1AT can be queried with the HPA? command. Refer to Adapting Settings (p. 108) for more information on parameters
ASCII commands (GCS)	Communication with the E-873.1AT can be made with commands (p. 117) from the PI General Command Set (GCS). The GCS syntax version (p. 106) can be queries with the CSV? command. Examples of the use of GCS: ■ Configuring the E-873.1AT ■ Setting the operating mode ■ Starting positioner motion ■ Getting system and position values The list of commands available in the E-873.1AT can be queried with the HLP? command.
Profile Generator	The profile generator does calculations in closed-loop operation to specify the target position, velocity, and acceleration of an axis for each point in time during motion. The result is the dynamics profile. For information, see: Generating the Dynamics Profile (p. 31)
Servo algorithm	Closed-loop operation: The position error that results from the difference between the commanded target position and the actual position (sensor feedback) runs through a servo algorithm. For information, see: Servo Algorithm and Other Control Value Corrections (p. 35)
Data recorder	The E-873.1AT contains a real-time data recorder. The data recorder can record various signals (e.g., position, control value) from different data sources (e.g., logical axes). Refer to Data Recorder (p. 80) for information

Component	Description
Macros	The E-873.1AT can save macros. 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-873.1AT is switched on or rebooted. The startup macro simplifies stand-alone operation (operation without a connection to the PC). Refer to Controller Macros (p. 97) for information

Commands

C		Page
CCL	Set Command Level	125
CCL?	Get Command Level	125
CSV?	Get Current Syntax Version	126
H		**Page**
HLP?	Get List Of Available Commands	147
HPA?	Get List Of Available Parameters	148
V		**Page**
VER?	Get Versions Of Firmware And Drivers	177

Parameters

4.9.3 Commandable Items

The following table contains the elements of the E-873.1AT that can be commanded with GCS commands.

Element	Quantity	ID	Description
Logical axis	1	1*	A logical axis represents the motion of the mechanics in E-873.1AT's firmware. It corresponds to the axis of a linear coordinate system. Motion for logical axes is commanded in the firmware of the E-873.1AT (i.e., for the directions of motion of a positioner). 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-_ The positioner type connected to an axis can be queried with the `CST?` command. 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`.
Analog inputs	8	1 to 8	The analog input lines with the identifiers 1 to 4 are the inputs 1 to 4 of the *I/O* socket. Their number is displayed with the `TAC?` command and their values can be queried with the `TAV?` command. Note that these lines can also be used as digital inputs (see below). Additional analog input lines are located at the *Joystick* socket. These lines are not output via `TAC?` and `TAV?`. The values of all inputs can be recorded via record option 81 of the `DRC` command. For information, see: Analog Input Signals (p. 89)
Digital outputs	4	1 to 4	1 to 4 identify digital output lines 1 to 4 of the *I/O* socket. For information, see: Digital Output Signals (p. 83)
Digital inputs	4	1 to 4	1 to 4 identify digital input lines 1 to 4 of the *I/O* socket, which can also be used as analog inputs (see above). For information, see: Digital Input Signals (p. 86)
HID (Human Interface Device)	1	1	The HID (e.g., joystick) is used for HID control of the E-873.1AT's logical axis. Information on the HID's axes and buttons can be queried with the `HIS?` command. For information, see: Controlling with an HID (p. 90)
HID axes	4	1 to 4	Two of the HID's axes can be connected to the *Joystick* socket. Connection options: ■ Pin 4 (0 to 3.3 V): Command as axis 1 of the HID ■ Pin 2 (0 to 3.3 V): Command as axis 2 of the HID

Element	Quantity	ID	Description
			Two further axes of the HID can be connected to the *I/O* socket. Connection options: ■ Pins 1 and 2 (TTL signals): Command as axis 3 of the HID ■ Pins 3 and 4 (TTL signals): Command as axis 4 of the HID
HID buttons	2	1, 2	The two buttons of the human interface device can be connected to the *Joystick* socket. Connection options: ■ Pin 5 (0 or 3.3 V): Command as button 1 of the HID ■ Pin 6 (0 or 3.3 V): Command as button 2 of the HID
Data recorder tables	4	1 to 4	The E-873.1AT has 4 data recorder tables (query with `TNR?`) with 7616 data points per table. For information, see: Data Recorder (p. 80)
Overall system	1	1	E-873.1AT as an overall system Information on name, serial number, and firmware version of the E-873.1AT can be queried with the `*IDN?` command. The ready state of the E-873.1AT can be queried with the `#7` command.

* modifiable

Commands

#		Page
#7	Request Controller Ready Status	122
*		Page
*IDN?	Get Device Information	123
C		Page
CST?	Get Assignment Of Stages To Axes	126
D		Page
DRC	Set Data Recorder Configuration	133
H		Page
HIS?	Get Configuration Of HI Device	146
S		Page
SAI	Set Current Axis Identifiers	166
SAI?	Get List Of Current Axis Identifiers	167
T		Page
TAC?	Tell Analog Channels	173
TAV?	Get Analog Input Voltage	173

TNR?	Get Number Of Record Tables	174

Parameters

0x3C	Stage Name	Positioner name. Default value: NOSTAGE String up to 20 characters

4.9.4 Physical Units

The E-873.1AT supports various units of length for positions. Adapting is done by a factor that converts the sensor counts into the desired physical unit of length. The conversion factor is set with the Numerator Of The Counts-Per-Physical-Unit Factor parameter (0xE) and Denominator Of The Counts-Per-Physical-Unit Factor parameter (0xF).

The unit symbol can be customized for display purposes with the Axis Unit parameter (0x7000601). The parameter Number of Decimal Places (0xE000102) defines the number of decimal points of floating point numbers.

Examples:

- 1 Encoder count = 100 nm
  Counts per physical unit of length: 10000:1
  → unit symbol: mm
- 1 Encoder count = 0.254 mm
  Counts per physical unit of length: 100:1
  → unit symbol: inch

Commands

Parameters

0xE	Numerator Of The Counts-Per-Physical-Unit Factor	Numerator of the factor for converting impulses to physical units. 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. 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. Decimal number

0xF	Denominator Of The Counts-Per-Physical-Unit Factor	Denominator of the factor for converting impulses to physical units. 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. 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. Decimal number
0x7000601	Axis Unit	Unit symbol. 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". String up to 20 characters
0xE000102	Number of Decimal Places	Number of decimal places for floating point numbers.

4.9.5 Operating Modes

Working Principle of a Piezo Inertia Drive

Piezo inertia drives are piezo-based drives with virtually unlimited travel range. They use 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. The operating voltage is therefore output by the E-873.1AT as a modified sawtooth signal with a maximum frequency of 20 kHz. The level of the operating voltage depends on the connected positioner and is automatically set when the corresponding set of parameters is loaded from the positioner database (p. 19).

The output of one period of the modified sawtooth signal generates one "step" of the runner.

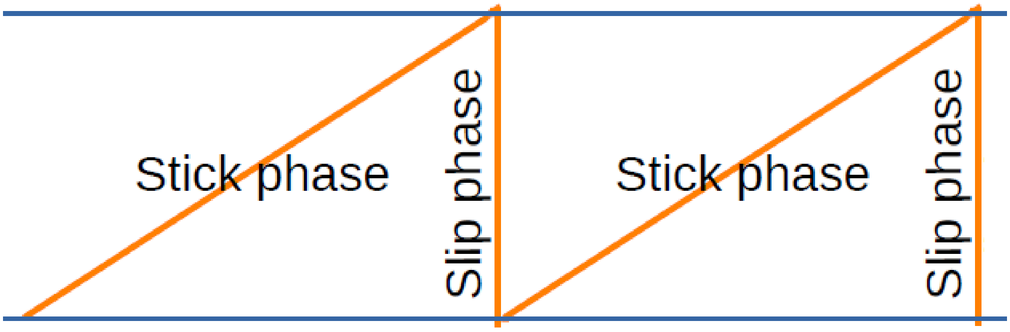


Figure 4: Stick-slip principle of a piezo inertia drive

A delay time between the piezo inertia drive's stick and the slip phases can be set via the PIShift Plateau And Slip Time parameter (0x1F0007A0). This results in smoother motion.

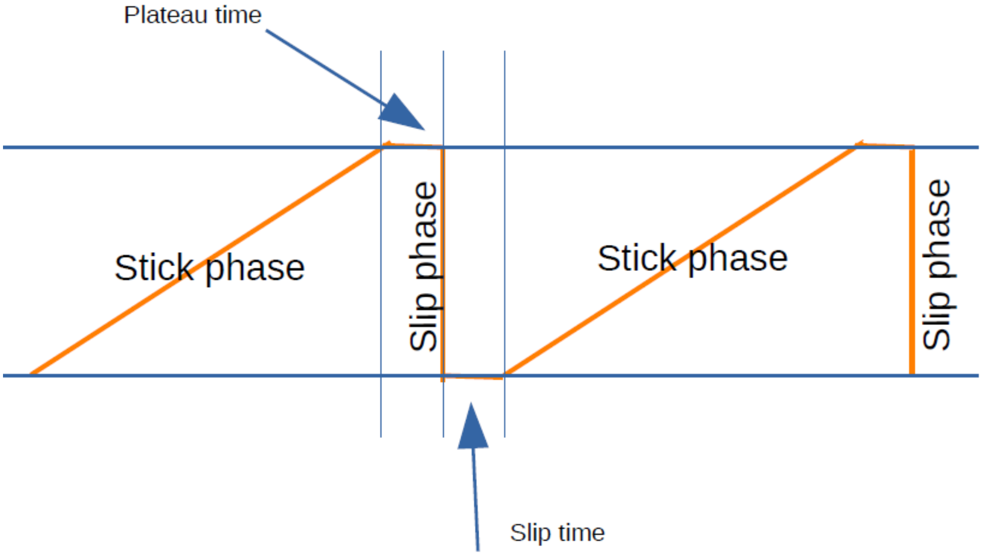


Figure 5: Stick-slip principle with delay time

Parameters for configuring the piezo inertia drive

These parameters influence the piezo voltage curve (applies to closed-loop and open-loop operation):

- PIShift Upper Supply Voltage (V) (0x1F000000)
  The parameter determines the level of the operating voltage. It depends on the connected positioner and is automatically set when the corresponding set of parameters is loaded from the positioner database. A wrong setting of this value can present a danger for the user and damage the connected positioner. Therefore, do **not** change the automatically set value!
- PIShift Forward Current (A) (0x1F000200)
- PIShift Backward Current (A) (0x1F000300)
- PIShift Offset Current (A) (0x1F000703)

Servo mode

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

- Closed-loop operation (servo mode On)
  A profile generator calculates the dynamics profile from the values specified for target position, velocity, acceleration, and deceleration.
  The position error resulting from the difference between the dynamics profile generated and the actual position (sensor feedback) runs through a PID servo algorithm (proportional-integral-derivative). Additional corrections can be made as well.
  The result is the control value that is converted into the control voltage for the drive electronics integrated in the E-873.1AT.
- Open-loop operation (servo mode Off)
  There is no servo control via a servo algorithm in open-loop operation. The E-873.1AT does not evaluate the position sensor's signals.
  Motion is possible in two ways: At a frequency determined by the level of the input voltage, or constantly at a fixed frequency. Use the PIShift Open-Loop Driving Mode paramter (0x1F000702) for this setting.

Information

The E-873.1AT is intended for closed-loop operation with incremental or absolute position sensors (servo mode On). After switch-on, open-loop operation is active by default (servo mode Off).

► Query the current operating mode with the SVO?, #4 or SRG? commands.

► Activate closed-loop operation with the SVO command.

► If necessary, program a startup macro that starts the E-873.1AT in closed-loop operation via the SVO command.

► Avoid motion in open-loop operation.

Motion in closed-loop operation

In order to adapt motion in closed-loop operation to different requirements, one of three modes can be set via the PIShift Closed-Loop Driving Mode parameter (0x1F0007A3):

- Q-Motion® mode (0): Operating mode for Q-Motion® drives
  The movement is done in fast step mode. There is an analog step-and-settle at the target position at the end of the movement.
- PiezoMike mode (1): Operating mode for PiezoMike drives
  The movement begins in fast step mode and continues in single-step mode. There is no analog step-and-settle at the end of the movement. This mode makes sure that the target position is reached quickly, and less so for nanometer-precision positioning.
- Silent mode (2): Silent operating mode for Q-Motion® drives
  This operating mode is identical to the Q-Motion® mode, however, the piezo inertia drive is excited by a frequency inaudible to humans so that noise emission is reduced considerably.

Depending on the operating mode selected, motion consists of the following sequences:

Operating modes	Motion sequences
Q-Motion® modes (0, 2)	1. Fast step mode (multi step) with variable frequencies 2. Analog step-and-settle at the target position
PiezoMike mode (1)	1. Fast step mode (multi step) with fixed frequency 2. Single-step mode

PiezoMike mode:

In fast step mode, the connected positioner is controlled with a fixed frequency. This results in the velocity travelled. It is constant and cannot be changed. However, fluctuations may occur, for example, due to friction or spindle tolerances.

When selecting the PiezoMike mode, the following parameters will be evaluated for motion in closed-loop operation:

Parameter	Description
PiezoMike Single Step Enter Factor (0x1F0007A7)	Entrance window for step mode
PiezoMike Single Step Exit Factor (0x1F0007A8)	Exit window for step mode
PIShift Delay (ms) (0x1F000701)	Delay time when switching between fast step (multi-step) mode and single-step mode of motion

Commands

#		Page
#4	Request Status Register	121
S		**Page**
SRG?	Query Status Register Value	170
SVO	Set Servo Mode	172
SVO?	Get Servo Mode	173

Parameters

0x1F0007A0	PIShift Plateau And Slip Time (µs)	Delay time between the stick and the slip phases of the piezo inertia drive. Depending on the velocity, minimal overshooting of the piezo actuator occurs at the end of the stick and the slip phases. The delay time gives the piezo actuator time to compensate for overshooting before the next phase of motion begins. This results in smoother motion.
0x1F000000	PIShift Upper Supply Voltage (V)	Maximum output voltage for piezo inertia drives. The level of the voltage depends on the connected positioner and is automatically set when the corresponding set of parameters is loaded from the positioner database. A wrong setting of this value can present a danger for the user and damage the connected positioner. Therefore, do not change the automatically set value! The value depends on the type of the drive.
0x1F000200	PIShift Forward Current (A)	Maximum output current for piezo inertia drives during forward motion. The value depends on the type of the drive.
0x1F000300	PIShift Backward Current (A)	Maximum output current for piezo inertia drives during backward motion. The value depends on the type of the drive.
0x1F000703	PIShift Offset Current (A)	Offset current in open-loop operation - 0.05 to 0.05 The preset value should not be changed.
0x1F000702	PIShift Open-Loop Driving Mode	Drive mode for piezo inertia drives in open-loop operation. This parameter is evaluated when motion is started in open-loop operation. The value of the parameter has no influence on motion started with the STE command in open-loop operation.

0x1F0007A3	PIShift Closed-Loop Driving Mode	Drive mode in closed-loop operation In order to adapt motion in closed-loop operation to different requirements, different modes can be set via this parameter: 0 - Q-Motion® mode 1 - PiezoMike mode 2 - Silent mode for Q-Motion®
0x1F0007A7	PiezoMike Single Step Enter Factor	Entrance window for single-step mode in PiezoMike mode for motion in closed-loop operation. Commanded movement is done in fast step mode (multi-step) until the entrance window for single-step mode has been reached. Motion continues in single-step mode until the entrance window around the target position (parameter 0x1F0007A4) has been reached. This parameter will only be evaluated if 0x1F0007A3 is set to 1 (PiezoMike mode). Specified as entrance window factor around the target position
0x1F0007A8	PiezoMike Single Step Exit Factor	Exit window for single-step mode in PiezoMike mode for motion in closed-loop operation. Defines a tolerance range around the entrance window for single-step mode (parameter 0x1F0007A7) to maintain single-step mode. When the current position leaves the exit window, the system switches to fast step mode (multi step) to reach the target position. This parameter will only be evaluated if 0x1F0007A3 is set to 1 (PiezoMike mode). Specified as exit window factor around the target position (parameter 0x1F0007A5)
0x1F000701	PIShift Delay (ms)	Delay time when switching between fast step (multi-step) mode and single-step mode of motion in PiezoMike mode. 0.05 to 100 [ms]

4.9.6 Generation of Dynamics Profile

The profile generator does calculations in closed-loop operation to specify the target position, velocity, and acceleration/deceleration of the axis for each point in time during motion (dynamics profile). The values calculated are called commanded values.

The dynamics profile generated by the profile generator of the E-873.1AT depends on the motion variables that are specified by parameters and/or HID.

Variables for Generating a Dynamics Profile

Acceleration (A)

Commands	Parameters	Remarks
ACC ACC?	Closed-Loop Acceleration (Phys. Unit/s2) (0xB) Change using the ACC command or SPA / SEP; query with ACC?	Is limited by the Maximum Closed-Loop Acceleration (Phys. Unit/s²) parameter (0x4A).

Deceleration (D)

Commands	Parameters	Remarks
DEC DEC?	Closed-Loop Deceleration (Phys. Unit/s2) (0xC) Change with the DEC command or with SPA / SEP; query with DEC?	Is limited by the Maximum Closed-Loop Deceleration (Phys. Unit/s²) parameter (0x4B).

Velocity (V)

Commands	Parameters	Remarks
VEL VEL?	Closed-Loop Velocity (Phys. Unit/s) (0x49) Change with the VEL command or with SPA / SEP; query with VEL?	Is limited by the Maximum Closed-Loop Velocity (Phys. Unit/s) (0xA) parameter.

The values for velocity (0x49), acceleration (0xB), and deceleration (0xC) are not evaluated for fast step mode in PiezoMike mode.

Target position at the end of the movement

Commands	Parameters	Remarks
MOV MVR	--	The soft limits are set as the respective target position during HID control of the velocity. The E-873.1AT sets the target position to the current position of the axis in the following cases: ■ Disabling HID control for the axis ■ Switching servo mode on with the SVO command ■ Stopping the motion with the #24, STP, or HLT commands

Explanation of the Dynamics Profile

The profile generator of the E-873.1AT only supports trapezoidal velocity profiles: The axis accelerates linearly (based on the acceleration value specified) until it reaches the specified velocity. It continues to move at this velocity until it decelerates linearly (based on the deceleration value specified) and stops at the specified target position.

The following applies in the figures: A = acceleration, D = deceleration, V = velocity

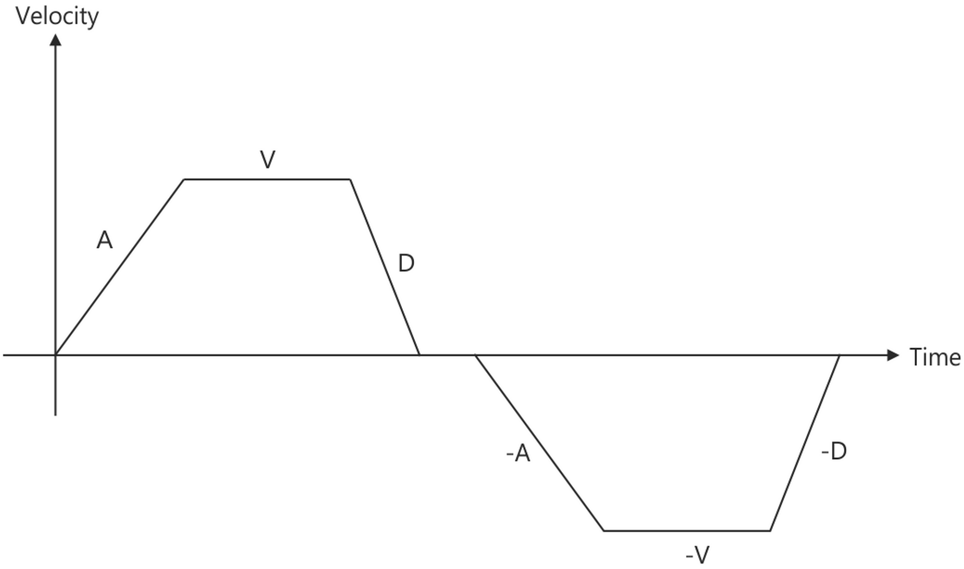


Figure 6: Simple trapezoid velocity profile

If deceleration has to begin before the axis reaches the specified velocity, the profile will not have a constant velocity portion and the trapezoid becomes a triangle.

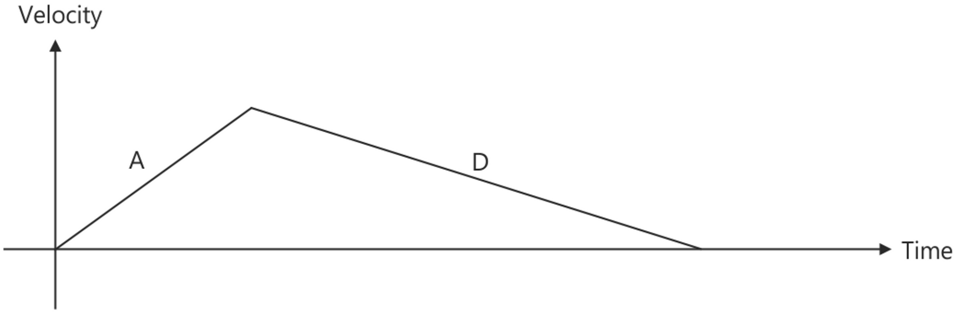


Figure 7: Simple trapezoid velocity profile without constant velocity

The edges for acceleration and deceleration can be symmetrical (acceleration = deceleration) or asymmetrical (acceleration ≠ deceleration). The acceleration value is always used at the start of the motion. After that, the acceleration value is used during an increase in the absolute velocity and the deceleration value during a decrease in the absolute velocity. If none of the motion variables are changed during the course of motion, the acceleration value is used until the maximum velocity is reached and the deceleration value is used for decreasing the velocity down to zero.

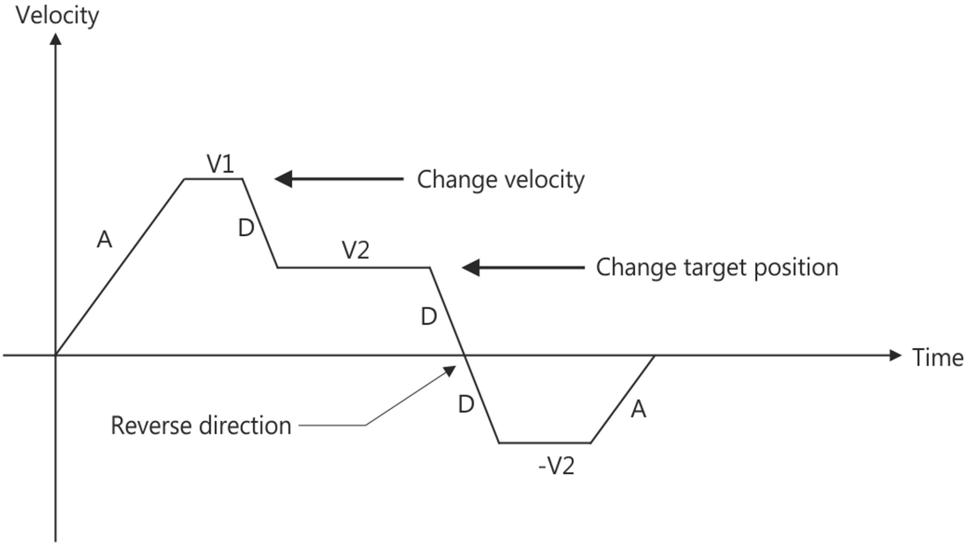


Figure 8: Complex trapezoidal profile with parameter changes

All motion variables can be changed while the axis is in motion. The profile generator will always attempt to stay within the permissible motion limits specified by the motion variables. If the target position is changed during motion so that overshooting is unavoidable, the profile generator will decelerate to a complete stop and reverse the direction of motion in order to reach the specified position.

Commands

A		Page
ACC	Set Closed-Loop Acceleration	124
ACC?	Get Closed-Loop Acceleration	124
D		Page
DEC	Set Closed-Loop Deceleration	129
DEC?	Get Closed-Loop Deceleration	130
M		Page
MOV	Set Target Position	160
MVR	Set Target Relative To Current Position	161
S		Page
SEP	Set Nonvolatile Memory Parameters	167
SPA	Set Volatile Memory Parameters	168
V		Page
VEL	Set Closed-Loop Velocity	176
VEL?	Get Closed-Loop Velocity	177

Parameters

0xB	Closed-Loop Acceleration (Phys. Unit/s^2)	Acceleration with dynamics profile in closed-loop operation. 0 to value of 0x4A [phys. units. / s2]

0x4A	Maximum Closed-Loop Acceleration (Phys. Unit/s2)	Maximum acceleration in closed-loop operation with dynamics profile. Specifies the maximum value for parameter 0xB.
0xC	Closed-Loop Deceleration (Phys. Unit/s^2)	Deceleration with dynamics profile in closed-loop operation. 0 to value of 0x4B [phys. units. / s2]
0x4B	Maximum Closed-Loop Deceleration (Phys. Unit/s2)	Maximum deceleration in closed-loop operation with dynamics profile. Specifies the maximum value for parameter 0xC.
0x49	Closed-Loop Velocity (Phys. Unit/s)	Velocity in closed-loop operation with dynamics profile. 0 to value of 0xA [phys. units / s]
0xA	Maximum Closed-Loop Velocity (Phys. Unit/s)	Maximum velocity with dynamics profile in closed-loop operation. Specifies the maximum value for parameter 0x49.

4.9.7 Servo Algorithm and Other Control Value Corrections

In closed-loop operation for Q-Motion® drives, the control value for the drive electronics integrated in the E-873.1AT and therefore the settling behavior is optimized by a servo algorithm and a notch filter.

Settings for the servo algorithm
The position error resulting from the difference between the commanded position and the actual position (sensor feedback) runs through a servo algorithm. Servo control parameters set the servo algorithm. The optimal servo control parameter setting depends on your application and your requirements (refer to "Optimizing Servo Control Parameters" (p. 72).

The servo algorithm depends on the Q-Motion® drive mode set in the **PIShift Closed-Loop Driving Mode** parameter (0x1F0007A3 (p. 228)). Setting the Q-Motion® mode (0x1F0007A3 = 0) applies a PID servo algorithm (proportional-integral-derivative), setting the silent mode for Q-Motion® (0x1F0007A3 = 2) applies a PI servo algorithm (proportional-integral).

Q-Motion® mode: PID servo algorithm

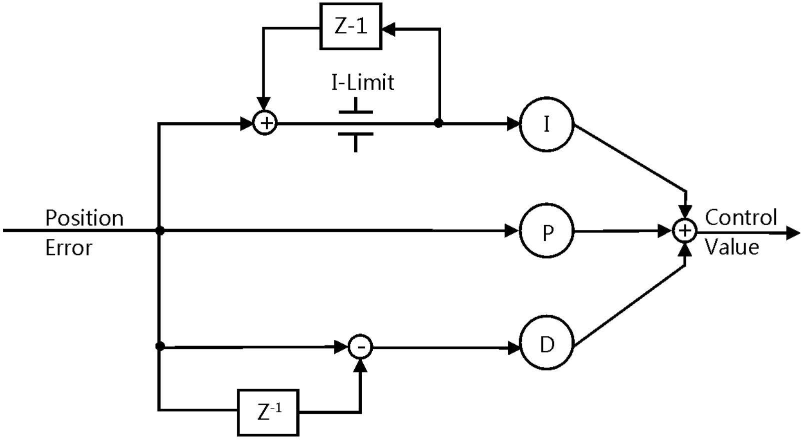


Figure 9: PID algorithm

The PID servo algorithm uses the following servo control parameters.

- P Term (0x1)
- I Term (0x2)
- D Term (0x3)

- I Limit (0x4)
- D-Term Delay (No. Of Servo Cycles) (0x71)

Silent mode for Q-Motion®: PI servo algorithm
The PI servo algorithm uses the following servo control parameters.

- PIShift P Term, Silent Mode (0x1F000711)
- PIShift I Term, Silent Mode (0x1F000712)

Servo algorithm input can be configured for the E-873.1AT with the following parameters:

- Numerator Of The Servo-Loop Input Factor (0x5A)
- Denominator Of The Servo-Loop Input Factor (0x5B)

The servo-loop input factor decouples the servo control parameters from the encoder resolution. The servo-loop input factor is independent of the factor for counts per physical length unit (0xE (p. 193) and 0xF (p. 193)). 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. It filters the part of the frequency that would cause undesired resonance in the mechanics. The notch filter can be configured with the following parameters: (refer to Setting the Notch Filter (p. 68)):

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

Commands

Parameters

0x1	P term	Proportional constant of the PID servo algorithm. Is used for fast correction of the position error. 0 to 65535
0x2	I term	Integration constant of the PID servo algorithm. Used for reducing static position error. 0 to 65535
0x3	D term	Differential constant of the PID servo algorithm. Used for damping rapid control oscillation. The D term can be calculated as a floating average over several servo cycles. The parameter 0x71 (D-Term Delay) specifies how many values (i.e., servo cycles) are to be used for calculating the average. 0 to 65535 The preset value of this parameter should not be changed.
0x4	I limit	Limit of the integration constant. 0 to 1048575
0x71	D-Term Delay (No. Of Servo Cycles)	D term delay. Determines how many values (i.e., servo cycles) are used for calculating the mean value of the D term. 0 to 15. The preset value of this parameter should not be changed.

0x1F000711	PIShift P Term, Silent Mode	Proportional constant of the PI servo algorithm for the silent mode of Q-Motion® drives. The proportional constant is used for fast correction of the position error. This parameter will only be evaluated if 0x1F0007A3 is set to 2 (silent mode). 0 to 65535
0x1F000712	PIShift I Term, Silent Mode	Integration constant of the PI servo algorithm for the silent mode of Q-Motion® drives. The integration constant is used for reducing static position errors. This parameter will only be evaluated if 0x1F0007A3 is set to 2 (silent mode). 0 to 65535
0x5A	Numerator Of The Servo-Loop Input Factor	Input factor numerator for the servo loop. Decimal number
0x5B	Denominator Of The Servo-Loop Input Factor	Input factor denominator for the servo loop. Decimal number
0x94	Notch Filter Frequency 1 (Hz)	Frequency of the first notch filter. The appropriate frequency component is reduced in the control value to compensate for undesired resonances in the mechanics. 40 to 20,000 [Hz]
0x95	Notch Filter Edge 1	Edge steepness of the first notch filter. The greater the value of this parameter, the narrower the notch filter bandwidth. 0.1 to 10 (dimensionless)

4.9.8 Triggering Motion

Motion is triggered in **closed-loop** operation either by commands or an HID, e.g., a joystick.

HID control is not possible in **open-loop** operation.

The motion status of the axes connected to the E-873.1AT can be queried with the #5 command.

Motion triggered by commands can be **stopped** with the following commands:

- #24, STP: Abrupt stop
- HLT: Gentle stop

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

Triggering motion in closed-loop operation

Motion is triggered in closed-loop operation either by commands or an HID, e.g., a joystick.

Motion commands for an axis are not permitted when HID control is activated for the axis.

Trigger of the motion	Commands	Description
Commands for point-to-point motion sent from the command line or via the PC software	MOV, MVR	Motion to absolute or relative target position
	STE	Jumps a specified distance and records the response
	FNL, FPL, FRF	Starts reference moves
	FED	Starts moves 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 while a macro is running on the controller. The macro content and motion commands received from the command line can overwrite each other.
HID control	HIN	Activates or deactivates control of the E-873.1AT's axes via the HID's axes.
	HIA	Configures HID control for the axes of the E-873.1AT. The following motion parameters of the E-873.1AT's axes can be controlled via the HID's axes: ■ Absolute target position ■ Relative target position (frequency with which the controlled axis of the mechanics is moved) ■ Velocity of the axis ■ Maximum velocity of the axis

Information

Absolute target positions can only be commanded for positioners with incremental position sensor when the axis has been previously referenced, refer to Referencing (p. 43).

Triggering motion in open-loop operation

HID control is not possible in open-loop operation.

Motion is triggered by the following commands:

Commands	Description
STE	Initiates a jump by a specified number of steps and records the response.
OSM	Moves an axis by a particular number of steps The number of steps that the axis still must do can be queried with the OSN? command.

Commands

#		Page
#24	Stop All Axes	123
#5	Request Motion Status	122

F		Page
FED	Find Edge	137
FNL	Fast Reference Move To Negative Limit	138
FPL	Fast Reference Move To Positive Limit	138
FRF	Fast Reference Move To Reference Switch	139
H		**Page**
HIA	Configure Control Done By HID Axis	143
HIN	Set Activation State For HID Control	145
HLT	Halt Motion Smoothly	148
M		**Page**
MAC	Call Macro Function	155
MOV	Set Target Position	160
MVR	Set Target Relative To Current Position	161
O		**Page**
OSM	Open-Loop Step Moving	162
OSN?	Read Number Steps	162
S		**Page**
STE	Start Step And Response Measurement	172
STP	Stop All Axes	172

Parameters

4.9.9 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

Depending on the drive mode selected via the parameter **PIShift Closed-Loop Driving Mode** (0x1F0007A3 (p. 228)), the E-873.1AT determines the on-target state based on the following criteria:

- For Q-Motion® modes: Settling window around the target position (Parameter **Settling Window (Encoder Counts)** (0x36))
- For PiezoMike mode: Entrance and exit window around the target position (parameters **PiezoMike Enter Window (Phys. Unit)** (0x1F0007A4) and **PiezoMike Exit Window (Phys. Unit)** (0x1F0007A5))
- For all drive modes: Delay time for setting the on-target state (Parameter **Settling Time (s)** (0x3F))

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

- The current position is around the settling/entrance window and remains there for a minimum of the duration of the delay time.

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

Commands

#		Page
#4	Request Status Register	121
O		Page
ONT?	Get On-Target State	161
S		Page
SRG?	Query Status Register Value	170

Parameters

0x36	Settling Window (Encoder Counts)	Settling window around the target position. Presets the window limits (half of the window width). If the current position enters the settling window, the target position is considered as reached. If the current position exits the settling window, the target position is no longer considered as reached. Can be changed only if servo mode is switched off. 0 to maximum value. The maximum value depends on the controller but may not exceed 231 encoder counts.
0x1F0007A4	PiezoMike Enter Window (Phys. Unit)	Entrance window around the target position for motion in closed-loop operation in PiezoMike mode. Commanded movement will be continued until the entrance window around the target position has been reached. The target position is then considered as reached and the on-target signal is output, if necessary, after a delay time (parameter 0x3F). This parameter will only be evaluated if 0x1F0007A3 is set to 1 (PiezoMike mode).
0x1F0007A5	PiezoMike Exit Window (Phys. Unit)	Exit window around the target position for motion in closed-loop operation in PiezoMike mode. Defines a tolerance range around the entrance window surrounding the target position (parameter 0x1F0007A4) to maintain the on-target signal after a movement. If the current position leaves the exit window, the on-target signal will no longer be output and the system will try to reach the target position once again. This parameter will only be evaluated if 0x1F0007A3 is set to 1 (PiezoMike mode).
0x3F	Settling Time (s)	Delay time for setting the on-target state. Recommended values: 0.000 to 1.000 seconds

4.9.10 Reference Switch Detection

The E-873.1AT receives signals from the reference switch at its axis connector (p. 241). The TRS? command can be used to determine whether an axis has a reference switch with direction sensing.

The signal from the reference switch of the positioner can be used for reference moves. In the case of a positioner with incremental position sensor, the controller knows the absolute axis position after a reference move (Referencing (p. 43)).

The following parameters can be used to configure how the E-873.1AT detects the reference switch:

- Invert Reference? (0x31)
- Has Reference? (0x14)
- Reference Signal Type (0x70)

Commands

T		Page
TRS?	Indicate Reference Switch	175

Parameters

0x31	Invert Reference?	Should the reference signal be inverted? Inverts the signal of the reference switch or a digital input that is used instead of the reference switch. 0 Reference signal not inverted 1 Reference signal inverted
0x14	Has Reference?	Do the mechanics have a reference switch? Activates respectively deactivates reference moves to the installed reference switch. 0 No reference switch 1 Reference switch available
0x70	Reference Signal Type	Reference signal type.

4.9.11 Detecting Limit Switches

The E-873.1AT receives signals from limit switches at its axis connector (p. 241). The LIM? command can be used to determine whether an axis has limit switches.

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 (p. 42) can be set via parameters of the E-873.1AT.

The limit switch signals can also be used for reference moves. In the case of a positioner with incremental position sensor, the controller knows the absolute axis position after a reference move (Referencing (p. 43)).

The following parameters can be used to configure how the E-873.1AT detects the limit switches:

- Limit Mode (0x18)
- Has No Limit Switches? (0x32)
- Use Limit Switches Only For Reference Moves? (0x77)

Commands

L		Page
LIM?	Indicate Limit Switches	154

Parameters

0x18	Limit Mode	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
0x32	Has No Limit Switches?	Do the mechanics not have limit switches? Activates motion stop at the installed limit switches. 0 Mechanics have limit switches 1 Mechanics do not have limit switches
0x77	Use Limit Switches Only For Reference Moves?	Should the limit switches only be used for reference moves? Is intended for use with rotation stages. Only evaluated when parameter 0x32 has the value 0.

4.9.12 Travel Range and Soft Limits

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

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

Settings for the travel range

The following parameters of the E-873.1AT reflect the physical travel range of the positioner:

- Value At Reference Position (Phys. Unit) (0x16)
- Distance From Negative Limit To Reference Position (Phys. Unit) (0x17)
- Distance From Reference Position To Positive Limit (Phys. Unit) (0x2F)

Settings for Soft Limits

The E-873.1AT determines the soft limits using the Maximum Travel In Positive Direction (Phys. Unit) parameter (0x15) and Maximum Travel In Negative Direction (Phys. Unit) parameter (0x30):

- **Maximum Travel In Positive Direction (Phys. Unit)** (0x15) and **Maximum Travel In Negative Direction (Phys. Unit)** (0x30):
  - The limits determine the limits for the permissible travel range.
  - 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 is selected from the positioner database.

Commands

Parameters

0x16	Value At Reference Position (Phys. Unit)	Position value at the reference switch. The current position is set to this value if the axis has performed a reference move to thereference switch. The parameter value is also used for calculating the position values set after reference moves to the limit switches; this also applies when the mechanics do not have a reference switch.
0x17	Distance From Negative Limit To Reference Position (Phys. Unit)	Gap between reference switch and negative travel range limit. The current position is set to the difference between the values of parameters 0x16 and 0x17 if the axis has done a reference move to the negative travel range limit.
0x2F	Distance From Reference Position To Positive Limit (Phys. Unit)	Gap between the reference switch and the positive travel range limit. If the axis has done a reference move to the positive travel range limit, the current position is set to the sum of the values of parameters 0x16 and 0x2F.
0x15	Maximum Travel In Positive Direction (Phys. Unit)	Soft limit in positive direction, in relation to the zero position. The positive travel range limit cannot be used for reference moves if this value is smaller than the position value for the positive travel range limit (that results from the sum of parameters 0x16 and 0x2F). The value can be negative.
0x30	Maximum Travel In Negative Direction (Phys. Unit)	Soft limit in a negative direction, in relation to the zero position. The negative travel range limit cannot be used for reference moves if this value is greater than the position value for the negative travel range limit (that results from the difference between parameters 0x16 and 0x17). The value can be negative.

4.9.13 Referencing

Whether referencing is necessary for the axis depends on the signal type of the position sensor:

- Absolute measuring position sensor: Referencing is not necessary.
- Incremental position sensor: Referencing is necessary.

Incremental sensors only supply relative motion information. When the positioner is equipped with an incremental position sensor, the controller does not know the absolute position of the axis when it is switched on or rebooted. Referencing must be done beforehand so that absolute target positions can be commanded and reached.

Referencing options

Referencing can be done in different ways:

- **Reference move** (default): A reference move moves the axis to a specifically defined point, e.g., to the reference switch, to a limit switch or to a hard stop. At this point, the current position is set to a defined value. The controller now knows the absolute axis position. The reference move can be started with one of the following commands:
  - FRF: Starts a reference move to the reference switch. The approach depends on the value of the Reference Signal Type parameter (ID 0x70).
  - FNL: Starts a reference move to the negative limit switch
  - FPL: Starts a reference move to the positive limit switch
- **Setting the absolute position manually**: If referencing method was activated by the RON command, you can set the current position of the axis to an arbitrary value at an arbitrary point using the POS command. The axis is not moved here. The controller knows the absolute axis position afterwards.

FRF? can be used to query whether an axis is referenced. RON? can be used to query the referencing method for an axis.

The Use Hard Stops For Referencing? parameter (0x7A) sets whether hard stops may be used for reference moves. The set parameter value can be queried with the HAR? command.

Referencing is done by a reference move by default when the PIMikroMove is used for starting and operating. Knowledge of the commands and parameters described here is not needed for referencing using PIMikroMove.

Reference Move Procedure

To achieve maximum repeatability when referencing, each reference move comprises the following steps:

1. First move to the selected switch.
   The maximum velocity is specified by the Closed-Loop Velocity (Phys. Unit/s) parameter (0x49) and the default direction for the reference move by the Reference Travel Direction parameter (0x47).
2. Stop when the switch edge has been reached.
   The higher the velocity on approach, the farther the axis overruns the edge of the switch (overshooting).
3. Move in the opposite direction to compensate for overshoot.
4. Second move to the selected switch.
   The maximum velocity is specified by the Velocity For Reference Moves (Phys. Unit/s) parameter (0x50).
5. Stop on reaching the switch edge.
6. Move in the opposite direction to compensate for overshoot.
7. Set the current position to a defined value.
   Referencing is finished.

The lower the velocity is when approaching the switch, the less the overshoot will be, and the higher the repeatability. Therefore, the maximum value of parameter 0x50 should be as large as the value of parameter 0x49, though ideally substantially less.

The actual velocities during the reference move are calculated from the values of the following parameters and can be lower than the maximum values:

- **Closed-Loop Velocity (Phys. Unit/s)** parameter (0x49) or **Velocity For Reference Moves (Phys. Unit/s)** parameter (0x50)
- Distance Between Limit And Hard Stop (Phys. Unit) parameter (0x63)
- Closed-Loop Deceleration (Phys. Unit/s2) parameter (0xC)

The Distance From Limit To Start Of Ref. Search (Phys. Unit) parameter (0x78) and Distance For Reference Search (Phys. Unit) parameter (0x79) are used for reference moves when both of the following conditions are fulfilled:

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

Reference move sequence:

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

Commands

F		Page
FNL	Fast Reference Move To Negative Limit	138
FPL	Fast Reference Move To Positive Limit	138
FRF	Fast Reference Move To Reference Switch	139
FRF?	Get Referencing Result	139
H		Page
HAR?	Indicate Hard Stops	140
P		Page
POS	Set Real Position	163
R		Page
RON	Set Reference Mode	164
RON?	Get Reference Mode	165

Parameters

0x70	Reference Signal Type	Reference signal type.
0x7A	Use Hard Stops For Referencing?	Should the hard stops be used for reference moves?
0x49	Closed-Loop Velocity (Phys. Unit/s)	Velocity in closed-loop operation with dynamics profile. 0 to value of 0xA [phys. units / s]
0x47	Reference Travel Direction	Default direction for the reference move. 0 Automatic detection 1 Negative direction 2 Positive direction
0x50	Velocity For Reference Moves (Phys. Unit/s)	Maximum velocity for reference moves. 0 to value of value 0x49 [phys. units / s]
0x63	Distance Between Limit And Hard Stop (Phys. Unit)	Gap between internal limit switch and hard stop. Determines the maximum stopping distance during reference moves. The actual velocities during a reference move are calculated on the basis of this value, the set deceleration (0xC) and set velocities (0x49 and 0x50).
0xC	Closed-Loop Deceleration (Phys. Unit/s^2)	Deceleration with dynamics profile in closed-loop operation. 0 to value of 0x4B [phys. units. / s2]

0x78	Distance From Limit To Start Of Ref Search (Phys. Unit)	Distance between the limit switch and the starting position for the reference move to the index pulse. Used for FRF when parameter 0x70 has the value 2.
0x79	Distance For Reference Search (Phys. Unit)	Maximum distance for motion to the index pulse.
0x70	Reference Signal Type	Reference signal type.

5 Unpacking / Transportation

5.1 Unpacking

Unpacking the E-873.1AT

1. Unpack the E-873.1AT with care.
2. If the E-873.1AT was supplied with ESD protective caps on the connectors: Do **not** remove the ESD protective caps.
3. Compare the contents with the scope of delivery according to the contract and the delivery note.
4. Inspect the contents for signs of damage. If any parts are damaged or missing, contact our customer service department (p. 10) immediately.
5. Keep all packaging materials in case the product needs to be returned.

5.2 Transportation

Pay attention to the ambient conditions and classifications (p. 239) when transporting the E-873.1AT.

1. Pack the E-873.1AT in the original packaging.
2. If the E-873.1AT is to be sent, use a stable outer box.

6 Installation

6.1 Mounting the E-873.1AT

The E-873.1AT can be used as a benchtop device or mounted on a surface in any orientation or installed in a control cabinet.

Overview

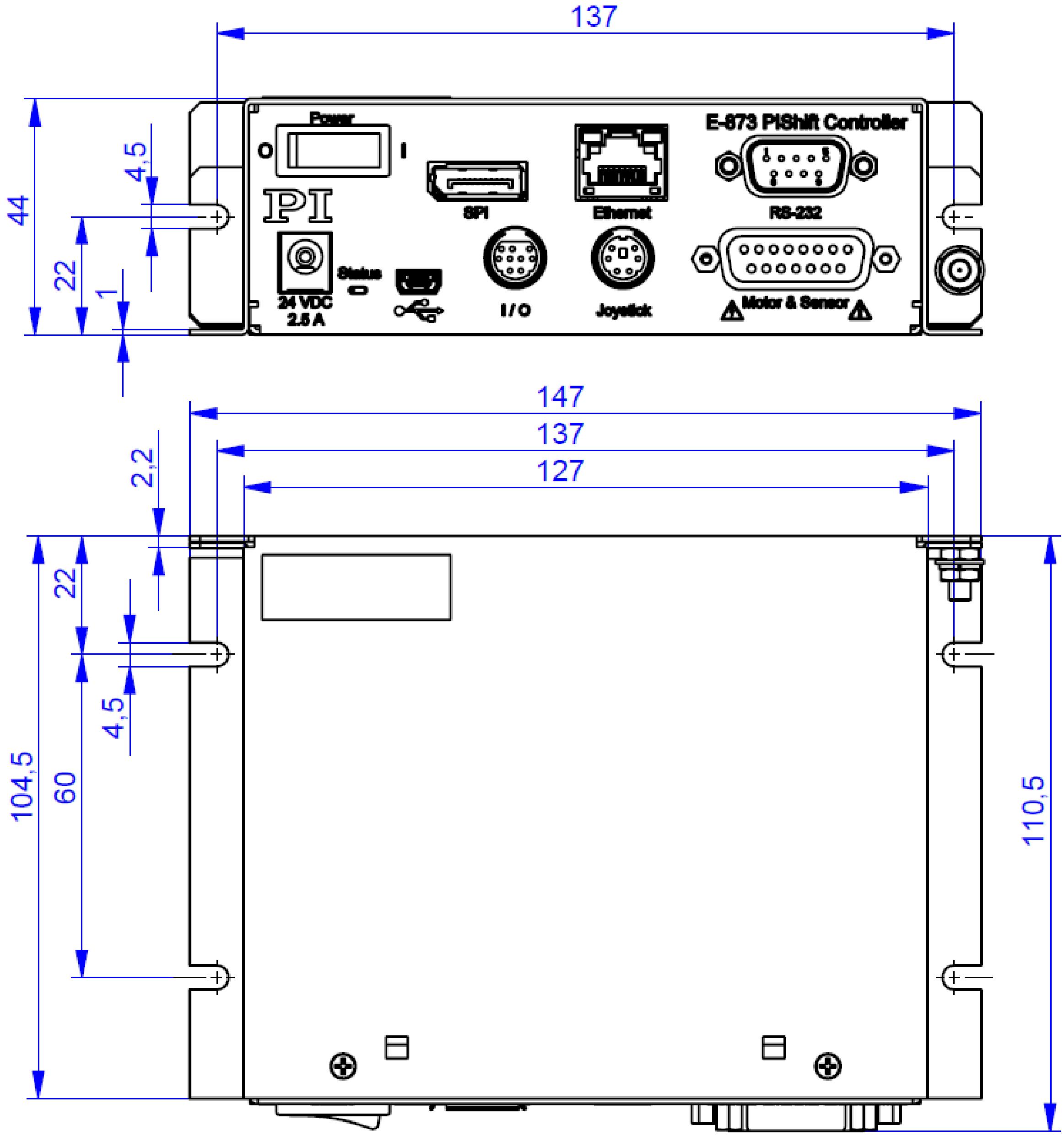


Figure 10: E-873.1AT: Dimensions and position of the recesses for mounting

Tools and Accessories

- Suitable screws
- Suitable screwdriver

Requirements

✓ You have read and understood the general safety instructions (p. 12).

NOTICE

Heating up of the E-873.1AT during operation!

High temperatures can overheat the E-873.1AT.

- ▶ Install the E-873.1AT with a gap of at least 10 cm to the top and rear panels and at least 5 cm to its sides. If this is not possible, make sure that the surroundings are cooled sufficiently.
- ▶ Ensure sufficient ventilation at the place of installation.
- ▶ Keep the ambient temperature at a noncritical level.

Mounting the E-873.1AT on a Surface

1. Drill the holes required into the surface.
   The arrangement of the recesses in the mounting rails of the E-873.1AT can be seen in the figure.
2. Tighten the E-873.1AT to the recesses provided with suitable screws.

6.2 Connecting the E-873.1AT to the Protective Earth Conductor

The E-873.1AT is not grounded via the voltage connection and must therefore be connected to the protective earth conductor. The protective earth connector is on the right-hand rear mounting rail of the E-873.1AT.

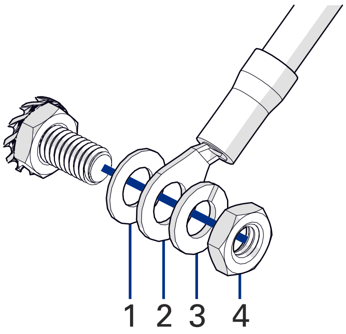


Figure 11: Connecting the protective earth conductor to the threaded bolt

1. Flat washer
2. Cable lug of the protective earth conductor
3. Spring washer
4. Nut

Tools and Accessories

- Suitable protective earth conductor with cable lug:
  - Cable cross section ≥ 0.75 mm²
  - Contact resistance < 0.1 ohm at 25 A at all connection points relevant for attaching the protective earth conductor
- Suitable wrench

Requirements

✓ You have read and understood the general safety instructions (p. 12).

✓ The E-873.1AT is **not** connected to the power supply.

Information

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

Connecting the E-873.1AT to the Protective Earth Conductor

1. If necessary, attach a suitable cable lug to the protective earth conductor.
2. Remove the outer nut and the spring washer from the threaded bolt of the protective earth connector.
3. Push the cable lug of the protective earth conductor and the spring washer onto the threaded bolt.
4. Screw the nut onto the threaded bolt again.

➔ *The cable lug of the protective earth conductor is clamped between the flat washer and the spring washer.*

5. Tighten the nut with at least three turns and a torque of 1.2 Nm to 1.5 Nm.

6.3 Connecting the Power Supply to the E-873.1AT

Tools and Accessories

- Included power adapter (alternative: Sufficiently rated power adapter)
- Included power cord (alternative: Sufficiently rated power cord)

Requirements

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

✓ The E-873.1AT is installed near the power supply so that the power plug can be quickly and easily disconnected from the mains.

Connect the Power Adapter to the E-873.1AT

1. Connect the power adapter's barrel connector to the E-873.1AT's barrel connector socket (*24 VDC 2.5 A*).
2. Connect the power cord to the power adapter.

6.4 Connecting the Positioner to the E-873.1AT

Tools and Accessories

- Compatible positioner with Q-Motion® or PiezoMike piezo inertia drive
- If necessary: Suitable adapter cable from PI, available as optional accessory for the positioner
- If the distance between the E-873.1AT and the positioner is too long: Suitable drive or extension cable from PI, available as optional accessory for the positioner

Requirements

✓ The power adapter is **not** connected to the power socket via the power cord or the E-873.1AT is switched off.

✓ You have read and understood the user manual for the positioner to be connected.

✓ You have installed the positioner to be connected according to the instructions in its respective user manual.

NOTICE

Damage if a wrong drive type is connected!

Connecting a positioner with incompatible drive type can cause irreparable damage.

► Connect the E-873.1AT only to positioners with Q-Motion® or PiezoMike piezo inertia drive.

Connecting the Stage to the E-873.1AT

1. If necessary: Connect a suitable adapter to the positioner's drive connector.
2. Connect the positioner's drive connector or the adapter to the E-873.1AT's ***Motor & Sensor*** socket.
3. Use the integrated screws to secure the connections against accidental disconnection.

6.5 Installing the PC Software

6.5.1 Installing the PC Software for the First Time

Tools and Accessories

- PC with Windows or Linux (p. 17) operating system and at least 30 MB free storage space
- Software from PI: On the storage device supplied or available for download from our website www.pi.de.
- Optional for custom positioners: Data storage device or archive file with the following content:
  - *Import PI CustomStage* program
  - Custom positioner database with the parameter set for the positioner

Installing the PC software in Windows

1. Insert the data storage device into the PC or go to the directory where you saved the downloaded software to.
2. Start the install wizard by double-clicking **PISoftwareSuite.exe**.

➔ *The* ***InstallShield Wizard*** *window opens for installing the PC software from PI.*

3. Follow the instructions on the screen.
   The PI software suite includes the following components:
   - Driver for use with NI LabVIEW software
   - Dynamic program library for GCS
   - PIMikroMove
   - PC software for updating the firmware of the E-873.1AT
   - PIUpdateFinder for updating the PC software
   - USB driver

Installing a Custom Positioner Database in Windows

If you have a **custom positioner database**, this must also be installed on the PC.

1. Insert the data storage device into the PC or go to the directory where you saved the downloaded software to.
2. Run the installation assistant for the custom positioner database by clicking *Import_PI_CustomStage.exe*.

➔ *The* ***Import PI Custom Stage*** *program is run and the parameter set is imported from the custom positioner database into PIStages3.*

3. If a message appears that installation of the custom positioner database failed:
   a) Update the PIStages3 positioner database on your PC (p. 52).

b) Repeat the installation of the custom positioner database.

Installing the PC Software in Linux

1. Insert the data storage device into the PC or go to the directory where you saved the downloaded software to.
2. Unzip the tar archive from the **/linux** directory into a directory on your PC.
3. Open a terminal and go to the directory where you unpacked the tar archive.
4. Log in as superuser (root privileges).
5. Enter ./INSTALL to start the installation.
   Pay attention to lower and upper case when entering commands.
6. Follow the instructions on the screen.
7. If you have received a **custom positioner database**: Copy the positioner database file into the following directory: /usr/local/PI/pi_gcs_translator/

6.5.2 Updating the PC Software

PI is constantly improving the PC software. Always install the latest version of the PC software and the positioner database.

PIUpdateFinder is a program which helps you find updates for your PI software on the PI server. It identifies the PI software installed on your computer and compares it with the software available on the PI server. This comparison is made using the Internet. If newer software versions are available on the PI server, you can download the software via a corresponding link.

Confidential customer data is not transmitted when comparing the software versions. The following information is transmitted:

- Software component and version
- Internet browser
- Operating system
- IP address

Updating the PC Software and Positioner Database in Windows

Requirements

✓ You have installed the PIUpdateFinder on the PC to be updated.

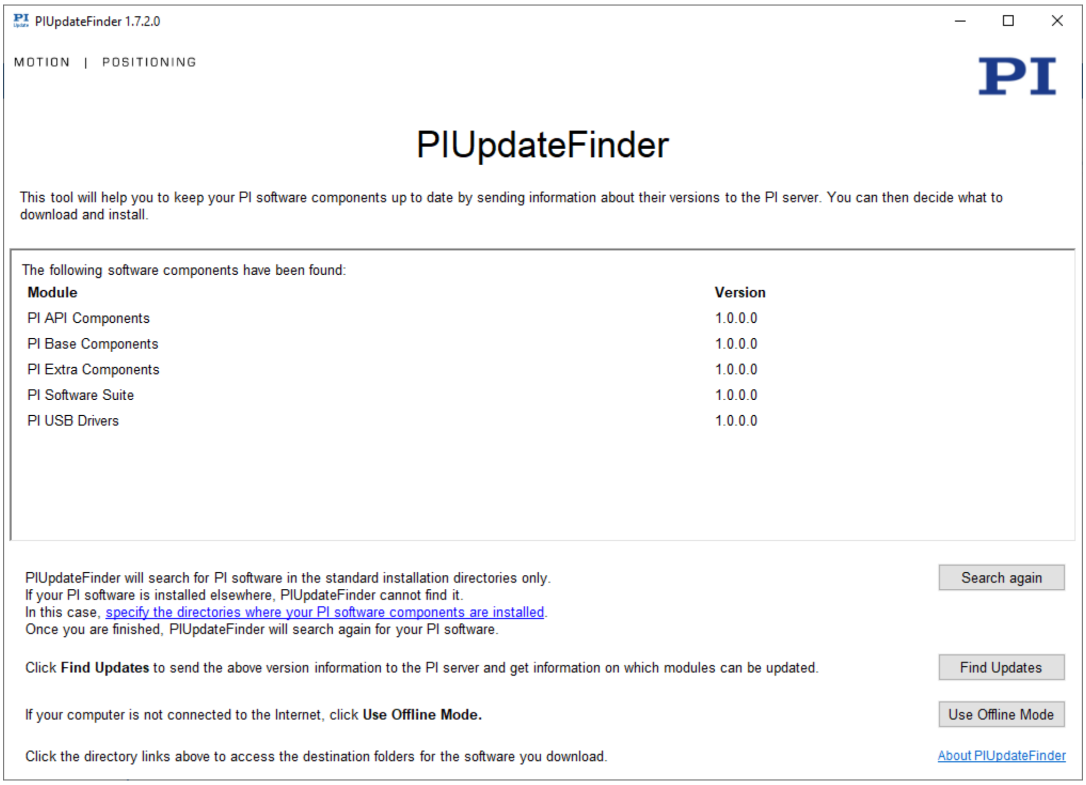
✓ In addition when updating the software on a PC without Internet connection:

  - PC with Internet connection
  - Portable data storage device, e.g., USB stick, for the data exchange between the computers

Updating the Software on a PC with Internet Connection

1. Start PIUpdateFinder on the PC to be updated.

➔ *A table is displayed with information on the PI software installed on your computer.*



PIUpdateFinder

This tool will help you to keep your PI software components up to date by sending information about their versions to the PI server. You can then decide what to download and install.

The following software components have been found:

Module	Version
PI API Components	1.0.0.0
PI Base Components	1.0.0.0
PI Extra Components	1.0.0.0
PI Software Suite	1.0.0.0
PI USB Drivers	1.0.0.0

PIUpdateFinder will search for PI software in the standard installation directories only.
If your PI software is installed elsewhere, PIUpdateFinder cannot find it.
In this case, specify the directories where your PI software components are installed.
Once you are finished, PIUpdateFinder will search again for your PI software.

Click **Find Updates** to send the above version information to the PI server and get information on which modules can be updated.

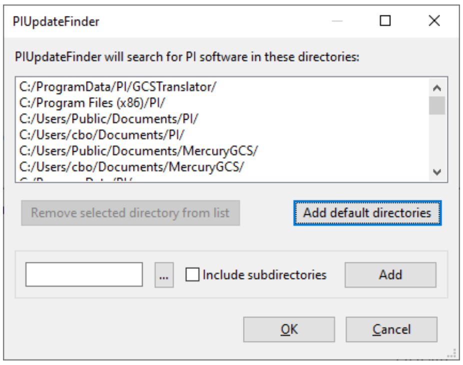
If your computer is not connected to the Internet, click **Use Offline Mode.**

Click the directory links above to access the destination folders for the software you download.

If the table is empty or incomplete, proceed as follows:

a) Click ***specify the directories where your PI software components are installed***.

➔ *A dialog window is opened, in which all directories are listed that PIUpdateFinder finds while searching for and comparing the software versions.*



PIUpdateFinder will search for PI software in these directories:

C:/ProgramData/PI/GCSTranslator/
C:/Program Files (x86)/PI/
C:/Users/Public/Documents/PI/
C:/Users/cbo/Documents/PI/
C:/Users/Public/Documents/MercuryGCS/
C:/Users/cbo/Documents/MercuryGCS/

b) Click the ***...*** button and select the directory on your hard disk, where the PI software is installed.

➔ *The directory is displayed in the input field at the bottom left of the window. You can also manually enter directories there.*

c) Activate the ***Include subdirectories*** checkbox to include subdirectories as well.

d) Click the ***Add*** button.

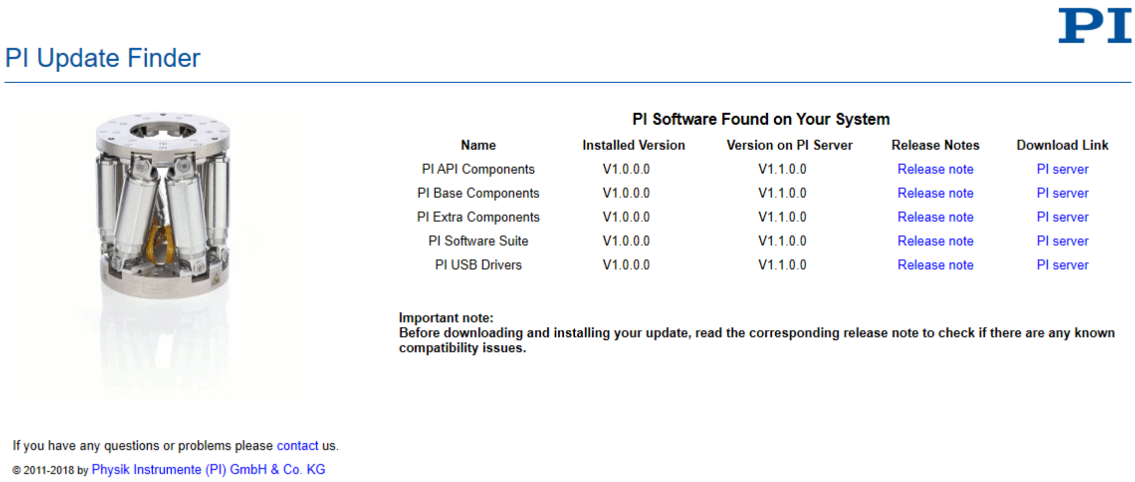
➔ *The directory, and any subdirectories, appear at the end of the list.*

e) Click ***OK*** to finalize input of the installation directory.

➔ *If PI software is found in the specified directories, it will be displayed in the table on the initial screen of the PIUpdateFinder.*

2. Click the ***Find Updates*** button.

➔ *A browser window opens and a table is displayed with the software information. Updates are available from the column* ***Download Link*** *via the* ***PI server*** *link.*



PI Update Finder

PI Software Found on Your System

Name	Installed Version	Version on PI Server	Release Notes	Download Link
PI API Components	V1.0.0.0	V1.1.0.0	Release note	PI server
PI Base Components	V1.0.0.0	V1.1.0.0	Release note	PI server
PI Extra Components	V1.0.0.0	V1.1.0.0	Release note	PI server
PI Software Suite	V1.0.0.0	V1.1.0.0	Release note	PI server
PI USB Drivers	V1.0.0.0	V1.1.0.0	Release note	PI server

Important note:
Before downloading and installing your update, read the corresponding release note to check if there are any known compatibility issues.

If you have any questions or problems please contact us.

© 2011-2018 by Physik Instrumente (PI) GmbH & Co. KG

3. To download an update, click *PI server* in the *Download Link* column.

➔ *You are then asked to specify whether the file is to be downloaded or run directly. Depending on the browser settings, it is possible that the update file is downloaded directly. It is then in the download directory of your computer.*

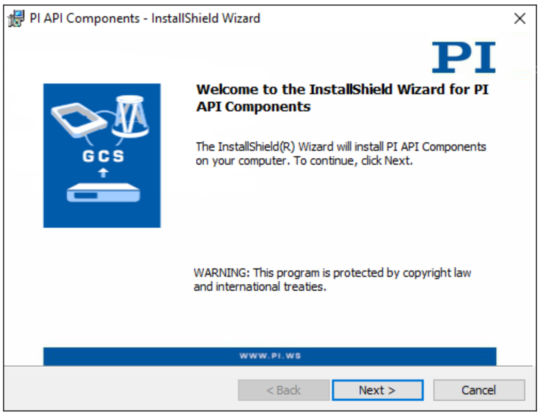
4. Activate the option in the browser query to save the update file to the your computer's hard disk.

➔ *The file is saved to the download directory of your computer. Depending on the browser settings, it is also possible to run the file directly or open it. In this case, continue with step 6.*

5. Install the update:
   - Executable setup files (.exe): Run the downloaded update file.
   - Data archive (.zip): Unpack the archive to a directory on your PC. Open this directory and run the desired setup file (.exe).

   You may have to confirm running the file.

➔ *This opens the InstallShield Wizard, e.g., for PI API Components:*



PI API Components - InstallShield Wizard

Welcome to the InstallShield Wizard for PI API Components

The InstallShield(R) Wizard will install PI API Components on your computer. To continue, click Next.

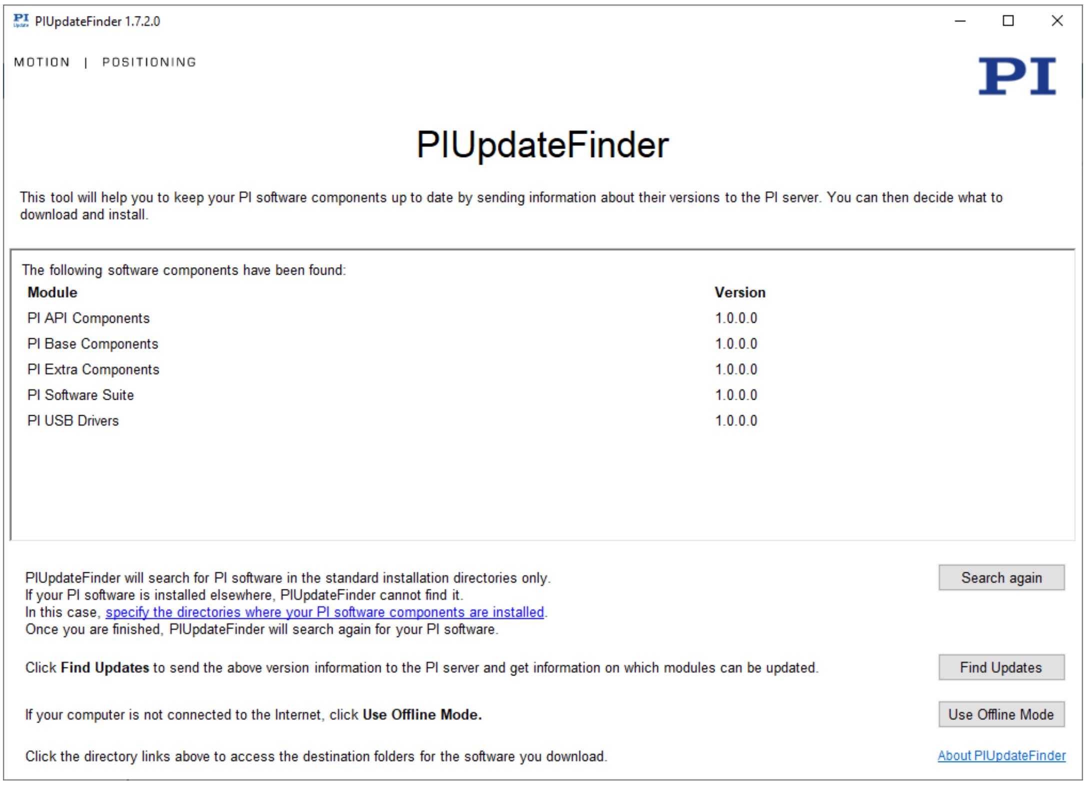
WARNING: This program is protected by copyright law and international treaties.

6. Follow the instructions in the InstallShield Wizard.

Updating the Software on a PC without Internet Connection

1. Connect the portable data storage device, e.g., an USB stick, with the PC to be updated.
2. Start PIUpdateFinder on the PC to be updated.

➔ *A table is displayed with information on the PI software installed on your computer.*



PIUpdateFinder

This tool will help you to keep your PI software components up to date by sending information about their versions to the PI server. You can then decide what to download and install.

The following software components have been found:

Module	Version
PI API Components	1.0.0.0
PI Base Components	1.0.0.0
PI Extra Components	1.0.0.0
PI Software Suite	1.0.0.0
PI USB Drivers	1.0.0.0

PIUpdateFinder will search for PI software in the standard installation directories only.
If your PI software is installed elsewhere, PIUpdateFinder cannot find it.
In this case, specify the directories where your PI software components are installed.
Once you are finished, PIUpdateFinder will search again for your PI software.

Click **Find Updates** to send the above version information to the PI server and get information on which modules can be updated.

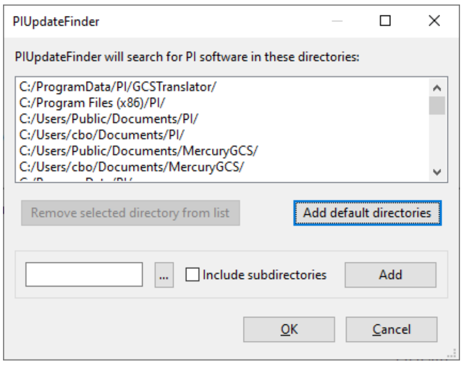
If your computer is not connected to the Internet, click **Use Offline Mode.**

Click the directory links above to access the destination folders for the software you download.

If the table is empty or incomplete, proceed as follows:

a) Click ***specify the directories where your PI software components are installed***.

➔ *A dialog window is opened, in which all directories are listed that PIUpdateFinder finds while searching for and comparing the software versions.*



PIUpdateFinder will search for PI software in these directories:

C:/ProgramData/PI/GCSTranslator/
C:/Program Files (x86)/PI/
C:/Users/Public/Documents/PI/
C:/Users/cbo/Documents/PI/
C:/Users/Public/Documents/MercuryGCS/
C:/Users/cbo/Documents/MercuryGCS/

b) Click the ***...*** button and select the directory on your hard disk, where the PI software is installed.

➔ *The directory is displayed in the input field at the bottom left of the window. You can also manually enter directories there.*

c) Activate the ***Include subdirectories*** checkbox to include subdirectories as well.

d) Click the ***Add*** button.

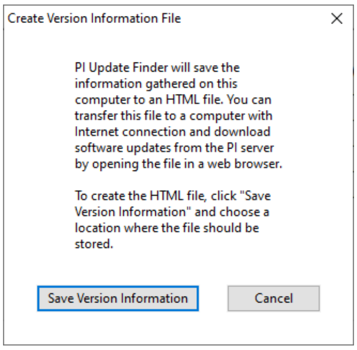
➔ *The directory, and any subdirectories, appear at the end of the list.*

e) Click ***OK*** to finalize input of the installation directory.

➔ *If PI software is found in the specified directories, it will be displayed in the table on the initial screen of the PIUpdateFinder.*

3. In the main window, click the ***Use Offline Mode*** button.

➔ *The window **Create Version Information File** will open.*



Create Version Information File

PI Update Finder will save the information gathered on this computer to an HTML file. You can transfer this file to a computer with Internet connection and download software updates from the PI server by opening the file in a web browser.

To create the HTML file, click "Save Version Information" and choose a location where the file should be stored.

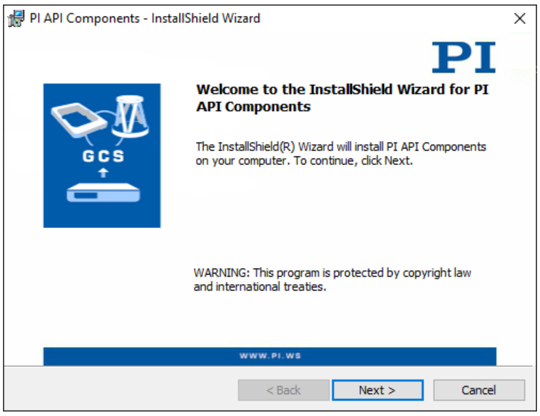
4. Click the ***Save Version Information*** button to save the software version information to a HTML file on your portable data storage device.
5. Transfer the HTML file to a computer connected to the Internet.
6. Open the HTML file in a web browser on the computer connected to the Internet.

➔ *A browser window opens and a table is displayed with the software information. Updates are available from the column **Download Link** via the **PI server** link.*

7. Download your updates and save them to your portable data storage device.
8. Transfer the updates to the PC without Internet connection.
9. Install the updates.
   - Executable setup files (.exe): Run the downloaded update file.
   - Data archive (.zip): Unpack the archive to a directory on your PC. Open this directory and run the desired setup file (.exe).

   You may have to confirm running the file.

➔ *This opens the InstallShield Wizard, e.g., for PI API Components:*



PI API Components - InstallShield Wizard

Welcome to the InstallShield Wizard for PI API Components

The InstallShield(R) Wizard will install PI API Components on your computer. To continue, click Next.

WARNING: This program is protected by copyright law and international treaties.

10. Follow the instructions in the InstallShield Wizard.

Updating the PC Software and Positioner Database in Linux

Requirements

✓ Active connection to the Internet.

Updating the PC software in Linux

1. Open the website www.physikinstrumente.com/en/products/software-suite
2. Scroll down to ***Downloads***.
3. Click ***ADD TO LIST*** for the ***PI Software Suite C-990.CD1***.
4. Click ***REQUEST***.
5. Fill out the download request form and send the request.

➔ *The download link will then be sent to the e-mail address entered.*

6. Via the link download the archive file with the ***PI Software Suite***.
7. Unpack the archive file on your PC to a separate directory.
8. Go to the ***linux*** subdirectory in the directory with the unpacked files.
   You will find an additional archive file there.
9. Unpack the archive file in the ***linux*** directory by entering the command tar -xvpf <name of the archive file> on the console.
10. Log into the PC as superuser (root privileges).
11. Install the update.
12. If you also received an updated PISTAGES3.DB database from our customer service department: Install that update onto the PC.

6.6 Connecting the PC

Communication between the E-873.1AT and a PC is required to configure the E-873.1AT and to command motion using the GCS commands.

Information

The E-873.1AT's communication interfaces are active at the same time. Commands are executed in the order in which the complete command lines arrive. However, simultaneous use of several communication interfaces can cause problems with the PC software.

► Always only use one interface of the E-873.1AT.

6.6.1 Connecting the E-873.1AT to a PC

Tools and Accessories

- Suitable cable for the selected communication interface (in the scope of delivery (p. 15) or available as optional accessory (p. 15))

Requirements

✓ The PC has a connection available for the selected communication interface.

✓ If necessary: The interface parameters of the PC have been adapted for the E-873.1AT (p. 16).

Connecting the E-873.1AT to the PC

1. Connect the cable to the selected communication interface of the E-873.1AT.
2. Connect the cable to an unused port on the PC.

6.6.2 Integrating the E-873.1AT into a Network

Tools and Accessories

- Suitable network cable (in the scope of delivery (p. 15) or available as optional accessory (p. 15))

Requirements

✓ A network access point is available for the E-873.1AT.

✓ The default settings of the TCP/IP interface (p. 17) do not collide with the settings of other devices in the network.

Integrating the E-873.1AT into a TCP/IP Network

1. Connect the RJ45 socket of the E-873.1AT to the network cable.
2. Connect the network cable to the network access point.

7 Startup and Operation

7.1 Switching on the E-873.1AT

Requirements

✓ You have read and understood the general safety instructions (p. 12).

✓ The E-873.1AT has been installed properly (p. 48).

CAUTION

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

If the protective earth conductor is missing or not properly connected, dangerous touch voltages can occur on the E-873.1AT in the event of malfunction or failure of the system. If there are touch voltages, touching the E-873.1AT can lead to minor injury due to electric shock.

- ► Connect the E-873.1AT to a protective earth conductor (p. 49) before startup.
- ► Do **not** remove the protective earth conductor during operation.
- ► If the protective earth conductor has to be removed temporarily (e.g., for modification), reconnect the E-873.1AT to the protective earth conductor before restarting.

Switching the E-873.1AT On

1. Connect the power cord of the power adapter to the power socket.
2. Move the toggle switch on the front panel to the — position to switch the E-873.1AT on.

➔ *The E-873.1AT boots the firmware and loads information from the nonvolatile memory to the volatile memory.*

3. Wait until the **Status** LED lights up green.

➔ *The information has loaded and the E-873.1AT is ready for normal operation.*

4. If the **Status** LED does not light up a few seconds after switching on, contact our customer service department (p. 10).

7.2 Establishing Communication with the PC

The procedure for PIMikroMove is described in the following.

The figures show the procedure for any electronics; the procedure for the E-873.1AT corresponds.

7.2.1 Establishing Communication via RS-232

To establish communication successfully the PC's and E-873.1AT's baud rates (p. 17) must match.

Requirements

✓ The E-873.1AT is connected to the RS-232 interface of the PC (p. 57).

✓ The E-873.1AT is switched on (p. 59).

✓ The PC is switched on.

✓ The required software and drivers are installed (p. 51) on the PC.

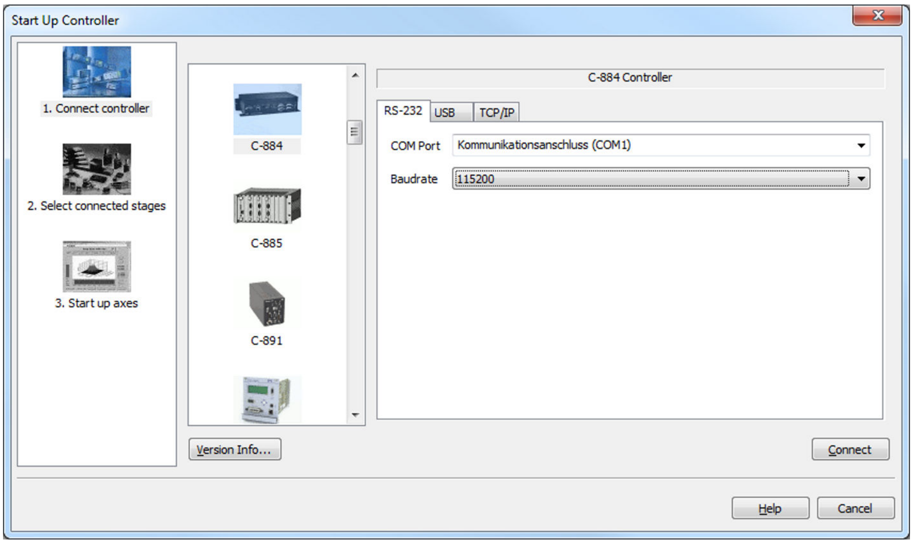
✓ You have read and understood the manual for the PC software. The links to the software manuals are in the A000T0081 file on the PI software data carrier.

Establishing Communication via RS-232

1. Start PIMikroMove.

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

2. If the ***Start up controller*** window does not automatically open, select the ***Connections > New...*** menu item in the main window.
3. Select ***E-873*** in the field for controller selection.
4. Select the ***RS-232*** tab on the right-hand side of the window.
5. Make the following settings in the ***RS-232*** tab:
   a) In the ***COM Port*** field, select the COM port of the PC to which you have connected the E-873.1AT.
   b) In the ***Baudrate*** field, set the value that is set for the E-873.1AT.
   ➔ *This adapts the baud rate of the PC to the baud rate of the E-873.1AT.*



COM Port Kommunikationsanschluss (COM1)

Baudrate 115200

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

➔ *If communication could not be established, look for a solution to the problem in the "Troubleshooting (p. 235)" chapter.*

➔ *If communication was established successfully, PIMikroMove guides you through the configuration of the E-873.1AT for the connected positioner, see "Starting Motion (p. 64)".*

7.2.2 Establishing Communication via USB

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

✓ The E-873.1AT is connected to the USB interface of the PC (p. 57).

✓ The E-873.1AT is switched on (p. 59).

✓ The PC is switched on.

✓ The required software and drivers are installed (p. 51) on the PC.

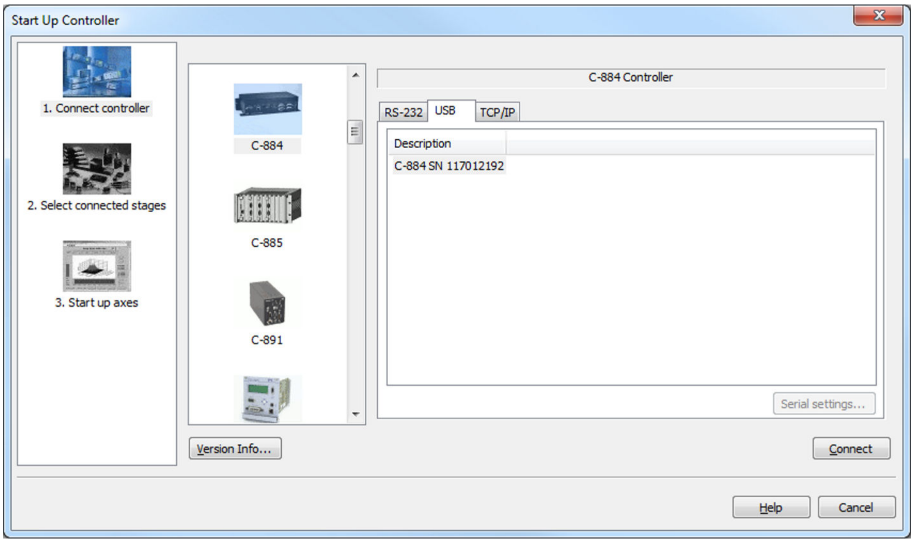
✓ You have read and understood the manual for the PC software. The links to the software manuals are in the A000T0081 file on the PI software data carrier.

Establishing Communication via USB

1. Start PIMikroMove.

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

2. If the *Start up controller* window does not open automatically, select the *Connections > New...* menu item in the main window.
3. Select *E-873* in the controller selection field.
4. Select the *USB* tab on the right-hand side of the window.



Description
C-884 SN 117012192

5. Select the connected E-873.1AT in the *USB* tab.
6. Click *Connect* to establish communication.

➔ *If communication could not be established, look for a solution to the problem in the "Troubleshooting (p. 235)" chapter.*

➔ *If communication was established successfully, PIMikroMove guides you through the configuration of the E-873.1AT for the connected positioner, refer to "Starting Motion (p. 64)".*

7.2.3 Establishing 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-873.1AT interface parameters is required. You can begin with setup of the communication (p. 63).
- Network without DHCP server or direct connection of the E-873.1AT to the PC's Ethernet socket: it is necessary to adapt the E-873.1AT's interface parameters (p. 62). Make the necessary adaptations before establishing communication.

Requirements

✓ The E-873.1AT is connected (p. 57) to the network or directly to the PC via the RJ45 Ethernet socket.

✓ If several E-873.1ATs are connected to the same network via their TCP/IP interfaces: You have the serial number of the E-873.1AT ready with which the communication is to be established. The serial number can be found on the type plate of the E-873.1AT (p. 14).

✓ The E-873.1AT is switched on (p. 59).

✓ The PC is switched on.

✓ The required software and drivers are installed (p. 51) on the PC.

✓ You have read and understood the manual for the PC software. The links to the software manuals are in the A000T0081 file on the PI software data carrier.

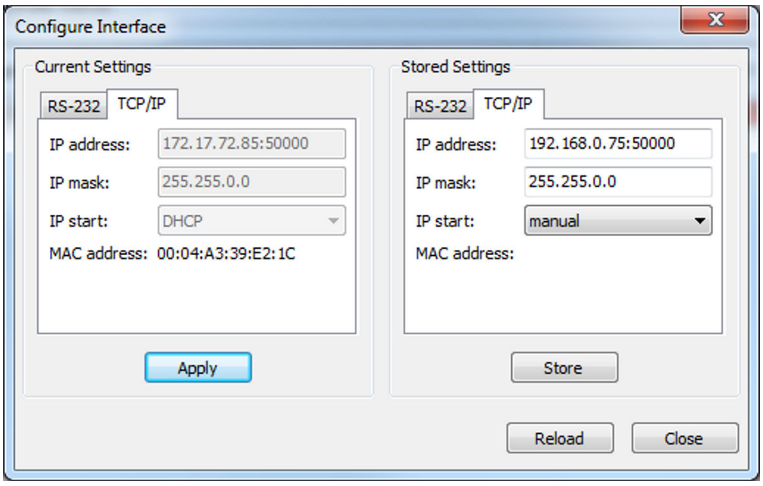
Adapting the TCP/IP Interface Parameters of the E-873.1AT

If you need to adapt the interface parameters of the E-873.1AT to use the E-873.1AT in a network, proceed as follows.

1. Establish communication between the E-873.1AT and the PC via a different interface (e.g., USB (p. 60)).
2. Select the ***E-873.1AT > Configure interface*** menu item in the main window of PIMikroMove.

➔ *The* ***Configure Interface*** *window opens.*

3. Select the ***TCP/IP*** tab in the ***Stored Settings*** area in the ***Configure Interface*** window.



Configure Interface

Current Settings
RS-232 | TCP/IP
IP address: 172.17.72.85:50000
IP mask: 255.255.0.0
IP start: DHCP
MAC address: 00:04:A3:39:E2:1C

Stored Settings
RS-232 | TCP/IP
IP address: 192.168.0.75:50000
IP mask: 255.255.0.0
IP start: manual
MAC address:

Figure 12: "Configure Interface" window with examples of settings

4. Make the necessary adaptations in the ***TCP/IP*** tab in the ***Stored Settings*** area:
   a) ***IP address*** field: E-873.1AT's IP address in format xxx.xxx.xxx.xxx:50000
   b) ***IP mask*** mask: Network's subnet mask
   c) ***IP start*** field: E-873.1AT's startup behavior

   manual: Manually specified, static IP address is used

   DHCP: IP address is assigned automatically by a DHCP server.

5. Save the changed settings to the nonvolatile memory of the E-873.1AT by clicking ***Store***.

➔ *The* ***Store interface settings*** *dialog opens.*

6. Click ***Store settings*** in the ***Store interface settings*** dialog.

➔ *The dialog closes. The settings were stored in the nonvolatile memory of the E-873.1AT .*

7. Close the ***Configure Interface*** window.
8. Close the connection with the E-873.1AT by selecting the ***Connections > Close > E-873.1AT*** menu item in the main window of PIMikroMove.
9. Switch the E-873.1AT off and on again via its toggle switch.

CAUTION

Risk of crushing from unexpected motion!

When the communication between the E-873.1AT and the PC is established via TCP/IP, the PC software offers all electronics for selection that are available in the same network. After selecting a E-873.1AT for the connection, all commands are transmitted to this device. If the wrong device is selected, unexpected motion could be commanded and result in bruising injuries to the operating and maintenance staff of the positioner connected.

► If several E-873.1AT entries are displayed in the PC software, make sure that you select the right E-873.1AT.

Information

Communication via TCP/IP can fail if the network cable was connected to the Ethernet socket of the E-873.1AT while the E-873.1AT was switched on.

► If communication cannot be established, switch the E-873.1AT off, connect the network cable, and switch the E-873.1AT on again.

Information

For communication via TCP/IP, the E-873.1AT only has one unchangeable port (50000) available that cannot be used for more than one connection at a time.

Establishing Communication via TCP/IP

1. Start PIMikroMove.

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

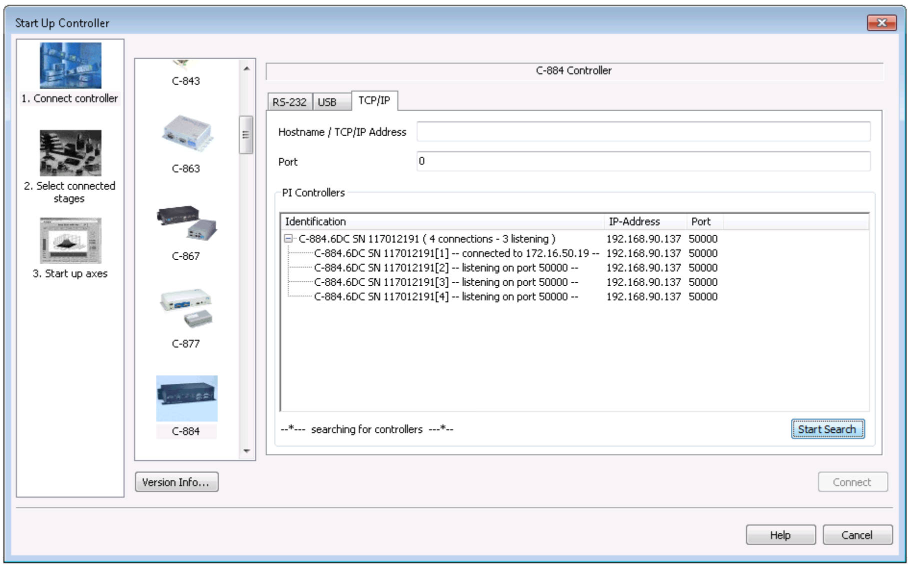
2. If the ***Start up controller*** window does not open automatically, select the ***Connections > New...*** menu item in the main window.
3. Select ***E-873*** in the controller selection field.
4. Select the *TCP/IP* tab on the right-hand side of the window.

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

Click ***Start Search*** if the search for E-873 type controllers does not start automatically.

➔ *Searching the network for the E-873 type of controller has started.*

*As long as the search is running, the **Connect** button is deactivated. If the search was successful, all E-873 controllers in the same network are displayed in the **PI Controllers** field.*



Hostname / TCP/IP Address

Port 0

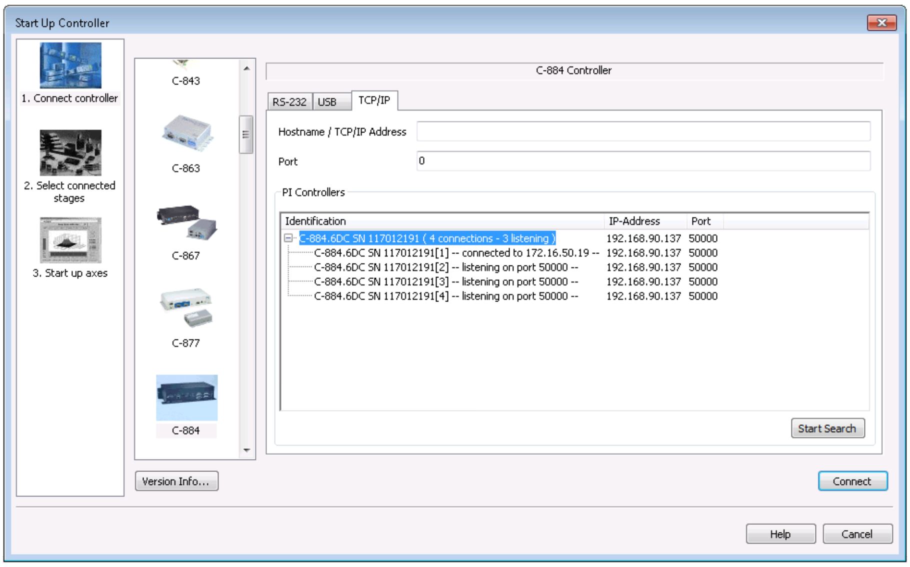
PI Controllers

Identification	IP-Address	Port
C-884.6DC SN 117012191 (4 connections - 3 listening)	192.168.90.137	50000
C-884.6DC SN 117012191[1] -- connected to 172.16.50.19 --	192.168.90.137	50000
C-884.6DC SN 117012191[2] -- listening on port 50000 --	192.168.90.137	50000
C-884.6DC SN 117012191[3] -- listening on port 50000 --	192.168.90.137	50000
C-884.6DC SN 117012191[4] -- listening on port 50000 --	192.168.90.137	50000

--*--- searching for controllers ---*--

5. Click the entry for your E-873.1AT found in the list of controllers. This must show the status "listening on port 50000".
   - 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 try to establish communication with this controller.
   - If several entries with the same name are shown, identify your E-873.1AT on the basis of its nine-digit serial number.
   - If the E-873.1AT is not displayed in the list of the controllers found, check the network settings. Consult your network administrator if necessary.

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



Hostname / TCP/IP Address

Port 0

Identification	IP-Address	Port
C-884.6DC SN 117012191 (4 connections - 3 listening)	192.168.90.137	50000
C-884.6DC SN 117012191[1] -- connected to 172.16.50.19 --	192.168.90.137	50000
C-884.6DC SN 117012191[2] -- listening on port 50000 --	192.168.90.137	50000
C-884.6DC SN 117012191[3] -- listening on port 50000 --	192.168.90.137	50000
C-884.6DC SN 117012191[4] -- listening on port 50000 --	192.168.90.137	50000

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

➔ *If communication could not be established, look for a solution to the problem in the "Troubleshooting (p. 235)" chapter.*

➔ *If communication was established successfully, PIMikroMove guides you through the configuration of the E-873.1AT for the connected positioner, refer to "Starting Motion (p. 64)".*

7.3 Starting Motion

The procedure for PIMikroMove is described in the following.

After communication has been established between the E-873.1AT and the PC, PIMikroMove guides you through the configuration of the E-873.1AT for the positioner. It is then possible to run the first motion tests of the 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-873.1AT:

- Stage Name (0x3C): The value is used by PIMikroMove as criterion for finding a suitable parameter set in the positioner databases.
- Stage Type (0x0F000100): The value was loaded from the ID chip of the connected positioner when the E-873.1AT was switched on.

Possible configuration steps:

- If the values of the parameters 0x3C and 0x0F000100 are not identical, the ***Stage Type Configuration*** window opens. A corresponding message is displayed when a suitable parameter set is not in the positioner database.
- If the value of parameter 0x0F000100 is empty, e.g., because the positioner does not have an ID chip, the ***Start up controller*** window switches to the ***Select connected stages*** step.
- When the values of the parameters 0x3C and 0x0F000100 are identical, PIMikroMove assumes that all parameters of the E-873.1AT have already been adapted to the connected positioner. The ***Start up controller*** window goes directly to the ***Start up axes*** step.

The figures show the procedure for any electronics; the procedure for the E-873.1AT corresponds.

Requirements

✓ PIMikroMove is installed on the PC (p. 51).

✓ You have installed the latest version of the PISTAGES3.DB database onto your PC (p. 51).

✓ If PI provided a custom positioner database for your positioner, the dataset was imported into PIStages3 (p. 51).

✓ 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 positioner to the E-873.1AT (p. 50).

✓ You have established communication between the E-873.1AT and the PC with PIMikroMove (p. 59).

WARNING

Risk of electric shock due to selection of an incorrect positioner type!

Selection of an incorrect positioner type in the PC software can lead to an improper setting of the piezo voltage that is output by the E-873.1AT.

If the piezo voltage is too high touching the positioner can cause serious injury due to electric shock. Also it can cause damage to the positioner.

► Make sure that the positioner type selected in the PC software matches the positioner connected.

NOTICE

Damage due to disabled limit switch evaluation!

The collision of a moving part at the end of the travel range, or with an obstacle, as well as high accelerations, can cause damage to or considerable wear on the stage.

► Avoid motion in open-loop operation.

► If motion is necessary in open-loop operation: Stop the positioner in time. For this purpose, use the `#24`, `STP` or `HLT` command, or set the control value to zero with the `SMO` command.

► Do **not** choose parameter settings that would deactivate limit switch evaluation by the electronics.

► Check the function of the limit switches at about 10 % to 20 % of the maximum velocity.

► In the event of a malfunction of the limit switches, stop the motion immediately.

NOTICE

Oscillation!

Unsuitable settings of the notch filter and the E-873.1AT'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 stage is oscillating (unusual operating noise), switch off servo mode or the E-873.1AT immediately.

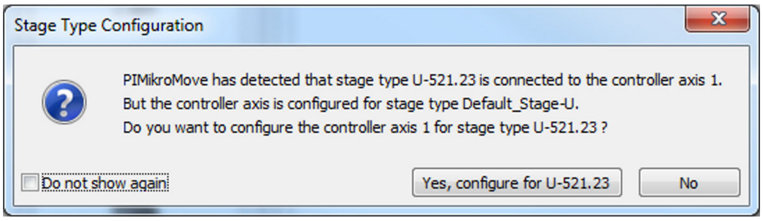
► Switch servo mode back on only after you have changed the E-873.1AT's notch filter settings and servo control parameters; see "Adjusting the Notch Filter (p. 68)" and "Optimizing the Servo Control Parameters (p. 72)".

► If a very high load causes oscillation during the reference move, follow the instructions on the reference move under "Troubleshooting" (p. 235).

Starting motion with PIMikroMove

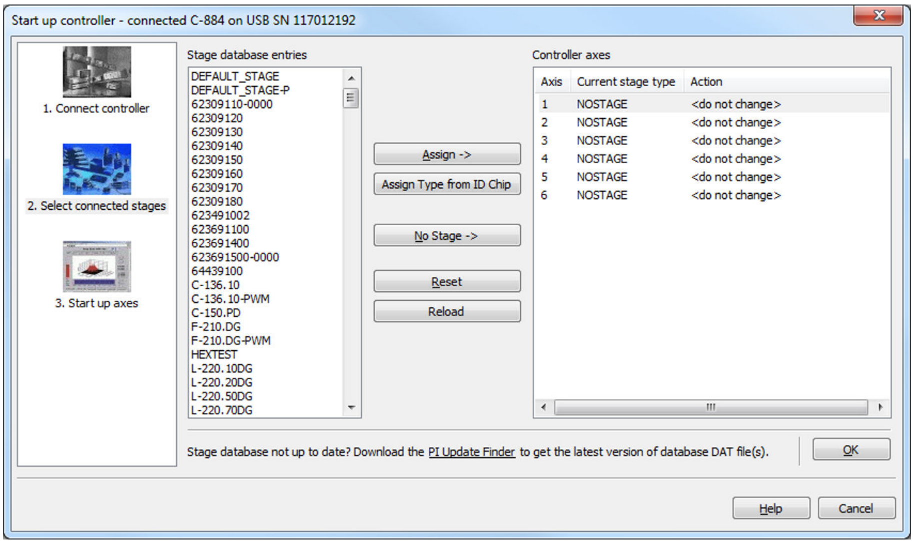
1. Load the parameter set from the positioner database into the E-873.1AT:

- When the ***Stage Type Configuration*** dialog has opened: Click the ***Yes, configure for ...*** button to load the matching parameter set from a positioner database into the E-873.1AT.



PIMikroMove has detected that stage type U-521.23 is connected to the controller axis 1.
But the controller axis is configured for stage type Default_Stage-U.
Do you want to configure the controller axis 1 for stage type U-521.23 ?

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



Stage database entries:
DEFAULT_STAGE
DEFAULT_STAGE-P
62309110-0000
62309120
62309130
62309140
62309150
62309160
62309170
62309180
623491002
623691100
623691400
623691500-0000
64439100
C-136.10
C-136.10-PWM
C-150.PD
F-210.DG
F-210.DG-PWM
HEXTEST
L-220.10DG
L-220.20DG
L-220.50DG
L-220.70DG

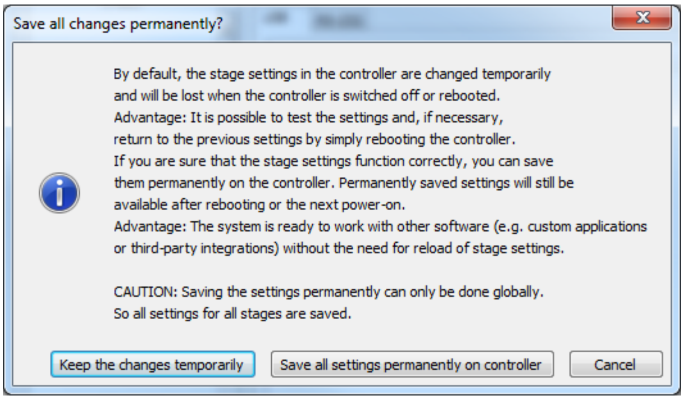
Axis	Current stage type	Action
1	NOSTAGE	<do not change>
2	NOSTAGE	<do not change>
3	NOSTAGE	<do not change>
4	NOSTAGE	<do not change>
5	NOSTAGE	<do not change>
6	NOSTAGE	<do not change>

Stage database not up to date? Download the PI Update Finder to get the latest version of database DAT file(s).

a) Select the appropriate positioner type: Click either ***Assign Type from ID Chip*** or mark the matching 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-873.1AT.

➔ *The **Save all changes permanently?** dialog is opened.*


Save all changes permanently?
Keep the changes temporarily
Save all settings permanently on controller
Cancel

By default, the stage settings in the controller are changed temporarily
and will be lost when the controller is switched off or rebooted.
Advantage: It is possible to test the settings and, if necessary,
return to the previous settings by simply rebooting the controller.
If you are sure that the stage settings function correctly, you can save
them permanently on the controller. Permanently saved settings will still be
available after rebooting or the next power-on.
Advantage: The system is ready to work with other software (e.g. custom applications
or third-party integrations) without the need for reload of stage settings.

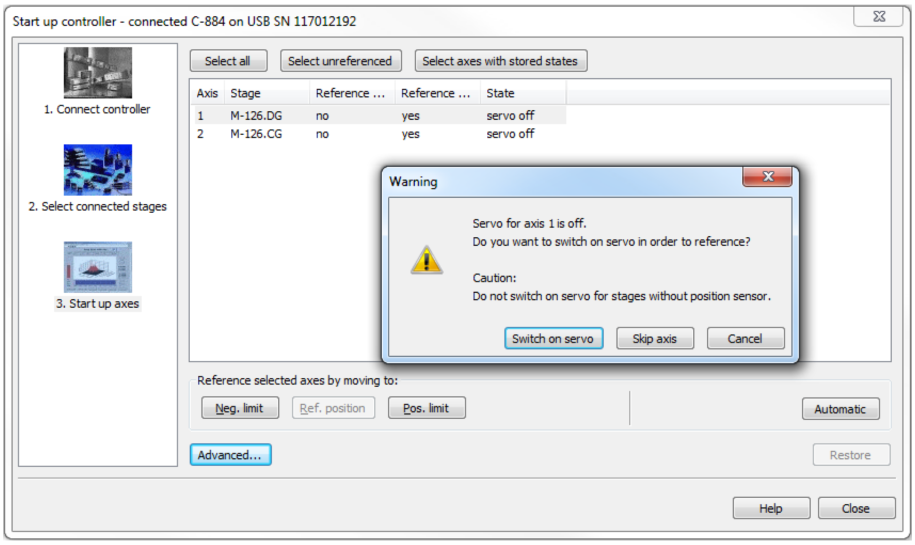
CAUTION: Saving the settings permanently can only be done globally.
So all settings for all stages are saved.

2. Specify how you want to load the parameter settings into the E-873.1AT 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-873.1AT. The settings are lost when the E-873.1AT 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-873.1AT. The settings are available immediately after switching on or rebooting the E-873.1AT and do not need to be reloaded.

➔ *The **Start up controller** window with the **Start up axes** step is displayed.*

3. For axes connected with incremental sensor: 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):
   - Start the reference move to the reference switch: Click ***Ref. position***.
   - Start the reference move to the negative limit switch: Click***Neg. limit***.
   - Start the reference move to the positive limit switch: Click***Pos. limit***.

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



Start up controller - connected C-884 on USB SN 117012192

Axis	Stage	Reference ...	Reference ...	State
1	M-126.DG	no	yes	servo off
2	M-126.CG	no	yes	servo off

Warning

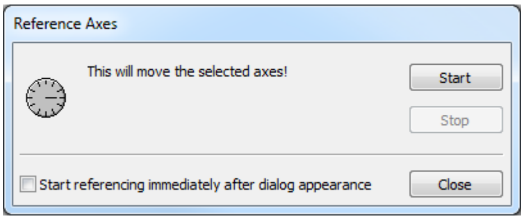
Servo for axis 1 is off.
Do you want to switch on servo in order to reference?

Caution:
Do not switch on servo for stages without position sensor.

Reference selected axes by moving to:

Switch on the servo mode by clicking on the ***Switch on servo*** button in the warning message.

➔ *The **Reference Axes** dialog appears after switching servo mode on:*



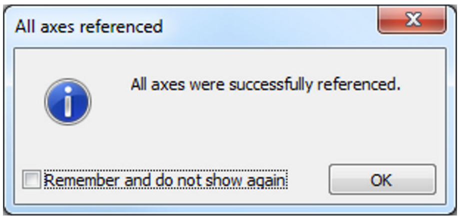
Reference Axes

This will move the selected axes!

□ Start referencing immediately after dialog appearance

   b) Click the ***Start*** button in the ***Reference Axes*** dialog.

➔ *The axis performs the reference move. The **All axes referenced** message is displayed after a successful reference move:*



All axes referenced

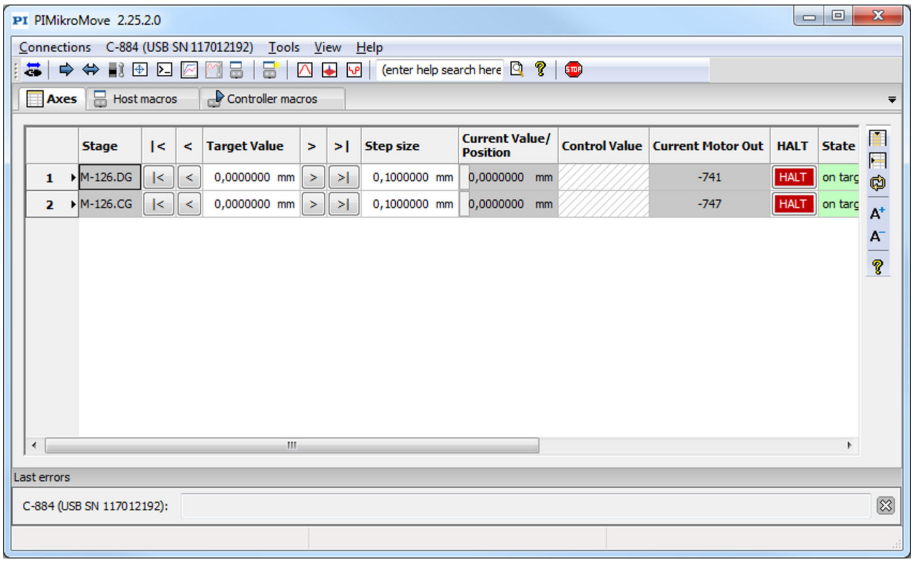
All axes were successfully referenced.

□ Remember and do not show again

   c) Close the ***All axes referenced*** message with ***OK***.

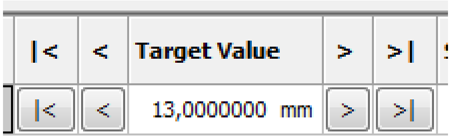
4. Click ***Close*** to close the ***Start up controller*** window.

➔ *The main window of PIMikroMove opens.*



	Stage	|<	<	Target Value	>	>|	Step size	Current Value/ Position	Control Value	Current Motor Out	HALT	State
1	M-126.DG	|<	<	0,0000000 mm	>	>|	0,1000000 mm	0,0000000 mm		-741	HALT	on targ
2	M-126.CG	|<	<	0,0000000 mm	>	>|	0,1000000 mm	0,0000000 mm		-747	HALT	on targ

5. Start testing axis motion: You can start motion by clicking the arrow keys for the corresponding axis in the PIMikroMove main window:



|<	<	Target Value	>	>|
|<	<	13,0000000 mm	>	>|

- **|<** or **>|**: Motion to the negative or positive limit of the travel range
- **<** or **>**: Motion along a certain path (specification in the ***Step size*** column)

7.4 Adapting the Dynamic Characteristics of the System

7.4.1 Adjusting the Notch Filter

The notch filter (p. 36) corrects the control value for the drive of the positioner connected to the E-873.1AT. The frequency component in the control value that would cause natural oscillation of the mechanics is reduced by the notch filter. Adjusting the notch filter frequency can be useful, particularly in the case of high loads.

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): Frequency component to be filtered out
- **Notch Filter Edge 1** (0x95): Bandwidth of the notch filter

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 fastening).

✓ Initial motion was started (p. 64) in PIMikroMove.

✓ All devices are still ready for operation.

Information

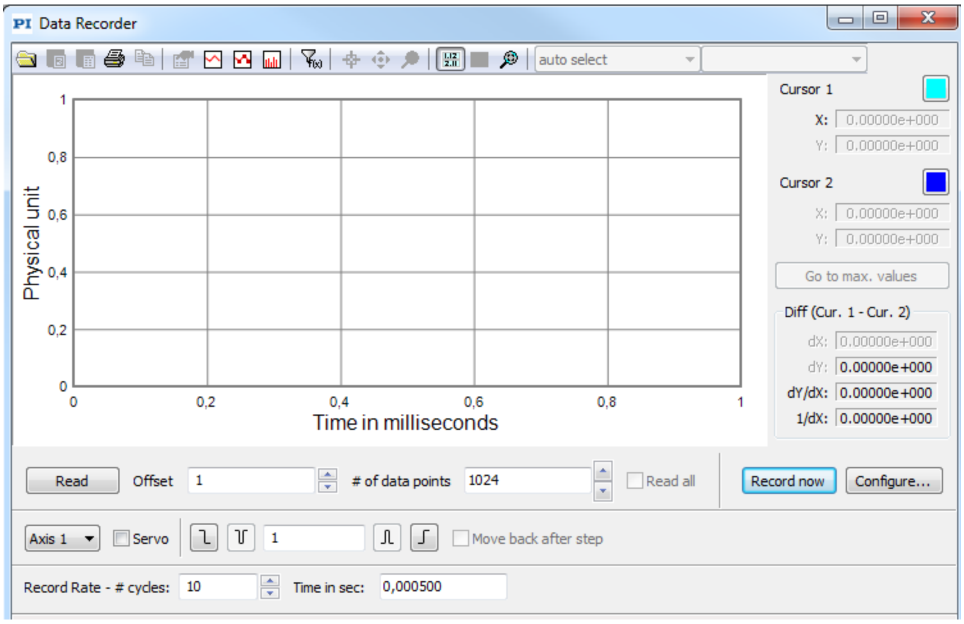
The settling behavior of the axis in closed-loop operation is influenced by the notch filter settings.

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

Recording the step response

1. Open the ***Data Recorder*** window in PIMikroMove via the ***E-873.1AT > Show data recorder*** menu item.

➔ *The* ***Data Recorder*** *window is opened.*


PI Data Recorder

Cursor 1 X: 0.00000e+000 Y: 0.00000e+000

Cursor 2 X: 0.00000e+000 Y: 0.00000e+000

Diff (Cur. 1 - Cur. 2) dX: 0.00000e+000 dY: 0.00000e+000 dY/dX: 0.00000e+000 1/dX: 0.00000e+000

Offset: 1

# of data points: 1024

□ Read all

Axis 1 □ Servo 1

□ Move back after step

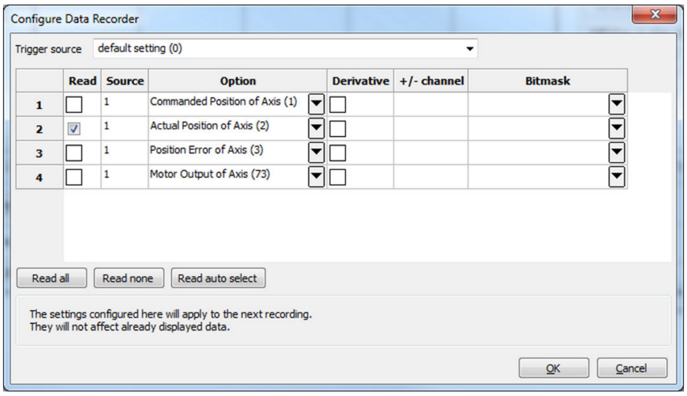
Record Rate - # cycles: 10

Time in sec: 0,000500

2. Make the following settings in the ***Data Recorder*** window:
   a) Use the ***Servo*** checkbox to make sure that servo mode is switched off: If the ***Servo*** checkbox is checked, uncheck to switch servo mode off.
   b) Set the number of data points to be read for the graphical display (max. 7616) in the ***# of data points*** field.
   c) Set the amplitude of the step to be performed to a value that is typical for your application (specified as physical units).
   d) Set the number of servo cycles to be recorded in the ***Record Rate - # cycles*** field.
3. Click the ***Configure...*** button to open the window for configuring the data recorder.

➔ *The* ***Configure Data Recorder*** *window is opened.*

4. In the ***Configure Data Recorder*** window, make sure that the "Actual Position of Axis" is selected as the variable to be recorded, and close the window with ***OK***.


Configure Data Recorder

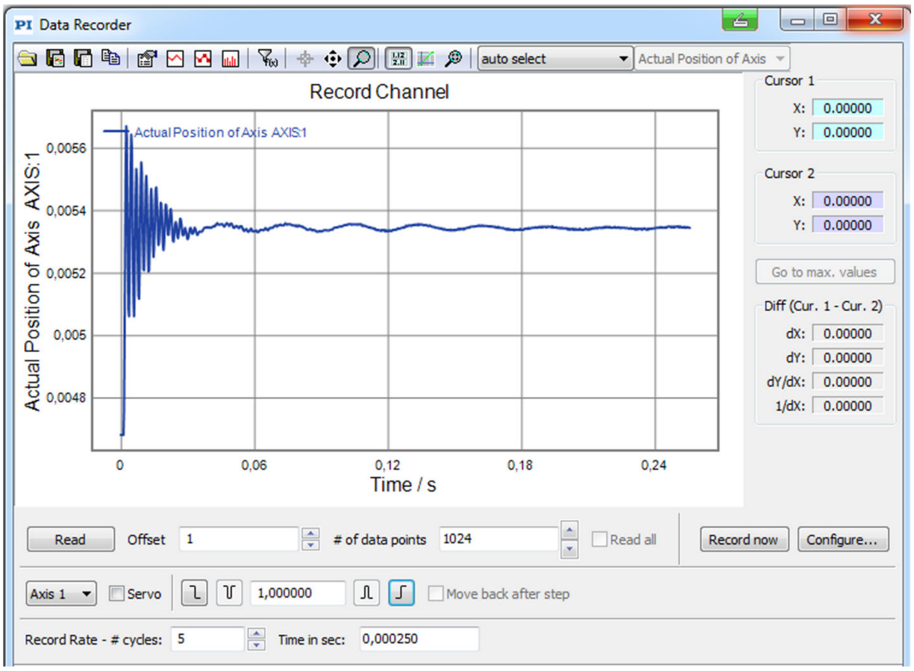
Trigger source: default setting (0)

	Read	Source	Option	Derivative	+/- channel	Bitmask
1	□	1	Commanded Position of Axis (1)	□		
2	☑	1	Actual Position of Axis (2)	□		
3	□	1	Position Error of Axis (3)	□		
4	□	1	Motor Output of Axis (73)	□		

The settings configured here will apply to the next recording.
They will not affect already displayed data.

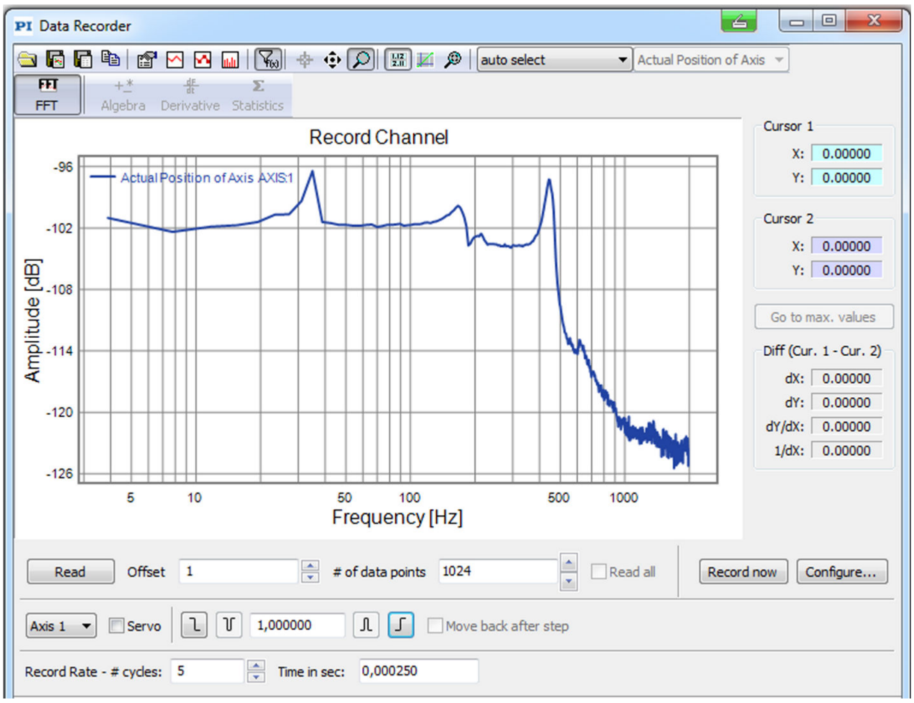
5. Start the jump in the positive direction as well as the recording by clicking the ⎍ button in the ***Data Recorder*** window.

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



6. Calculate the FFT (Fast Fourier Transformation) of the step response:
   a) Display the Data Toolbar via the button.
   b) Click the button to calculate the FFT.

➔ *The FFT of the step response is displayed graphically.*



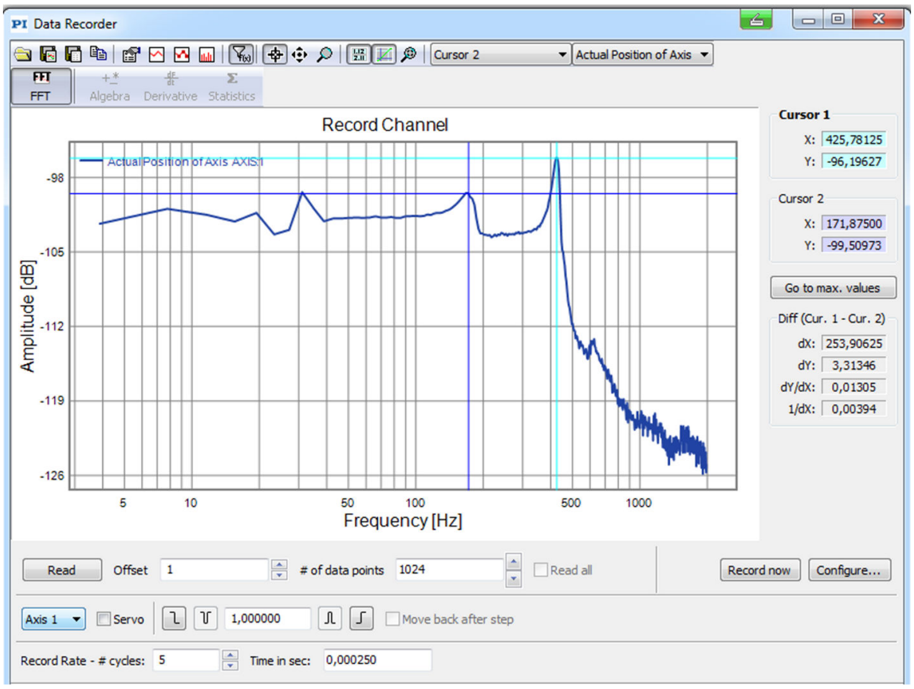
7. Determine the resonant frequency of the axis from the step response:

   If necessary, enlarge the view: Click the button, press and hold down the left mouse button, and drag the magnifying glass symbol over an area in the graphical display (clicking with the right mouse button in the graphic field returns the view to its original size).

   a) Display the cursors in the graphical display by clicking the button.
   b) Activate the cursor movement with the mouse by clicking the button.

c) Click the ***Go to max. values*** button to position cursors 1 and 2 over the resonant frequencies.

➔ *The resonant frequencies are recognizable at the distinct maximum in the FFT diagram and are displayed in the respective **X:** field of the **Cursor 1** and **Cursor 2** areas on the right next to the graphical display.*



Cursor 1
X: 425,78125
Y: -96,19627

Cursor 2
X: 171,87500
Y: -99,50973

Diff (Cur. 1 - Cur. 2)
dX: 253,90625
dY: 3,31346
dY/dX: 0,01305
1/dX: 0,00394

Offset 1 # of data points 1024

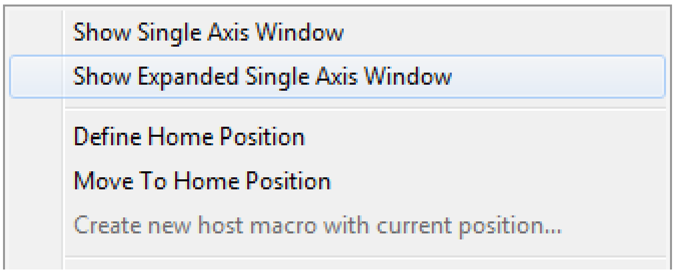
1,000000

Record Rate - # cycles: 5 Time in sec: 0,000250

Figure 13: In the example, the first resonant frequency can be seen at 425 Hz, a second at 171 Hz.

Adjusting the Notch Filter

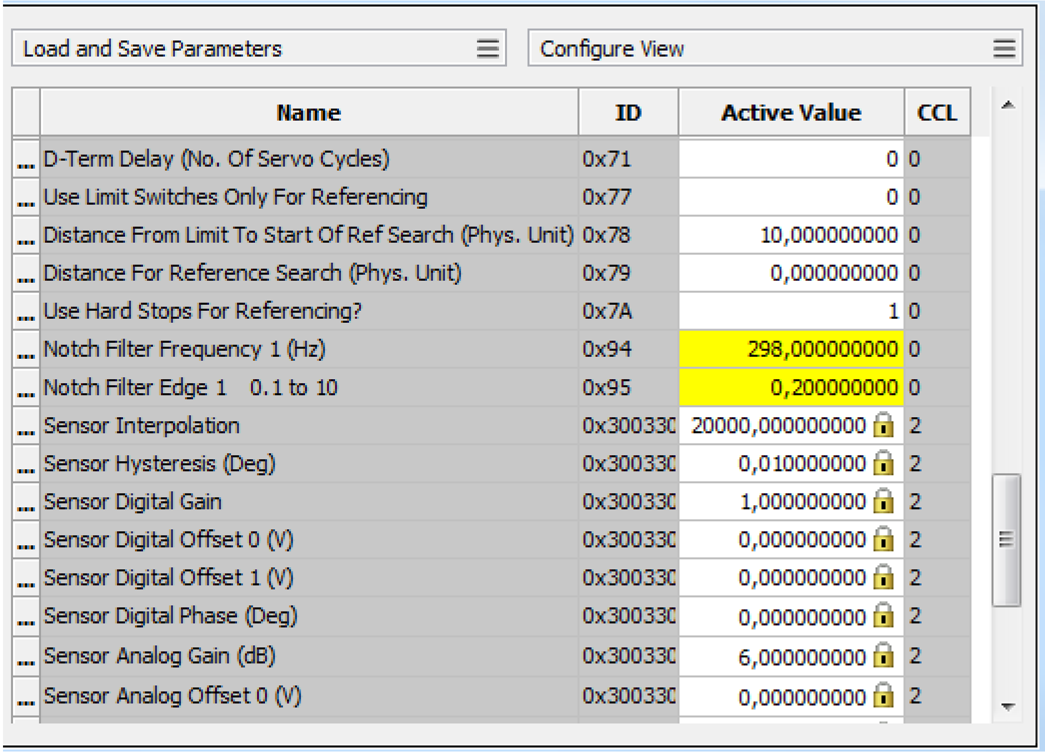
1. Open the expanded single axis window of the connected positioner in the main window of PIMikroMove: Click the corresponding line of the ***Axes*** tab with the right mouse button and select the ***Show Expanded Single Axis Window*** in the context menu.


Show Single Axis Window
Show Expanded Single Axis Window
Define Home Position
Move To Home Position
Create new host macro with current position...

2. Adapt the parameter values for the notch filter in the expanded single axis window:
   a) If the **Notch Filter Frequency 1** and **Notch Filter Edge 1** parameters are not in the list on the right-hand side of the window, click ***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.

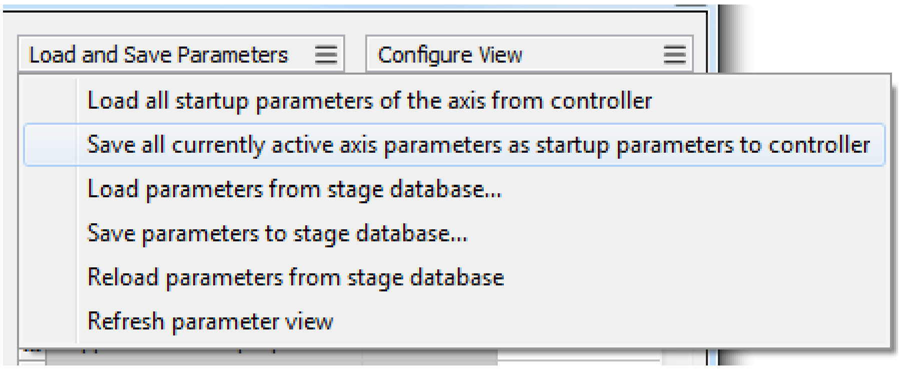
The value of the **Notch Filter Frequency 1** parameter must be set to the previously determined resonant frequency. If two resonant frequencies were determined, it should be set to a value that is roughly in the middle between both resonant frequencies. In this case, the value of the **Notch Filter Edge 1** parameter must be reduced to a value in a range of 0.3 to 0.2 in order to increase the bandwidth of the notch filter.

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.



	Name	ID	Active Value	CCL
...	D-Term Delay (No. Of Servo Cycles)	0x71	0	0
...	Use Limit Switches Only For Referencing	0x77	0	0
...	Distance From Limit To Start Of Ref Search (Phys. Unit)	0x78	10,000000000	0
...	Distance For Reference Search (Phys. Unit)	0x79	0,000000000	0
...	Use Hard Stops For Referencing?	0x7A	1	0
...	Notch Filter Frequency 1 (Hz)	0x94	298,000000000	0
...	Notch Filter Edge 1 0.1 to 10	0x95	0,200000000	0
...	Sensor Interpolation	0x300330	20000,000000000 🔒	2
...	Sensor Hysteresis (Deg)	0x300330	0,010000000 🔒	2
...	Sensor Digital Gain	0x300330	1,000000000 🔒	2
...	Sensor Digital Offset 0 (V)	0x300330	0,000000000 🔒	2
...	Sensor Digital Offset 1 (V)	0x300330	0,000000000 🔒	2
...	Sensor Digital Phase (Deg)	0x300330	0,000000000 🔒	2
...	Sensor Analog Gain (dB)	0x300330	6,000000000 🔒	2
...	Sensor Analog Offset 0 (V)	0x300330	0,000000000 🔒	2

3. Save the new settings. You have the following options:
   - Save a parameter set to the positioner database on the PC, see "Saving the Parameter Set to the Positioner Database (p. 76)".
   - Transfer the current values of the listed parameters from the volatile memory to the nonvolatile memory of the E-873.1AT by clicking ***Load and Save Parameters -> Save all currently active axis parameters as startup parameters to controller***.


Load and Save Parameters
Configure View
Load all startup parameters of the axis from controller
Save all currently active axis parameters as startup parameters to controller
Load parameters from stage database...
Save parameters to stage database...
Reload parameters from stage database
Refresh parameter view

7.4.2 Optimizing the Servo Control Parameters

Adjusting the servo controller (p. 35) optimizes the system's dynamic properties (overshoot and settling time). The optimal servo controller settings depends on your application and your requirements.

Typically, optimization is determined empirically, i.e., the behavior of the positioner is monitored with different values in closed-loop operation.

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

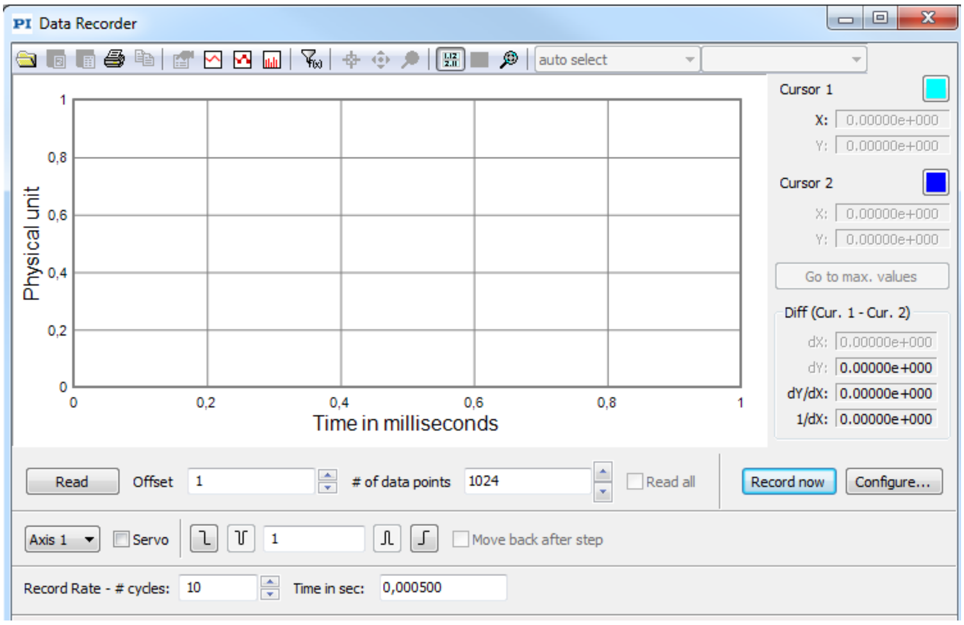
Requirements

✓ You have installed the positioner in the same way as it will be used in your application (corresponding load, orientation, and mounting).
✓ You have started initial motion (p. 64) with PIMikroMove.
✓ If necessary, you have set the notch filter (p. 68).
✓ All devices are still ready for operation.

Checking the Servo Control Parameters: Recording the Step Response

1. Open the ***Data Recorder*** window in PIMikroMove via the ***E-873.1AT > Show data recorder*** menu item.

➔ *The* ***Data Recorder*** *window is opened.*

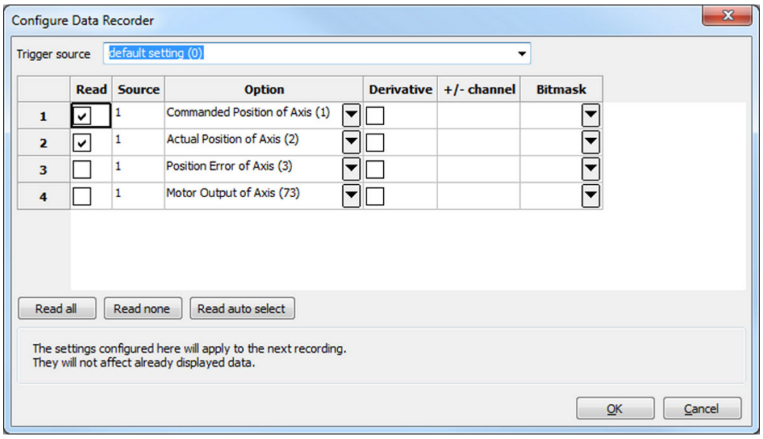


Cursor 1 X: 0.00000e+000 Y: 0.00000e+000
Cursor 2 X: 0.00000e+000 Y: 0.00000e+000
dX: 0.00000e+000
dY: 0.00000e+000
dY/dX: 0.00000e+000
1/dX: 0.00000e+000
Offset: 1
# of data points: 1024
□ Read all
Axis 1
□ Servo
1
□ Move back after step
Record Rate - # cycles: 10
Time in sec: 0,000500

2. Make the following settings in the ***Data Recorder*** window:
   a) If necessary, switch on servo mode by checking the ***Servo*** checkbox.
   b) Set the number of data points to be read for the graphical display (max. 7616) in the ***# of data points*** field.
   c) Set the amplitude of the step to be performed to a value that is typical for your application (specified as physical units).
   d) Set the number of servo cycles to be recorded in the ***Record Rate - # cycles*** field.
3. Click the ***Configure...*** button to open the window for configuring the data recorder.

➔ *The* ***Configure Data Recorder*** *window is opened.*

4. Make sure that "Commanded Position of Axis" and "Actual Position of Axis" are selected as variables to be recorded in the ***Configure Data Recorder*** window and click ***OK*** to close the window.



Configure Data Recorder

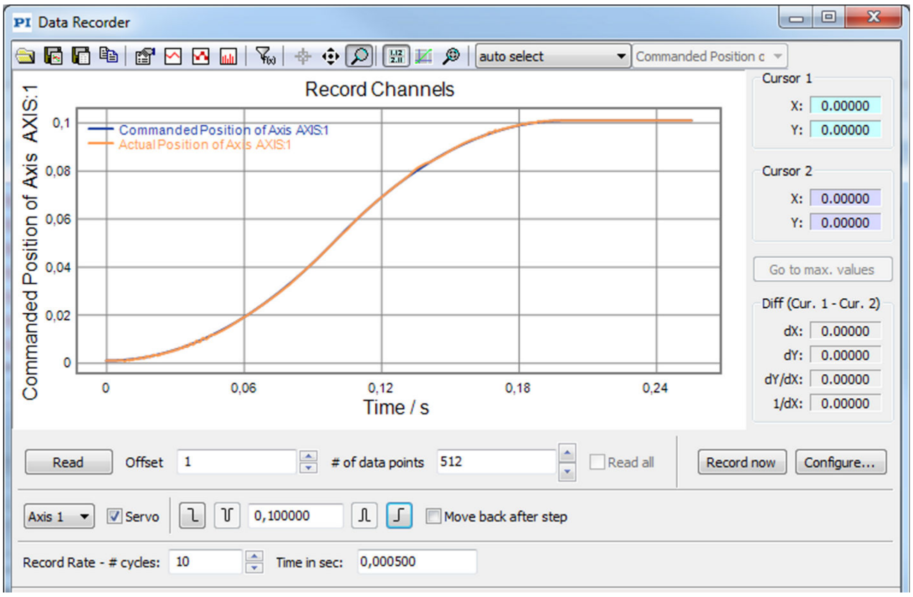
Trigger source: default setting (0)

	Read	Source	Option	Derivative	+/- channel	Bitmask
1	☑	1	Commanded Position of Axis (1)	□		
2	☑	1	Actual Position of Axis (2)	□		
3	□	1	Position Error of Axis (3)	□		
4	□	1	Motor Output of Axis (73)	□		

The settings configured here will apply to the next recording.
They will not affect already displayed data.

5. Start the jump in the positive direction as well as the recording by clicking the ⎍ button in the ***Data Recorder*** window.

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



Offset: 1

# of data points: 512

□ Read all

☑ Servo

0,100000

□ Move back after step

Record Rate - # cycles: 10

Time in sec: 0,000500

6. Check the displayed step response.

   If necessary, enlarge the view: Click the 🔍 button, press and hold down the left mouse button, and drag the magnifying glass symbol over an area in the graphical display (clicking with the right mouse button in the graphic field returns the view to its original size).

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

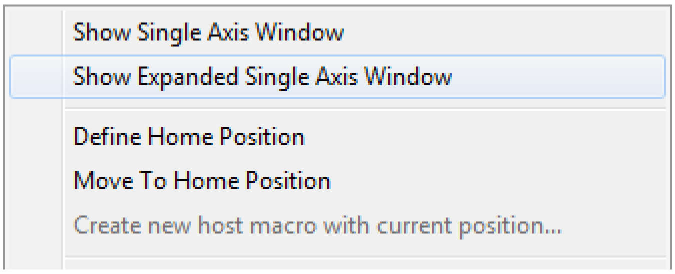
   You already have optimum parameter settings and do not have to do anything further.

   - If the result is not satisfactory:

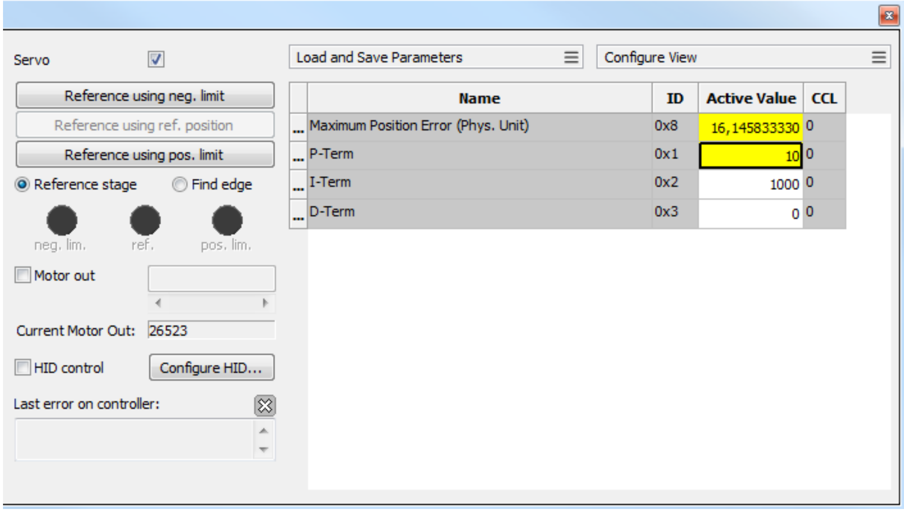
   Optimize the servo control parameters, see below.

Setting the Servo Control Parameters

1. Open the expanded single axis window of the connected positioner in the main window of PIMikroMove: Click on the corresponding line of the ***Axes*** tab with the right mouse button and select the ***Show Expanded Single Axis Window*** in the context menu.


Show Single Axis Window
Show Expanded Single Axis Window
Define Home Position
Move To Home Position
Create new host macro with current position...

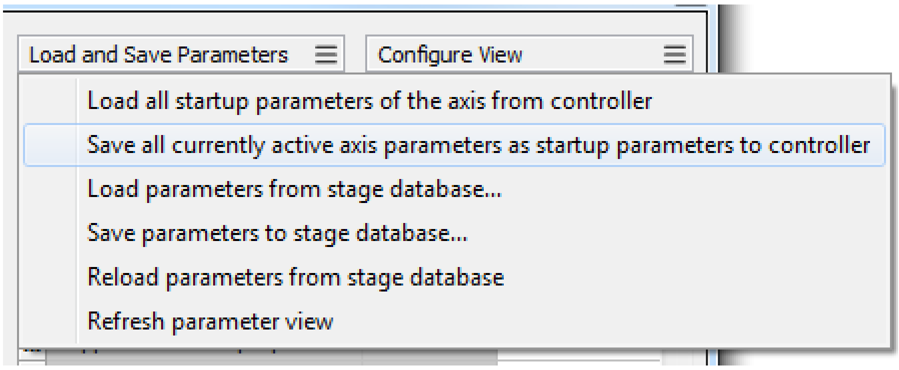
2. Enter new values for the parameters to be adapted into the expanded single axis window:
   a) If the parameters to be changed are not included in the list on the right side of the window, click on ***Configure View -> Select parameters...*** and add them to the list.
   b) Type the new parameter value 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.



Current Motor Out: 26523

	Name	ID	Active Value	CCL
...	Maximum Position Error (Phys. Unit)	0x8	16,145833330	0
...	P-Term	0x1	10	0
...	I-Term	0x2	1000	0
...	D-Term	0x3	0	0

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.
4. If you are satisfied with the result and want to keep the changed parameter settings, save the new settings. You have the following options:
   - Save a parameter set to the positioner database on the PC, see "Saving the Parameter Set to the Positioner Database (p. 76)".
   - Transfer the current values of the listed parameters from the volatile memory to the nonvolatile memory of the E-873.1AT by clicking ***Load and Save Parameters -> Save all currently active axis parameters as startup parameters to controller***.


Load and Save Parameters
Configure View
Load all startup parameters of the axis from controller
Save all currently active axis parameters as startup parameters to controller
Load parameters from stage database...
Save parameters to stage database...
Reload parameters from stage database
Refresh parameter view

7.5 Making Data Backups

Saving the Parameter Set to the Positioner Database

1. Click ***Load and Save Parameters -> Save parameters to stage database...*** in the expanded single axis window in PIMikroMove.

➔ *The **Save Parameters as User Stage Type** dialog opens.*

2. Save the changed parameter values as new positioner type in the ***Save Parameters as User Stage Type*** dialog:
   a) Leave the entry in the ***Parameters of axis*** field unchanged.
   b) Enter the name for the new positioner type into the ***Save as*** field.
   c) Click ***OK***.

➔ *The new positioner type was saved to the PISTAGES3.DB database. The display of the connected positioner type was updated in the single axis window and in the main window of PIMikroMove.*

7.5.1 Saving Parameter Values

The E-873.1AT is configured via parameters, e.g., for adapting to the connected positioner. The parameter values can be saved to a text file so that they can be restored at a later time.

Information

Changing parameter values can cause undesirable results.

► Create a backup copy on the PC before changing the parameter settings of the E-873.1AT. 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-873.1AT to a particular positioner.

Parameter values saved in a text file on the PC can be loaded back to the E-873.1AT in PIMikroMove or PITerminal. The ***Send file...*** button is available for this purpose in the send command window. Before loading into the E-873.1AT, 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 between the E-873.1AT and the PC with PIMikroMove (p. 59) or PITerminal.

Saving Parameter Values in a Text File

1. If you are using PIMikroMove, open the window for transmitting commands: Select ***Tools > Command entry*** in the main window or press F4 on the keyboard.

➔ *After communication has been established, the main window is opened in PITerminal automatically and commands can then be sent.*

2. Query the parameter values that you want backup.
   - If you want to save the parameter values from the volatile memory of the E-873.1AT: Send the SPA? command.
   - If you want to save the parameter values from the nonvolatile memory of the E-873.1AT: Send the SEP? command.
3. Click the ***Save...*** button.

➔ *The **Save content of terminal as textfile** window opens.*

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

7.5.2 Saving Controller Macros

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

The procedure for PIMikroMove is described in the following.

Saving Controller Macros on the PC

1. Select the ***Controller macros*** tab in PIMikroMove's 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 a macro.
   - To select several macros, hold down the Shift button and click the desired entries in the list.
   - To deselect, click an open 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) in the directory selected of the PC.*

Loading Controller Macros from the PC to the E-873.1AT

1. Select the ***Controller macros*** tab in PIMikroMove's main window.
2. Load macros from the PC to the E-873.1AT:
   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-873.1AT.
   c) Click ***Open***.

➔ *For each selected text file (<macro name>.txt), the content is loaded as a macro <macro name> into the E-873.1AT.*

8 E-873.1AT Functions

8.1 Protective Functions of the E-873.1AT

The E-873.1AT has functions that are intended to protect it against damage.

8.1.1 Protection Against Overheating

If a certain internal temperature is reached, the E-873.1AT 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. 79) for the E-873.1AT.

8.1.2 Behavior with Motion Error

There is a motion error when the position error (i.e., the difference between the current and commanded position) exceeds the specified maximum value in closed-loop operation. The maximum value for the deviation is defined by the following parameters:

- Maximum Position Error (Phys. Unit) (0x8): Exceeding the maximum value will trigger a motion error.

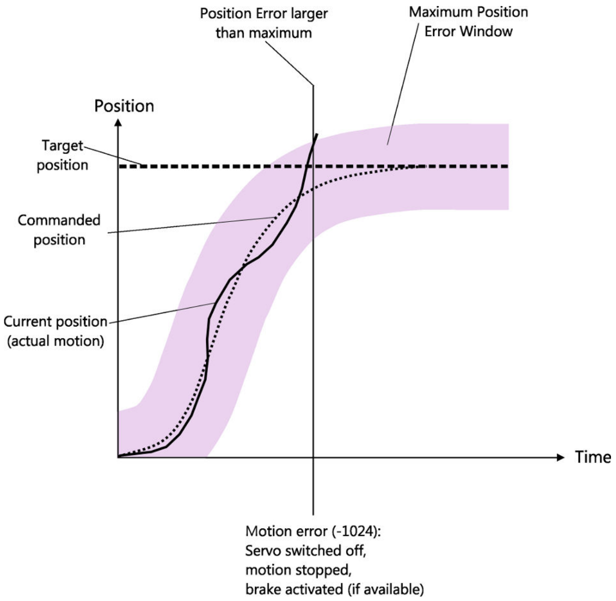

Position Error larger than maximum
Maximum Position Error Window
Position
Target position
Commanded position
Current position (actual motion)
Time
Motion error (-1024): Servo switched off, motion stopped, brake activated (if available)

Figure 14: Determining motion errors

- PIShift Threshold DeltaPos For Hardstop (Phys. Unit) (0x1F0007C2): A motion error will only be triggered after the maximum value has been exceeded for a specific time.

Motion errors can have the following causes, for example:

- Malfunction of the drive
- Malfunction of the position sensor
- Fault in the mechanics

If motion error occurs, the E-873.1AT reacts as follows to protect the system against damage:

- Servo mode is switched off for the axis.

- If applicable, the brake is activated for the axis in question.
- All motion is stopped.
- Error code -1024 is output.

Then restore the operational readiness (p. 79) for the E-873.1AT.

8.1.3 Behavior During System Errors

There is a system error when E-873.1AT is not responsive.

For example, the cause of a system error can be a buffer overflow in the firmware of the E-873.1AT.

If a system error occurs the E-873.1AT reacts as follows:

- After a certain delay, the safety function of the Watchdog Timer initiates a reboot of the E-873.1AT.

8.1.4 Restoring Operational Readiness

Restoring the E-873.1AT's Operational Readiness

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-873.1AT has not overheated.
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.*

8.1.5 Commands

E		Page
ERR?	Get Error Number	136
S		**Page**
SVO	Set Servo Mode	172

8.1.6 Parameters

0x8	Maximum Position Error (Phys. Unit)	Maximum position error. A motion error occurs when the difference between the current position and the commanded position exceeds this value and the movement of the axis is stopped.

0x1F0007C2	PIShift Threshold DeltaPos For Hardstop (Phys. Unit)	Maximum position error for certain period of time. If the difference between the commanded and actual position is greater than this value for a certain period of time, a motion error is triggered and stops axis motion. Difference to parameter 0x8 (Maximum Position Error): The maximum position error triggers a motion error immediately and stops the movement of the axis.

8.2 Data Recorder

The E-873.1AT contains a real-time data recorder. The data recorder can record different variables, e.g., the current position of an axis.

The recorded data is stored temporarily in data recorder tables. Each data recorder table contains the data of one data source.

The E-873.1AT has 4 data recorder tables with 7616 data points each.

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.

8.2.1 Setting up the Data Recorder

Reading general information from the data recorder

1. Send the HDR? command.

➔ *The options available for recording and triggering are displayed together with the information on additional parameters and commands for data recording.*

Determining the data to be recorded

1. Configure the data recorder with the DRC command.

➔ *This assigns the data sources and the recording options to the data recorder tables. The current configuration can be read out with the DRC? command.*

Data recorder tables with the record option 0 are deactivated, i.e., nothing is recorded. By default, the data recorder tables of the E-873.1AT 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

Setting the trigger for recording

1. Set the trigger option with the DRT command.
   The trigger specifies how the recording is to be triggered. The current trigger option can be queried with the DRT? command.

➔ *The trigger option applies to all data recorder tables whose record option is not set to 0.*

Setting the record table rate

1. Send the RTR command to set the recording rate of the data recorder.
   The parameter indicates the number of servo cycles required for recording each data point. The recording rate can be read out with the RTR? command.
   As the record table rate increases, the maximum duration of the data recording is increased.

Configuring data processing

1. Configure recorded data processing with the following parameters:
   - Recorded Points Per Trigger (0x16000001)

8.2.2 Starting Data Recording

Starting the Recording

1. Start the recording with the trigger option set with DRT.
   Irrespective of the set trigger option, data recording is always triggered when step response measuring is started with STE.
   The data recording always takes place for all data recorder tables whose record option is not set to 0. It ends when the data recorder tables are full.

8.2.3 Reading Out Recorded Data

Information

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

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

Reading data out

1. Read the last recorded data out with the DRR? command.
➔ *The data is output in GCS array format.*
2. Query the number of points contained in the last recording with the DRL? command.

8.2.4 Commands

D		Page
DRC	Set Data Recorder Configuration	133
DRC?	Get Data Recorder Configuration	134
DRL?	Get Number Of Recorded Points	134
DRR?	Get Recorded Data Values	134
DRT	Set Data Recorder Trigger Source	135
DRT?	Get Data Recorder Trigger Source	136
H		**Page**
HDR?	Get All Data Recorder Options	141
R		**Page**
RTR	Set Record Table Rate	166
RTR?	Get Record Table Rate	166
S		**Page**
STE	Start Step And Response Measurement	172

8.2.5 Parameters

0x16000001	Recorded Points Per Trigger	Number of data points recorded per trigger impulse. 0 Unlimited (default) n>0 n data points are recorded Maximum value for n at E-873.1AT: 7616

8.3 Digital Inputs and Outputs

The E-873.1AT's digital inputs and outputs are available at the I/O socket. The number of available input and output lines to the E-873.1AT can be queried with the `TIO?` command.

Overview

Connector of the E-873.1AT for digital inputs and outputs

8.3.1 Connecting the Digital Outputs

Tools and Accessories

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

Information

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

Connecting a Device to be Triggered

1. Connect an appropriate device to one of pins 5, 6, 7, and 8 of the *I/O* socket of the E-873.1AT.

8.3.2 Connecting the Digital Inputs

Tools and Accessories

- Suitable signal source:
  - If the digital inputs are to be used in macros, it is possible for example, to connect the C-170.PB pushbutton box, which is available as an optional accessory.
  - 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)

Information

Digital TTL input signals can be fed to the E-873.1AT via pins 1, 2, 3, and 4 of the *I/O* socket.

The digital inputs can be used as follows:

- Use in macros
- Source for the reference switch and limit switch signals of the axis

Information

The digital inputs on the *I/O* socket can also be used as analog inputs.

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

Connecting a Digital Signal Source

1. If you want to use the digital inputs in macros or as switch signals: Connect a suitable signal source to one of the pins 1, 2, 3, or 4 of the *I/O* socket of the E-873.1AT.
2. If you want to use the digital inputs for HID control, follow the instructions in "Connecting an HID (p. 91)".

8.3.3 Digital Output Signals

The number of output lines available on the E-873.1AT can be queried with the `TIO?` command.

External devices can be triggered via the digital outputs of the E-873.1AT. Potential applications:

- Linking the trigger output to the motion of the axis (`CTO`, `TRO`; query with `CTO?`, `TRO?`)
- Direct switching of output lines (`DIO`), e.g., in macros

Setting "Position Distance" Trigger Mode

The **Position Distance** trigger mode is suitable for scanning applications. A trigger pulse is output as soon as the axis has covered the distance set with CTO parameter ID = 1 (TriggerStep). The pulse width is one servo cycle. The distance unit (TriggerStep) depends on the settings of the Numerator Of The Counts-Per-Physical-Unit Factor parameter (0xE) and the Denominator Of The Counts-Per-Physical-Unit Factor parameter (0xF). Default is mm.

1. Configure the digital output line <TrigOutID> to be used as trigger output:
   a) Send `CTO <TrigOutID> 2 A`, where A indicates the axis to be moved.
   b) Send `CTO <TrigOutID> 3 0`, where 0 determines the Position Distance trigger mode.
   c) Send `CTO <TrigOutID> 1 S`, where S indicates the distance.
2. If you want to enable the conditions for trigger output, send `TRO <TrigOutID> 1`.

Example: A pulse on digital output line 1 is output each time axis 1 of the mechanics covers 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 determine the start and stop values for triggering for limiting the range and 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.

The following commands must be sent to configure the <TrigOutID> digital output line to be used as trigger output:

`CTO <TrigOutID> 2 A`, where A indicates the axis to be moved

`CTO <TrigOutID> 3 0`, where 0 determines the Position Distance trigger mode

`CTO <TrigOutID> 1 S`, where S indicates the distance

`CTO <TrigOutID> 8 Start`, where Start indicates the start value

`CTO <TrigOutID> 9 Stop`, where Stop indicates the stop value

Example for positive direction of axis motion: A pulse on digital output line 1 is output every time axis 1 of the mechanics covers a distance of 0.1 µm, as long as axis 1 is moving in a 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
```

Example for negative direction of axis motion: A pulse on digital output line 1 is output every time axis 1 of the mechanics covers a distance of 0.1 µm, as long as axis 1 is moving in a negative direction of motion within the range of 0.55 µm to 0.2 µm (stop value < start value).

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
```

Setting Up "On Target" Trigger Mode

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

1. Configure the digital output line <TrigOutID> to be used as trigger output:
   a) Send `CTO <TrigOutID> 2 A` where A indicates the axis to be moved.
   b) Send `CTO <TrigOutID> 3 2`, where 2 specifies the On Target trigger mode.
2. If you want to enable the conditions for trigger output, send `TRO <TrigOutID> 1`.

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

Send:

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

Setting Up "Motion Error" Trigger Mode

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

Information

A motion error occurs when the difference between the current and commanded position exceeds the maximum specified value in closed-loop operation. The maximum value for the deviation is specified by the **Maximum Position Error (Phys. Unit)** parameter (0x8).

1. Configure the digital output line <TrigOutID> to be used as trigger output:
   a) 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`.

Setting "In Motion" Trigger Mode

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

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

Example: Digital output 1 is to be active when axis 1 of the mechanics is in motion.

Send:

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

Setting "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 has been 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 the stop value depends on the settings for the Numerator Of The Counts-Per-Physical-Unit Factor parameter (0xE) and the Denominator Of The Counts-Per-Physical-Unit Factor parameter (0xF). Default is mm.

1. Configure the digital output line <TrigOutID> to be used as trigger output:
   a) Send `CTO <TrigOutID> 2 A`, where A indicates the axis to be moved.
   b) Send `CTO <TrigOutID> 3 7`, where 7 determines the Position+Offset trigger mode.
   c) Send `CTO <TrigOutID> 1 S`, where S indicates the distance.
   d) Send `CTO <TrigOutID> 10 TriPos`, where TriPos indicates the position for the output of the first trigger pulse.
   e) Send `CTO <TrigOutID> 9 Stop`, where Stop indicates the stop value.
2. If you want to enable the conditions for trigger output, send `TRO <TrigOutID> 1`.

Example 1: The first trigger pulse is to be output on digital output line 1 if the absolute position of axis 1 is 1.5 mm. A pulse should then be output on this line every time axis 1 has covered a distance of 0.1 µm 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 µm 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
TRO 1 1
```

Setting "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 specified position (TriggerPosition).

The unit for TriggerPosition depends on the settings for the Numerator Of The Counts-Per-Physical-Unit Factor parameter (0xE) and the Denominator Of The Counts-Per-Physical-Unit Factor (0xF) parameter. Default is mm.

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

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

Send:

```
CTO 1 2 1
CTO 1 3 8
CTO 1 10 1.5
TRO 1 1
```

8.3.4 Digital Input Signals

The number of input lines available on the E-873.1AT can be queried with the `TIO?` command. The status of the digital input lines can be queried with the `DIO?` command.

Application possibilities of the digital input signals:

- Use in macros (`CPY`, `JRC`, `MEX`, `WAC`)
- Use as switch signals (`FED`, `FNL`, `FPL`, `FRF`)
- Use for HID control (p. 90)

Using Digital Input Signals as Switch Signals

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

Information

The level of the digital input signal that you use instead of the reference switch or a built-in limit switch may only change once across the entire travel range.

► Use suitable signal sources.

► If necessary, invert the signal logic of the digital input lines by setting the corresponding parameters.

Using Digital Input as Reference Signal

Information

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

1. Select the source of the reference signal for the axis by changing the Source Of Reference Signal parameter (0x5C).
2. If necessary, invert the signal logic of the digital input line by setting the Invert Reference? parameter (0x31).

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.

The Has No Limit Switches? parameter (0x32) determines whether the E-873.1AT evaluates the signals of the limit switch built into the connected mechanics. This parameter has no influence on the use of digital input lines as the source of the limit switch signal.

1. Select the source(s) of the negative limit switch signal for an axis by changing the Source Of Negative Limit Signal parameter (0x5D).
2. Select the source(s) of the positive limit switch signal for an axis by changing the Source Of Positive Limit Signal parameter (0x5E).
3. If necessary, invert the signal logic of the digital input lines by setting the Invert Digital Input Used For Negative Limit parameter (0x5F) and Invert Digital Input Used For Positive Limit parameter (0x60).

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.

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

Using Digital Input Signals for the HID Control

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

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

These HID axes can be used to control the relative target position of the axis connected to the E-873.1AT.

8.3.5 Commands

C		Page
CPY	Copy Into Variable	125
CTO	Set Configuration Of Trigger Output	127
CTO?	Get Configuration Of Trigger Output	129
D		Page
DIO	Set Digital Output Lines	132

DIO?	Get Digital Input Lines	132
F		Page
FED	Find Edge	137
FNL	Fast Reference Move To Negative Limit	138
FPL	Fast Reference Move To Positive Limit	138
FRF	Fast Reference Move To Reference Switch	139
J		Page
JRC	Jump Relatively Depending On Condition	154
M		Page
MEX	Stop Macro Execution Due To Condition	159
T		Page
TIO?	Tell Digital I/O Lines	173
TRO	Set Trigger Output State	175
TRO?	Get Trigger Output State	175
W		Page
WAC	Wait For Condition	178

8.3.6 Parameters

0x14	Has Reference?	Do the mechanics have a reference switch? Activates respectively deactivates reference moves to the installed reference switch. 0 No reference switch 1 Reference switch available
0x5C	Source Of Reference Signal	Reference signal source for axis motion to the reference switch.
0x31	Invert Reference?	Should the reference signal be inverted? Inverts the signal of the reference switch or a digital input that is used instead of the reference switch. 0 Reference signal not inverted 1 Reference signal inverted
0x32	Has No Limit Switches?	Do the mechanics not have limit switches? Activates motion stop at the installed limit switches. 0 Mechanics have limit switches 1 Mechanics do not have limit switches
0x5D	Source Of Negative Limit Signal	Reference signal source for axis motion to the negative travel range limit.
0x5E	Source Of Positive Limit Signal	Reference signal source for axis motion to the positive travel range limit.
0x5F	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.

0x60	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.

8.4 Analog Input Signals

The analog inputs of the E-873.1AT are available on the *I/O* socket. The number of input lines available on the E-873.1AT can be queried with the TAC? command. The voltage at the analog inputs can be queried with the TAV? command. The analog input signals can be recorded with the data recorder (DRC command with recording option 81).

Potential applications:

- Use in macros (CPY, JRC, MEX, WAC)
- Scan applications with PIMikroMove

8.4.1 Connecting Analog Signal Sources

Overview

Analog input signals (0 V to + 5 V) can be fed into the E-873.1AT via pins 1, 2, 3, and 4 of the *I/O* socket.

Tools and Accessories

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

Information

The analog inputs of the *I/O* socket can also be used as digital inputs.

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

Connecting an Analog Signal Source

1. Connect an appropriate signal source to one of pins 1, 2, 3 or 4 of the *I/O* socket of the E-873.1AT.

8.4.2 Commands

C		Page
CPY	Copy Into Variable	125
D		Page
DRC	Set Data Recorder Configuration	133
J		Page
JRC	Jump Relatively Depending On Condition	154

M		Page
MEX	Stop Macro Execution Due To Condition	159
T		Page
TAC?	Tell Analog Channels	173
TAV?	Get Analog Input Voltage	173
W		Page
WAC	Wait For Condition	178

8.4.3 Parameters

8.5 Controlling with HID

HID (Human Interface Device) denotes an input or output device connected to the E-873.1AT and is intended to be operated manually. Joysticks and gamepads are typical HIDs.

HID control means controlling motion variables of a positioner's axis connected to the E-873.1AT by displacing an axis of the HID.

8.5.1 How it Works

An HID's axis can control the following motion variables of a positioner's axis connected to the E-873.1AT:

- Absolute target position: The relationship between the displacement of the axis of the human interface device and the target position of the positioner axis is created from a lookup table by the E-873.1AT.
- Relative target position: The displacement of the HID axis determines the frequency for moving the controlled positioner axis: The further the HID axis is displaced, the higher the frequency and therefore the velocity of the positioner axis.
- Velocity: Product of the lookup table corresponding to the current displacement of the axis of the HID and the currently valid velocity of the controller axis.
- Maximum Velocity: Product of the lookup table value corresponding to the current displacement of the axis of the HID and the currently valid maximum velocity of the controller axis.

Information

The ***maximum velocity*** motion variable can only be assigned to an HID axis when the ***velocity*** motion variable has been assigned to a different HID axis.

Information

Motion commands are not permitted when HID control is enabled for the axis.

HID control is not possible in open-loop operation (servo mode Off).

8.5.2 HID Control Configuration

Control of the axis/axes connected to the E-873.1AT by the HID's axes is configured via the `HIA` command. The current HID control configuration can be queried with the `HIA?` command. The direction of motion of HID-controlled axes can be inverted via the **Invert Direction Of Motion For Joystick-Controlled Axis?** parameter (0x61).

Assigning a lookup table to an HID axis is done with the HDT command. The current lookup table assignments to HID axes can be queried with the HDT? command. The values in the lookup table are factors that are applied to the motion parameter to be controlled during HID control. The E-873.1AT's firmware gives a choice of two predefined lookup table types (linear and parabolic) and allows four customer-specific lookup tables to be filled with individual values. Lookup tables can be filled with values with the HIT command. HIT? queries the values of the points in the lookup tables.

Use the HIN command to activate/deactivate HID control of the axes connected to the E-873.1AT. The HIN? command queries the activation state of HID control. When HID control is deactivated, the target position is set to the current position of the controlled axis.

The HIS? command queries the properties of the HID's operating elements. The current status of the HID buttons can be queried with HIB? and the current displacement of HID axes with HIE?.

8.5.3 Programming HID Control

HID output units (e.g., buttons and LEDs) can be used for example, in controller macros to program HID control.

8.5.4 Connecting an HID

Overview

E-873.1AT's connector for a manually operated device such as a joystick or gamepad

Tools and Accessories

- Analog joystick for operating with 0 to 3.3 V (e.g., C-819.20, or C-819.30, available as optional accessory (p. 15))
- If a C-819.20 joystick is to be connected to two controllers: C-819.20Y Y cable (available as optional accessory)

Connecting an HID to the E-873.1AT

1. If you want to use axis 1 and/or axis 2 of the HID, connect the following to the E-873.1AT's *Joystick* socket:
   - If you want to operate a C-819.20 joystick with this controller only, connect it directly to the controller.
   - If you want to operate a C-819.20 joystick with two controllers (i.e., two axes), connect the joystick to the C-819.20Y cable and connect both controllers to the X and Y branches of the cable. The power is supplied to the joystick via the X branch. For this reason, the X branch has to be connected to a controller even if HID control is not to be activated for this controller.
   - If you want to connect an axis of a C-819.30 joystick, connect the corresponding cable of the joystick to the controller.
2. If you want to use axis 3 and/or 4 of the HID, connect an HID to the E-873.1AT's *I/O* socket.

First of all, we recommend testing the HID's operating elements after connecting it to the E-873.1AT. If the response behavior of the HID's axes does not meet your requirements, it is possible to calibrate its axes.

HID control can be set up and activated after testing and optional calibration of the axes.

The procedure for PIMikroMove is described in the following.

8.5.5 Testing the HID and Calibrating the Axes

The positioner does not need to be connected to the E-873.1AT for testing the HID and calibrating its axes.

When calibrating the HID's axes in PIMikroMove, the appropriate lookup table must be selected. The parabolic lookup table allows for greater sensitivity when moving slowly. Lookup tables named "User Table" are intended to be filled with individual values.

Requirements

✓ PIMikroMove is installed on the PC (p. 51).

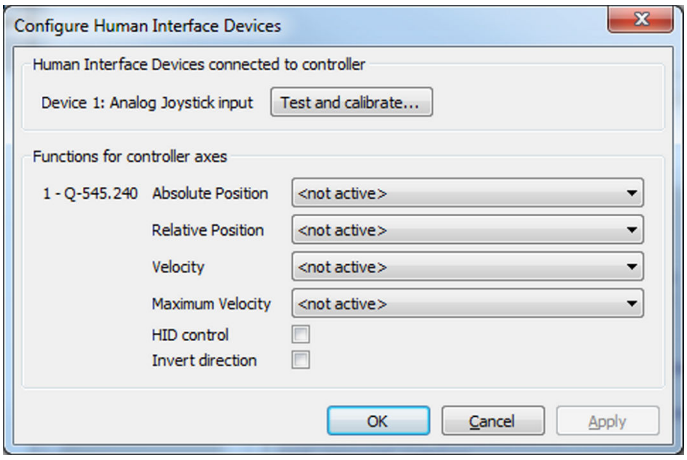
✓ PIMikroMove has established communication between the E-873.1AT and the PC (p. 59).

✓ The E-873.1AT has been connected to the HID (p. 91).

Testing the HID

1. Open the window for configuring HID control via the ***E-873.1AT > Configure controller HIDevice(s)...*** menu item in PIMikroMove's main window.

➔ *The **Configure Human Interface Devices** window opens.*



Configure Human Interface Devices

Human Interface Devices connected to controller

Device 1: Analog Joystick input

Functions for controller axes

1 - Q-545.240 Absolute Position <not active>

Relative Position <not active>

Velocity <not active>

Maximum Velocity <not active>

HID control □

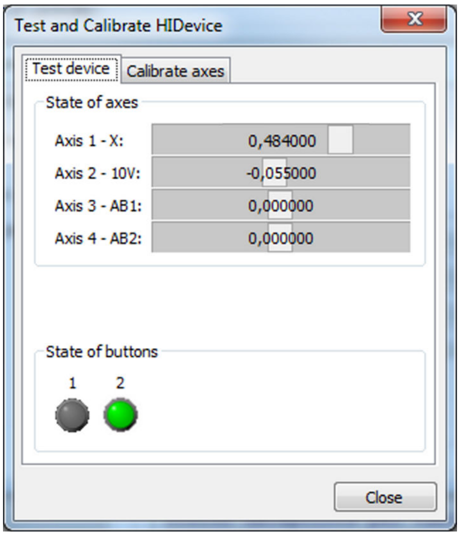
Invert direction □

2. Open the window for testing and calibrating the HID by clicking ***Test and calibrate...***.

➔ *The **Test and Calibrate HIDevice** window opens.*

3. Select the ***Test device*** tab in the ***Test and Calibrate HIDevice*** window and test the HID's operating elements:
   a) Move the HID's axes and at the same time, watch the status indicators in the ***State of axes*** area.
   b) Press the HID's buttons and at the same time, watch the status indicators in the ***State of buttons*** area.
   c) Enter various values into the fields in the ***State of LEDs*** area (if any) and at the same time, watch the behavior of the corresponding operating elements on the HID.

   ➔ *In the example shown, a C-819.20 joystick is connected to the **Joystick** socket of the E-873.1AT. The E-873.1AT supports one axis of this joystick. The identifier of the axis is 1, the name is X. The two buttons of the C-819.20 joystick are available on the E-873.1AT via the identifiers 1 and 2. Current status in the figure: The axis of the joystick is displaced in the positive direction, and button 2 is pressed.*



State of axes

Axis 1 - X: 0,484000

Axis 2 - 10V: -0,055000

Axis 3 - AB1: 0,000000

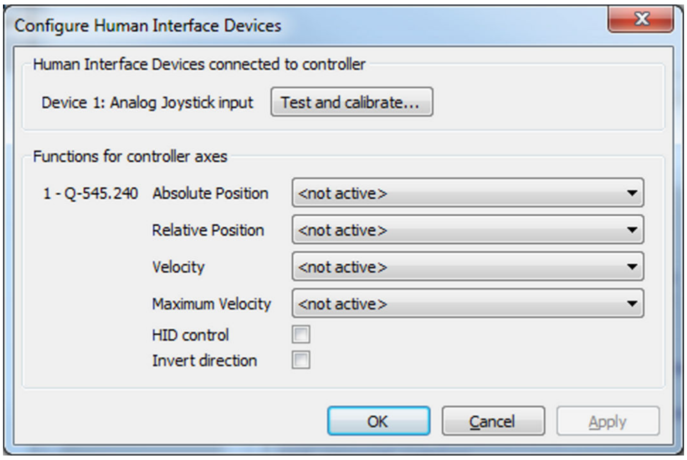
Axis 4 - AB2: 0,000000

4. Depending on what is next, do the following:
   - If you want to calibrate the HID's axes directly afterwards, proceed as described below (p. 93).
   - If you want to set up and activate HID control for the E-873.1AT directly afterwards, close the ***Test and Calibrate HIDevice*** window with ***Close*** and continue as described in "Setting up and Activating HID Control (p. 95)".
   - If you do not want to make any further settings at this point, close ***Test and Calibrate HIDevice*** with ***Close*** and the ***Configure Human Interface Devices*** window with ***OK***.

Calibrating HID Axes

1. If necessary, open the window for configuring HID control via the ***E-873.1AT > Configure controller HIDevice(s)…*** menu item in PIMikroMove's main window.

➔ *The* ***Configure Human Interface Devices*** *window opens.*


Configure Human Interface Devices
Human Interface Devices connected to controller
Test and calibrate...
Functions for controller axes
OK
Cancel
Apply

Device 1: Analog Joystick input

1 - Q-545.240

Absolute Position <not active>

Relative Position <not active>

Velocity <not active>

Maximum Velocity <not active>

HID control ☐

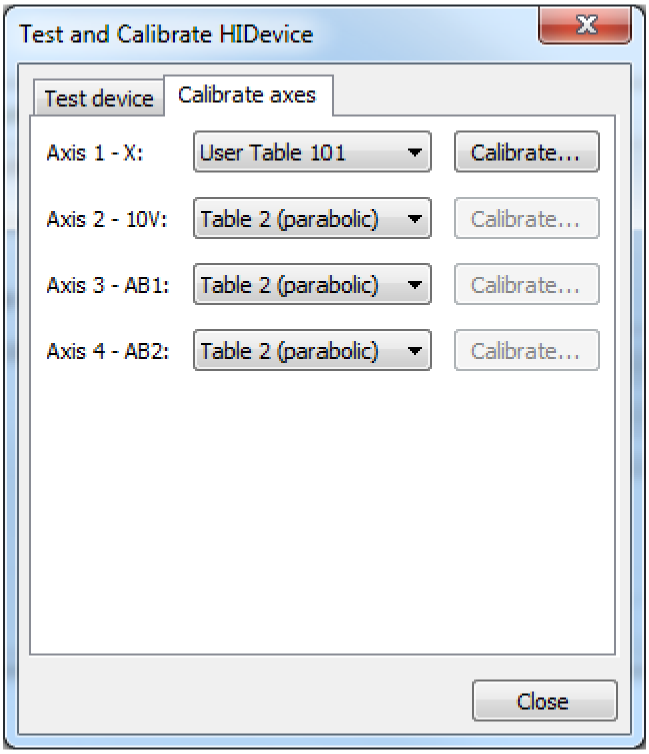
Invert direction ☐

2. If necessary, open the window for calibrating the HID and click the ***Test and calibrate...*** button.

➔ *The* ***Test and Calibrate HIDevice*** *window opens.*

3. Select the ***Calibrate axes*** tab in the ***Test and Calibrate HIDevice*** window.
4. Select the respective lookup tables to be used for the HID's axes via the selection fields in the ***Calibrate axes*** tab.

➔ *The example shows that a user-defined lookup table was selected for axis 1. The respective predefined parabolic lookup table was retained for axes 2 to 4.*



Test and Calibrate HIDevice

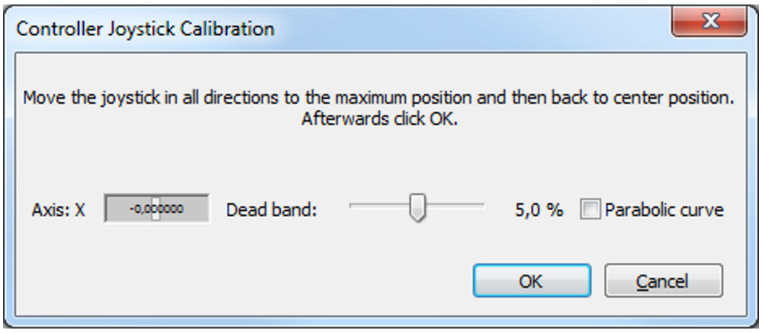
Axis 1 - X: User Table 101 Calibrate...

Axis 2 - 10V: Table 2 (parabolic) Calibrate...

Axis 3 - AB1: Table 2 (parabolic) Calibrate...

Axis 4 - AB2: Table 2 (parabolic) Calibrate...

5. If you have selected a user-defined lookup table and want to fill the table with values:
   a) Click the corresponding ***Calibrate...*** button to open the ***Controller Joystick Calibration*** window.



Controller Joystick Calibration

Move the joystick in all directions to the maximum position and then back to center position. Afterwards click OK.

Axis: X -0,000000 Dead band: 5,0 % □ Parabolic curve

   b) Move the HID's axes to all extreme positions. The custom lookup table values are determined in this way.
   c) Let go of the axis.
   d) If you want to change the neutral area of the axis (i.e., the area around the center position of the axis where no change in the controlled motion variable is triggered), set the ***Dead band*** slider accordingly.
   e) If the values in the user-defined lookup table are to describe a parabolic waveform, click the ***Parabolic curve*** checkbox.
   f) Click ***OK*** in the ***Controller Joystick Calibration*** window to copy the appropriate values from the lookup table.

   ➔ *The lookup table values are written to the E-873.1AT's volatile memory. The writing progress is indicated in a separate window. The window for the writing process and the Controller Joystick Calibration window automatically close after the writing process has finished.*
6. Close the ***Test and Calibrate HIDevice*** window with ***Close***.
7. Depending on what is next, do the following:

- If you want to set up and activate HID control for the E-873.1AT directly afterwards, continue as described in "Setting up and Activating HID Control (p. 95)".
- If you want to save the lookup table assignments to the HID's axes and the content of user-defined lookup table to the E-873.1AT's nonvolatile memory directly afterwards, close the ***Configure Human Interface Devices*** window with ***OK*** and continue as described in "Saving the Configuration of HID Control Permanently (p. 96)".
- If you do not want to make any further settings at this point, close the ***Configure Human Interface Devices*** window with ***OK***.

8.5.6 Setting and Activating HID Control

The following motion variables of the E-873.1AT's axes can be controlled via HID:

- Absolute Position - Motion of the axis to an absolute position
- Relative Position - Motion of the axis relative to the current position
- Velocity - Speed for axis motion
- Maximum Velocity - Maximum speed for axis motion

Before activating HID control, the following steps are recommended:

- Testing the HID
- Calibrating the HID axes

Refer to "Testing the HID and Calibrating the Axes (p. 92)" for a description of these steps.

Requirements

✓ If necessary, you have done a successful respective reference move (p. 64) for the axes of the E-873.1AT.

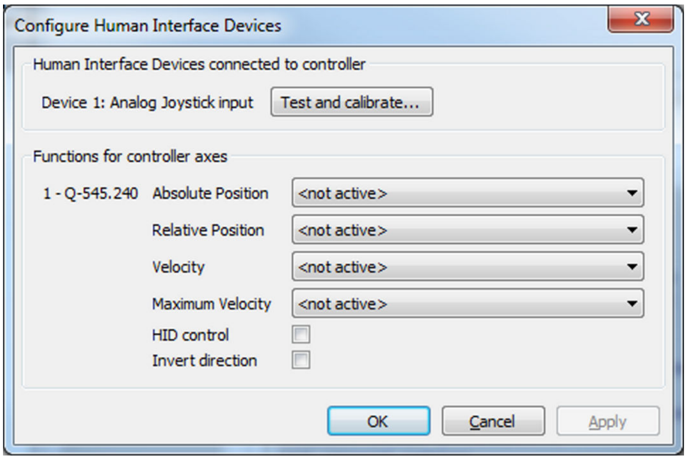
✓ You have connected the HID to the E-873.1AT (p. 91).

✓ All devices are still ready for operation.

Setting Up and Activating HID Control

1. If necessary, open the window for configuring HID control via the ***E-873.1AT > Configure controller HIDevice(s)...*** menu item in PIMikroMove's main window.

➔ *The* ***Configure Human Interface Devices*** *window opens.*


Configure Human Interface Devices

Human Interface Devices connected to controller

Device 1: Analog Joystick input

Functions for controller axes

1 - Q-545.240 Absolute Position <not active>

Relative Position <not active>

Velocity <not active>

Maximum Velocity <not active>

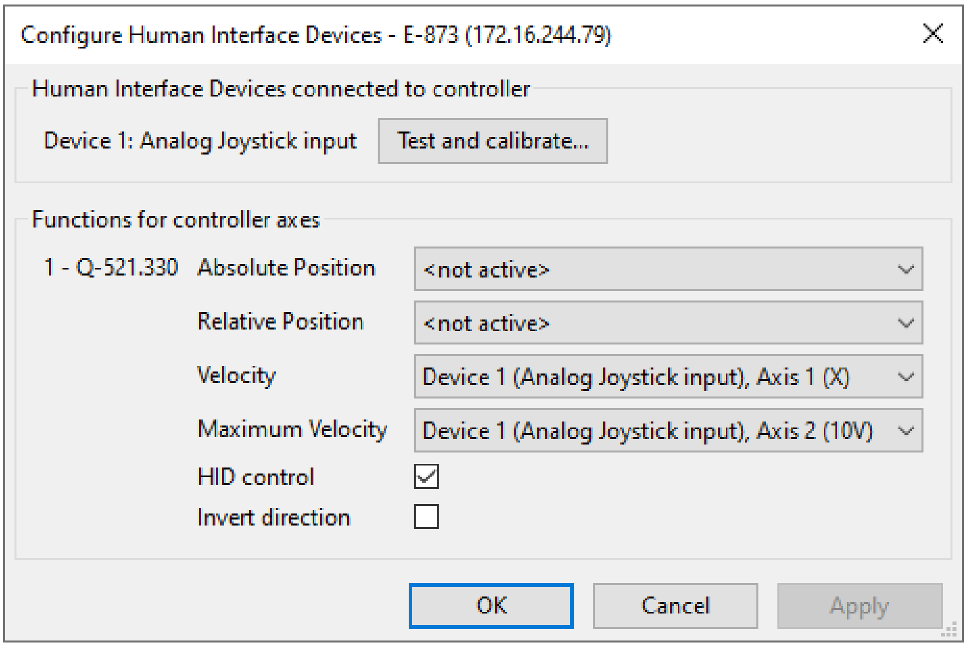
HID control □

Invert direction □

2. Make the following respective settings for the E-873.1AT's axes displayed in the ***Functions for controller axes*** area:
   a) Select the HID's axis in the corresponding field that is to be used for the motion variable to be controlled.
   b) Activate HID control by clicking the ***HID control*** checkbox.

c) If the direction of motion is to be inverted during HID control, click the ***Invert direction*** checkbox.

➔ *In the example shown, the X axis of HID 1 is set and HID control is activated for the controller's axis 1.*



Configure Human Interface Devices - E-873 (172.16.244.79)

Human Interface Devices connected to controller

Device 1: Analog Joystick input

Functions for controller axes

1 - Q-521.330 Absolute Position: <not active>

Relative Position: <not active>

Velocity: Device 1 (Analog Joystick input), Axis 1 (X)

Maximum Velocity: Device 1 (Analog Joystick input), Axis 2 (10V)

HID control ☑

Invert direction □

3. Click the ***Apply*** button in the ***Configure Human Interface Devices*** window to activate the settings.
4. Send the settings for setting up HID control to the E-873.1AT by clicking the ***OK*** button.

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

5. Make sure that servo mode for the E-873.1AT's axes is switched on in PIMikroMove (e.g., by clicking the ***Servo*** checkbox in the ***Axes*** tab in PIMikroMove's main window).

➔ *The E-873.1AT's axes can now be controlled by the HID according to the settings made.*

6. If you want to save the new settings for HID control to the E-873.1AT's nonvolatile memory, continue as described in "Saving the Configuration of HID Control Permanently (p. 96)".

Saving the Configuration of the HID Control Permanently

Select the ***E-873.1AT > Save parameters to non-volatile memory*** menu item in PIMikroMove's main window.

➔ *The **Save Parameters to Non-Volatile Memory** dialog opens.*

1. Enter either the password HID in the selection field of the ***Save Parameters to Non-Volatile Memory*** dialog, or select the ***Settings of HDT, HIA, HIT (HID)*** entry.
2. Click ***OK*** to save and to close the dialog.

8.5.7 Commands

H		Page
HDT	Set HID Default Lookup Table	141
HDT?	Get HID Default Lookup Table	142
HIA	Configure Control Done By HID Axis	143

HIA?	Get Configuration Of Control Done By HID Axis	144
HIB?	Get State Of HID Button	144
HIE?	Get Deflection Of HID Axis	145
HIN	Set Activation State For HID Control	145
HIN?	Get Activation State Of HID Control	145
HIS?	Get Configuration Of HI Device	146
HIT	Fill HID Lookup Table	147
HIT?	Get HID Lookup Table Values	147

8.5.8 Parameters

0x61	Invert Direction Of Motion For Joystick-Controlled Axis?	Inverts the direction of motion for HID-controlled axes. 0 Direction of motion not inverted (default) 1 Direction of motion inverted

8.6 Controller Macros

The E-873.1AT can save and process command sequences as macros.

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

- Several macros can be stored at the same time.
- Any macro can be defined as the startup macro. The startup macro runs each time the E-873.1AT is switched on or rebooted.
- Processing a macro and stopping a macro can be linked to conditions. In this way, loops can be realized as well.
- Macros can call up themselves or other macros at several nesting levels.
- Variables 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.

Working with Macros

- The E-873.1AT can save up to 32 macros simultaneously.
- Up to 10 nesting levels are possible in macros.
- Local and global variables can be used in macros.
- A macro is overwritten if a macro with the same name is rerecorded.
- 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.
- ThePITerminal or the PIMikroMove's ***Command entry*** window can be used for entering commands, e.g., for starting macros.

GCS commands in macros

Basically all GCS commands can be included in a macro. Exceptions:

- `RBT` for rebooting the E-873.1AT
- `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 because macros do not send responses to interfaces.

8.6.1 Recording Macros

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

A macro is overwritten if a macro with the same name is rerecorded.

If you record a macro on a controller whose address differs from 1, pay attention to the following when entering commands that need to be an integral part of the macro:

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

Recording macros for PITerminal and PIMikroMove is described in the following.

1. Start macro recording.
   - If you are working with PITerminal or in the ***Command entry*** in the PIMikroMove's window: Send the `MAC BEG macro name` command where "macro name" is the name of the macro.
   - If you are working in PIMikroMove in the ***Controller macros*** tab: Click the ***Create new empty macro*** icon to create a tab for entering a new macro.
2. Enter the commands to be included in the "macro name macro" line-by-line using the normal command syntax.
   Macros can call up themselves or other macros at several nesting levels.
3. End the macro recording.
   - If you are working with PITerminal or in PIMikroMove's ***Command entry*** window: Send the `MAC END` command.
   - If you are working in PIMikroMove in the ***Controller macros*** tab: Click the ***Send macro to controller*** symbol and enter the macro name into a separate dialog window.

➔ *The macro has been stored in the nonvolatile memory of the E-873.1AT.*

4. If you want to check in PITerminal or in the PIMikroMove's ***Command entry*** window whether the macro was recorded correctly:
   a) Query which macros are saved in the E-873.1AT by sending the `MAC?` command.
   b) Query the content of "macro name" with the `MAC? macro name` command.
5. If you want to check PIMikroMove's ***Controller macros*** tab to see whether the macro was recorded correctly:
   a) Click the ***Read list of macros from controller*** icon.
   b) Mark the macro to be checked in the list on the left-hand side and click the ***Load selected macro from controller*** icon.

Example Macro: Moving the Axis Back and Forth

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

Record the macros by sending:

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

8.6.2 Running the Macros

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

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

You can link the conditions for running the macro with the JRC and WAC commands. The commands must be included in the macro.

A delay time for running the macro can be specified with DEL.

Variables can be used in macros. Setting is done via the VAR command and querying variable values via VAR?. Responses to query commands can be copied into variables with CPY and values can be added and saved to variables with ADD.

In the following, PITerminal or PIMikroMove's ***Command entry*** window is used to enter commands.

1. If the macro should continue running despite an error: Set the Ignore Macro Error? parameter (0x72) accordingly: Send the `SPA 1 0x72 Status` command where "Status" can take the value 0 (stop macro on error [standard]) or 1 (ignore macro error).
2. Start the macro:
   - If the macro is to be run once, send `MAC START macro name string` where "macro name" is the name of the macro.
   - If the macro is to be run n times, send the `MAC NSTART macro name n string` command where "macro name" is the name of the macro and "n" indicates the number of times to be run.

   Specifying "string" is optional and stands for the values of local variables. The values only need to be specified 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 that the macro is running:
   - Query whether a macro is running on the controller by sending the #8 command.
   - Query the name of the macro currently running on the controller by sending the RMC? command.

Example Macro: Moving an Axis with Variable Travel Range Back and Forth

The axis <axis> 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 macros used. Local and global variables are therefore used.

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 that the E-873.1AT is switched on or rebooted, since they are only written to the volatile memory of the E-873.1AT.

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 <axis> $1
WAC ONT? <axis> = 1
MAC END
```

MOVWAI moves axis <axis> to the target position which is specified 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 <axis> alternately moves to the positions 5 and 15 five times. You can also select any other value for the number of executions.

Example macro: Realizing multiple calls of a macro in a loop

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). A further macro is recorded with a loop so that the TESTDION macro does not have to be called separately for each input line.

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 line pointer subsequently jumps 2 lines back, so that TESTDION can be started for the next digital input line.

8.6.3 Stopping the Macros

Stopping the macro can be linked to a condition with the MEX command. The command must be included in the macro.

1. Stop the macro with the #24 or STP commands.
2. If you want to check whether an error occurred while the macro was running, send the MAC ERR? command. The response shows the last error that has occurred.

8.6.4 Configuring a Startup Macro

Any macro can be defined as the startup macro. The startup macro runs each time the E-873.1AT is switched on or rebooted.

Deleting a macro does **not** delete its selection as a startup macro.

1. Set a macro as the startup macro:
   - Send the `MAC DEF macro name` command to set a macro as startup macro where "macro name" is 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, send `MAC DEF` only.
2. Query the name of the currently defined startup macro by sending the `MAC DEF?` command.

Example Macro: Preparing an Axis for Closed-Loop Operation via a Startup Macro
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
FRF 5
MAC END
MAC DEF startcl
```

When this macro is used, the E-873.1AT's parameter settings should be adapted to the connected positioner in the nonvolatile memory. Alternatively, the parameter settings can also be configured in the volatile memory via the startup macro.

8.6.5 Deleting Macros

A macro cannot be deleted while it is running.

Deleting a macro does not delete its selection as a startup macro.

1. Delete a macro with the `MAC DEL macro name` command where "macro name" is the name of the macro.

8.6.6 Saving and Loading Macros

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

The use of the ***Controller macros*** tab in PIMikroMove is recommended for backing up and loading controller macros.

Saving and loading controller macros with PIMikroMove is described in the following.

Saving Controller Macros on the PC

1. Select the ***Controller macros*** tab in PIMikroMove's 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 a macro.
   - To select more than one macro, press and hold down the shift key and click the desired entries in the list.
   - To deselect, click an open area in the list.

➔ *The **Save selected macros to PC** button becomes active when selecting one or more macros.*

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) in the selected directory of the PC.*

Loading Controller Macros from the PC to the E-873.1AT

1. Select the ***Controller macros*** tab in PIMikroMove's main window.
2. Load macros from the PC to the E-873.1AT:
   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-873.1AT.
   c) Click ***Open***.

➔ *For each selected text file (<macro name>.txt), the content is loaded as a macro <macro name> into the E-873.1AT.*

8.6.7 Example Macros

Example Macro: Stopping Motion by Pushbutton

1. Connect digital input line 1 on the *I/O* socket to an appropriate signal source.
   The digital input signal can be used e. g. for a conditional jump of the macro execution pointer.
2. Record the HALT macro on the controller.

```
MAC BEG halt
MVR 1 5
JRC 2 DIO? 1 = 1
JRC -1 ONT? 1 = 0
HLT 1
MAC END
```

➔ *The macro has the following tasks:*

Start relative motion of axis 1

Set condition: If digital input line 1 has the high state, the macro execution pointer jumps two lines forward. This stops the axis. Otherwise, the macro continues to the next line.

Set condition: As long as axis 1 has not yet reached the target position, the macro execution pointer jumps back one line. A loop is established as a result.

3. Start the HALT macro on the controller.

```
MAC START halt
```

➔ *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* `HLT` *command.*

4. If error code 10 interferes: Record alternative HALTVAR macro which uses a variable.

```
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
```

➔ *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.*

5. Start the HALTVAR macro on the controller.

```
MAC START haltvar
```

➔ *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.*

Example Macro: HID Control Alternating with Relative Motion

The identifier of axis 1 was changed in X with the `SAI command`. The absolute target position of axis X is to be controlled by axis 1 of the HID (e.g., a joystick axis). The buttons of a connected C-170.PB pushbutton box are to be used for the following tasks:

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

1. Connect C-170.PB pushbutton box from PI to the ***I/O*** socket.

➔ *Digital input lines 1 to 4 are switched to high state as long as the respective button is pressed.*

2. Connect C-819.20 or C-819.30 joystick to the ***Joystick*** socket.

➔ *For commands, the connected joystick axis is accessible as axis 1 HID 1.*

3. Record the STARTUP macro on the controller.

```
MAC BEG startup
HIN X 0
SVO X 1
FRF X
WAC ONT? X = 1
HIA X 0 0 0
HIA X 1 1 1
HIN X 1
MAC START LOOP
MAC END
```

The macro has the following tasks:

- Switch on the servo mode for axis X.
- Start a reference move for axis X.
- Configure HID control for axis X: The absolute target position is to be controlled by axis 1 of HID 1.
- Activate HID control for axis X
- Start the LOOP macro for the main loop

4. Record the LOOP macro on the controller.

```
MAC BEG loop
MAC START button3
MAC START loop
MAC END
```

The macro has the following tasks:

- Start BUTTON3 macro
- Call itself to set up the main loop

5. Record the BUTTON3 macro on the controller.

```
MAC BEG button3
MEX DIO? 3 = 0
HIN X 0
MAC START pbloop
MAC END
```

The macro has the following tasks:

- If button 3 is not pressed: End the execution of BUTTON3
- If button 3 is pressed: Disable HID control and start the loop for checking buttons 1, 2 and 4

6. Record the PBLOOP macro on the controller.

   ```
   MAC BEG pbloop
   MAC START button1
   MAC START button2
   MAC START button4
   MAC START pbloop
   MAC END
   ```

   The macro has the following tasks:

   - Start the BUTTON1, BUTTON2 and BUTTON4 macros in succession
   - Call itself up to create the loop to check buttons 1, 2 and 4

7. Record the BUTTON1 macro on the controller.

   ```
   MAC BEG button1
   MEX DIO? 1 = 0
   MVR X 1
   WAC ONT? X = 1
   MAC END
   ```

   The macro has the following tasks:

   - If button 1 is not pressed: End the execution of BUTTON1
   - If button 1 is pressed: Start a motion of axis X over distance 1 in the positive direction and pause the macro execution until axis X is at the target position

8. Record the BUTTON2 macro on the controller.

   ```
   MAC BEG button2
   MEX DIO? 2 = 0
   MVR X -1
   WAC ONT? X = 1
   MAC END
   ```

   The macro has the following tasks:

   - If button 2 is not pressed: End the execution of BUTTON2
   - If button 2 is pressed: Start a motion of axis X over distance 1 in the negative direction and pause the macro execution until axis X is at the target position

9. Record the BUTTON4 macro on the controller.

   ```
   MAC BEG button4
   MEX DIO? 4 = 0
   HIN X 1
   MAC START LOOP
   MAC END
   ```

   The macro has the following tasks:

   - If button 4 is not pressed: End the execution of BUTTON4
   - If button 4 is pressed: Enable HID control and start the main loop

10. Start the STARTUP macro on the controller.

    ```
    MAC START startup
    ```

Axis X starts a reference move to the reference switch. HID control is then enabled for axis X, so that the absolute target position can be controlled with the joystick. As long as HID control is enabled, buttons 1 and 2 have no effect. HID control is disabled by pressing button 3. Relative motion of axis X can then be started with buttons 1 and 2, and HID control can be enabled again with button 4.

8.6.8 Commands

#		Page
#24	Stop All Axes	123
#8	Query If Macro Is Running	123
A		Page
ADD	Add And Save To Variable	124
C		Page
CPY	Copy Into Variable	125
D		Page
DEL	Delay The Command Interpreter	130
J		Page
JRC	Jump Relatively Depending On Condition	154
M		Page
MAC BEG	Call Macro Function: BEG	155
MAC DEF	Call Macro Function: DEF	155
MAC DEF?	Call Macro Function: DEF?	155
MAC DEL	Call Macro Function: DEL	156
MAC END	Call Macro Function: END	156
MAC ERR?	Call Macro Function: ERR?	156
MAC NSTART	Call Macro Function: NSTART	156
MAC START	Call Macro Function: START	157
MAC?	List Macros	157
MEX	Stop Macro Execution Due To Condition	159
R		Page
RBT	Reboot System	163
RMC?	List Running Macros	164
S		Page
STP	Stop All Axes	172
V		Page
VAR	Set Variable Value	176
VAR?	Get Variable Values	176

W		Page
WAC	Wait For Condition	178

8.6.9 Parameters

0x72	Ignore Macro Error?	Ignore macro error? 0 Stop macro on error (default) 1 Ignore error

8.7 Working with GCS Commands

8.7.1 GCS Command Syntax

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 repeated specifications, i.e., it is possible to access more than one element (e.g., several axes) in one command line.
␊	Line Feed (ASCII character 10), default termination character (character at the end of a command line)
␣	Space (ASCII character 32), empty space
⇥	Horizontal tab (ASCII character 9)
#…	Single-character command, "…" indicates the ASCII character in decimal notation, e.g., #7 for ASCII character ␇.

Syntax

A GCS command consists of three letters, e.g., `CMD`, or three letters and a question mark, e.g., `CMD?`.

Exceptions:

- Single-character commands consist of only one ASCII character. In this manual, 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 and channel identifiers, parameters etc.) must be separated from each other by a space (␣). The command line ends with the termination character (␊).

- `CMD[{{␣}<Argument>}]␊`
- `CMD?[{{␣}<Argument>}]␊`

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.

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

When all arguments are optional and not specified, the command is executed for all possible argument values.

Sending a Command

The axis identified with "1" is to be moved to position 10.0. The unit depends on the controller (e.g., µm or mm).

1. Send
   ```
   MOV␣1␣10.0␊
   ```

➔ *Axis 1 moves to position 10.0 (physical unit).*

Sending the Command with Several Arguments

Two axes are to be moved that are connected to the same controller: The axis with axis identifier "1" is to be moved to position 17.0 and the axis with axis identifier "2" is to be moved to position 2.05. The unit depends on the controller (e.g., µm or mm).

Information

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

1. Send
   ```
   MOV␣1␣17.3␣2␣2.05␊
   ```

➔ *Axis 1 moves to position 17.0 (physical unit), axis 2 moves to position 2.05 (physical unit).*

Sending commands without arguments

The position of all axes is to be queried.

1. Send
   ```
   POS?␊
   ```

➔ *Outputs the position of all axes. The response syntax is as follows:*

`{[<Argument>[{␣<Argument>}]"="]<Wert>␣␊}`

`[<Argument>[{␣<Argument>}]"="]<Wert>␊` *(for the last line)*

8.7.2 Variables

The electronics support variables for more flexible programming. While global variables are always available, local variables are only valid for a specified macro. Typically, variables are used when working with macros.

Variables are in volatile memory (RAM) only. The variable values are 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 begin with a letter.
- Names of local variables must not contain alphabetic characters. Possible characters are 0 to 9.
- The variable name can also be specified 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, no braces are necessary.

Note that when braces are omitted for multicharacter variable names, the first character after the “$” is interpreted as the variable name.

8.8 Adapting Settings

The properties of the E-873.1AT and the mechanics connected are stored in the E-873.1AT as parameter values.

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

Write permission for the parameters is determined by command levels.

Each parameter is in the E-873.1AT's volatile and nonvolatile memory. The values in the nonvolatile memory are loaded to the volatile memory as default values when switching on or rebooting the E-873.1AT. 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.

Use the HPA? command to query the parameters available in the E-873.1AT.

⚠ *WARNING*

Risk of electric shock due to 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-873.1AT and take effect immediately. Unsuitable parameter settings can lead to a piezo voltage that is too high for the connected positioner.

If the piezo voltage is too high touching the positioner can cause serious injury due to electric shock. Also it can cause damage to the positioner.

► Change parameter values only after careful consideration.

► Save the current parameter values to the PC before you make changes in the nonvolatile memory.

8.8.1 Parameter Commands

The following general commands are available for changing parameters:

Command	Function
CCL	Change to another command level
CCL?	Query active command level
HPA?	Query available controller parameters
RPA	Copy a parameter value from the nonvolatile to the volatile memory
SEP	Modify parameter value in nonvolatile memory
SEP?	Query parameter values from the nonvolatile memory
SPA	Modify parameter value in volatile memory
SPA?	Query 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.

The following special commands only change the corresponding parameters in the volatile memory. When necessary, the changed values must be written to the nonvolatile memory with the WPA command.

Command	Adaptable parameters
ACC	Acceleration in closed-loop operation (0xB)
DEC	Deceleration in closed-loop operation (0xC)
VEL	Velocity in closed-loop operation (0x49)

The values for velocity (0x49), acceleration (0xB), and deceleration (0xC) are not evaluated for fast step mode in PiezoMike mode.

8.8.2 Saving Parameter Values in a Text File

Overview

The E-873.1AT is configured via parameters, e.g., for adapting to the mechanics. Changing parameter values can cause undesirable results.

► Create a backup copy on the PC before changing the parameter settings of the E-873.1AT. 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-873.1AT to specific mechanics.

Parameter values saved in a text file on the PC can be loaded back to the E-873.1AT in PIMikroMove or PITerminal. The ***Send file...*** button is available for this purpose in the send command window. Before loading into the E-873.1AT, 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 between the E-873.1AT and the PC (p. 59) with PIMikroMove or PITerminal.

Saving Parameter Values in a Text File

1. If you are using PIMikroMove, open the window for transmitting commands: Select ***Tools > Command entry*** in the main window or press F4 on the keyboard.
   After communication has been established, the main window is opened PITerminal automatically and commands can be sent.
2. Query the parameter values that you want to create a backup.
   - If you want to save the parameter values from the volatile memory of the E-873.1AT: Send the SPA? command.
   - If you want to save the parameter values from the nonvolatile memory of the E-873.1AT: Send the SEP? command.
3. Click the ***Save...*** button.
4. The ***Save content of terminal as textfile*** window opens.
5. Save the queried parameter values to a text file on your PC in the ***Save content of terminal as textfile*** window.

8.8.3 Changing Parameter Values

Overview

The following procedure is generally recommended for changing parameter values:

1. Change the parameter values in the volatile memory.

2. Check whether the E-873.1AT 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.

Requirements

✓ If you want to change parameter values in the E-873.1AT's nonvolatile memory: You have saved the E-873.1AT's parameter values in a text file on the PC (p. 109).

✓ You have established communication between the E-873.1AT and the PC (p. 59) with PIMikroMove or PITerminal.

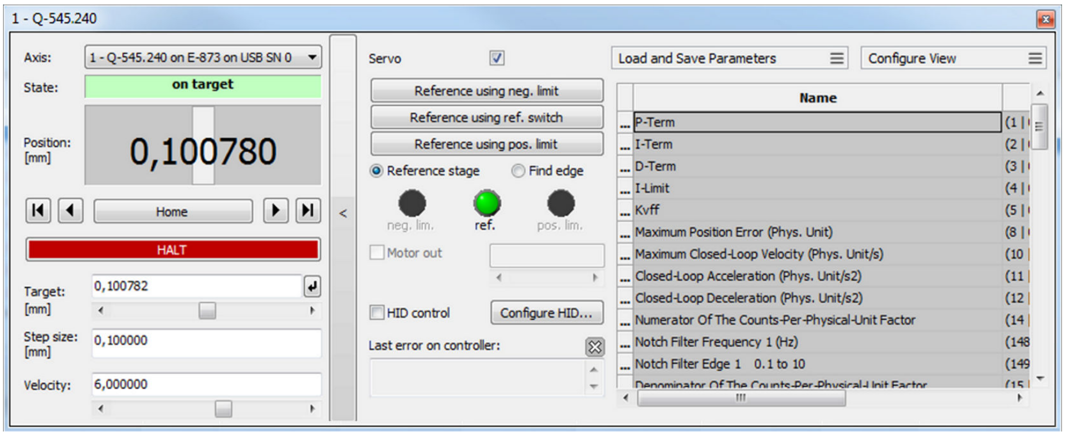
Information

Write access for the parameters of the E-873.1AT 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.

Changing Parameter Values

1. If you want to change the axis-related parameters of the E-873.1AT:
   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.

➔ *The list of axis-related parameters is displayed.*



1 - Q-545.240

Axis: 1 - Q-545.240 on E-873 on USB SN 0

State: on target

Position: [mm] 0,100780

Target: [mm] 0,100782

Step size: [mm] 0,100000

Velocity: 6,000000

Servo ☑

◉ Reference stage ○ Find edge

□ Motor out

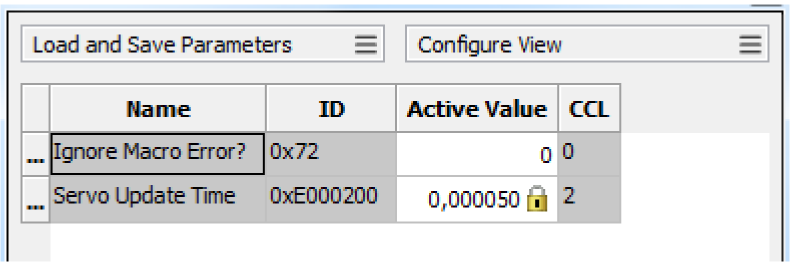
□ HID control

Last error on controller:

Name	
P-Term	(1 |
I-Term	(2 |
D-Term	(3 |
I-Limit	(4 |
Kvff	(5 |
Maximum Position Error (Phys. Unit)	(8 |
Maximum Closed-Loop Velocity (Phys. Unit/s)	(10
Closed-Loop Acceleration (Phys. Unit/s2)	(11
Closed-Loop Deceleration (Phys. Unit/s2)	(12
Numerator Of The Counts-Per-Physical-Unit Factor	(14
Notch Filter Frequency 1 (Hz)	(148
Notch Filter Edge 1 0.1 to 10	(149
Denominator Of The Counts-Per-Physical-Unit Factor	(15

2. If you want to change the system-related parameters of the E-873.1AT:
   a) Open the window for the system-related parameters of the E-873.1AT in the main window of PIMikroMove by selecting *E-873.1AT> Show system parameters* in the menu.

➔ *The list of system-related parameters is displayed.*



Load and Save Parameters ☰ Configure View ☰

	Name	ID	Active Value	CCL
...	Ignore Macro Error?	0x72	0	0
...	Servo Update Time	0xE000200	0,000050 🔒	2

3. If you want to change the parameter values in the E-873.1AT's **volatile memory**, you have the following options:
   a) Type the new parameter values into the corresponding input field in the *Active Value* column and press the enter key on the PC's keyboard or click the mouse button outside of the input field.
   ➔ *The modified parameter value is transferred to the E-873.1AT's volatile memory.*
   b) Click *Load and Save Parameters -> Load all startup parameters of the axis / system from controller*.
   ➔ *The values of all axis-related / system-related parameters are loaded from the E-873.1AT's nonvolatile memory.*
   c) Click *Load and Save Parameters > Load parameters from stage database...* in the expanded single axis window.

   You can use ***Load and Save Parameters > Reload parameters from stage database...*** to reload the currently loaded parameter set.

   ➔ *A selected parameter set for the axis is loaded from the positioner database.*
4. If you want to change parameter values in the E-873.1AT's **nonvolatile memory**, you have the following options:
   a) Type the new parameter value into the corresponding input field in the list's *Startup Value* column and press the enter key on the PC's keyboard or click the mouse button outside of the input field.
   ➔ *The changed parameter value is transferred to the E-873.1AT's volatile memory.*
   b) Click *Load and Save Parameters -> Save all currently active axis / system parameters as startup parameters to controller*.

   You can skip parameters that do not have write access on the current command level.

   ➔ *The values of all axis-related / system-related parameters are written from the E-873.1AT's volatile memory to the nonvolatile memory.*

   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.

8.8.4 Creating or Changing Parameter Sets for Positioners

Overview

You can create and edit new parameter records in the PIStages3 database. This can be required in the following cases, for example:

- You want to operate a positioner with different servo control parameter settings than the one from the default parameter set.
- You want to adapt the soft limits of the positioner to your application.
- You have a custom positioner.

Possibilities for creating and editing parameter sets in the PISTAGES3.DB database:

- You can create a new positioner type easily by changing an existing parameter set in PIMikroMove and saving it under a new name.
- You can open and edit the positioner database directly with the PIStages3Editor.

The following describes how to use PIMikroMove to create and change a parameter set for a positioner.

Requirements

✓ You have installed the latest version of the PISTAGES3.DB database onto your PC.

✓ If PI provided a custom positioner database for your positioner, the dataset was imported into PIStages3.

✓ PIMikroMove has established communication between the E-873.1AT and the PC.

Creating a Parameter Set for Positioners

1. Select the ***E-873.1AT > Select connected stages...*** menu item in the main window of PIMikroMove.

➔ *The* ***Start up stages/axes for E-873.1AT*** *window opens and the* ***Select connected stages*** *step is active.*

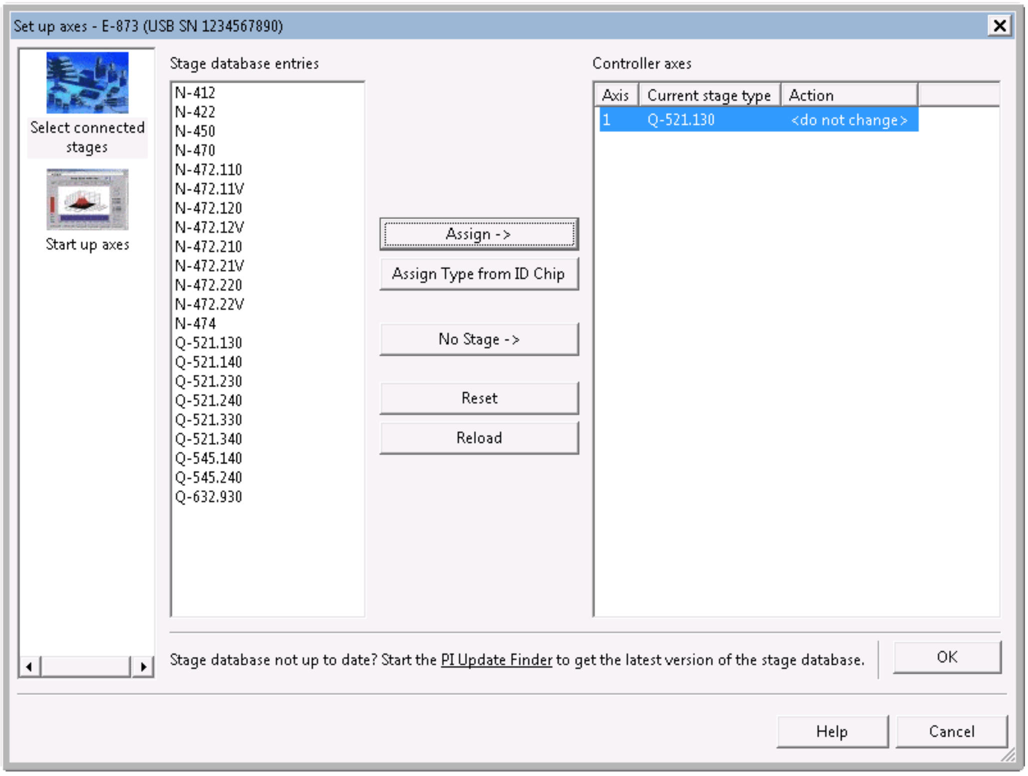
2. Select an appropriate type of positioner during the ***Select connected stages*** step:

   a) Click ***Assign Type from ID Chip***.

   or

   a) Highlight the positioner in the ***Stage database entries*** list.

   b) Click ***Assign***.



Set up axes - E-873 (USB SN 1234567890)

Stage database entries

N-412
N-422
N-450
N-470
N-472.110
N-472.11V
N-472.120
N-472.12V
N-472.210
N-472.21V
N-472.220
N-472.22V
N-474
Q-521.130
Q-521.140
Q-521.230
Q-521.240
Q-521.330
Q-521.340
Q-545.140
Q-545.240
Q-632.930

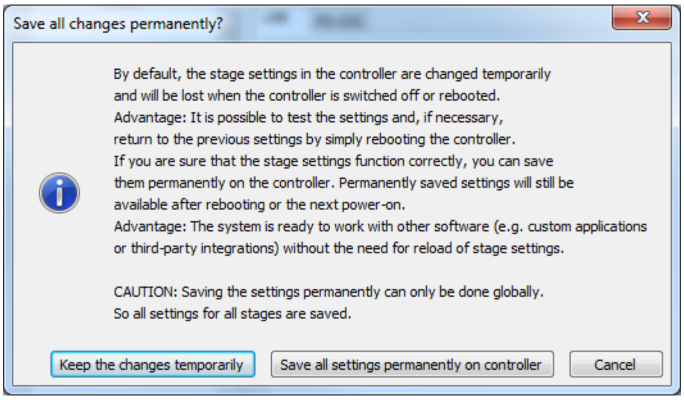
Controller axes

Axis	Current stage type	Action
1	Q-521.130	<do not change>

Stage database not up to date? Start the PI Update Finder to get the latest version of the stage database.

   c) Confirm the selection with ***OK***.

➔ *The* ***Save all changes permanently?*** *dialog is opened.*



Save all changes permanently?

By default, the stage settings in the controller are changed temporarily and will be lost when the controller is switched off or rebooted.
Advantage: It is possible to test the settings and, if necessary, return to the previous settings by simply rebooting the controller.
If you are sure that the stage settings function correctly, you can save them permanently on the controller. Permanently saved settings will still be available after rebooting or the next power-on.
Advantage: The system is ready to work with other software (e.g. custom applications or third-party integrations) without the need for reload of stage settings.

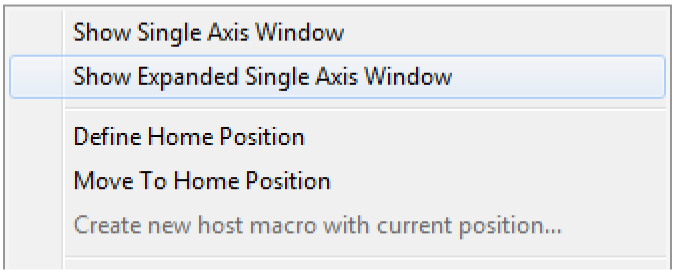
CAUTION: Saving the settings permanently can only be done globally.
So all settings for all stages are saved.

Keep the changes temporarily
Save all settings permanently on controller
Cancel

3. Click ***Keep the changes temporarily*** in the ***Save all changes permanently*** dialog to load the parameter settings into the volatile memory of the E-873.1AT.

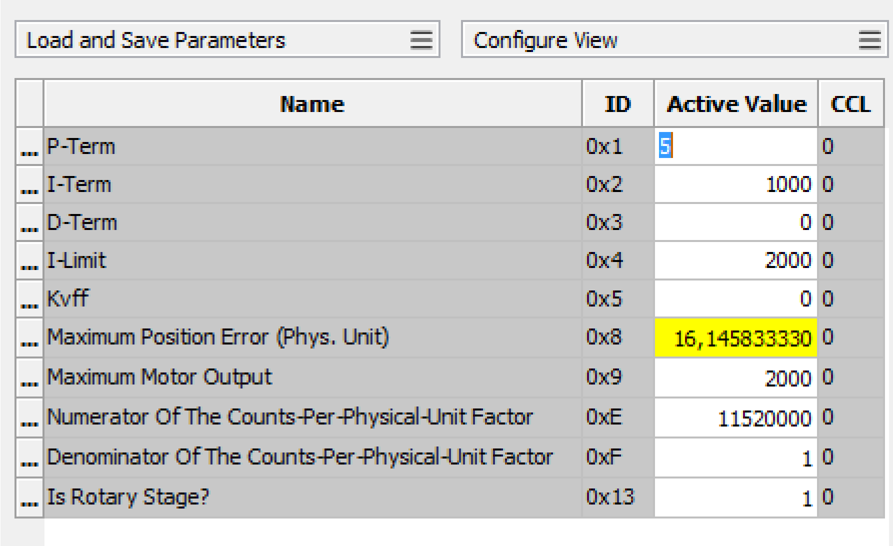
➔ *The **Start up stages/axes** window changes to the **Start up axes** step.*

4. Click ***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.



Show Single Axis Window
Show Expanded Single Axis Window
Define Home Position
Move To Home Position
Create new host macro with current position...

6. Enter new values for the parameters to be changed:



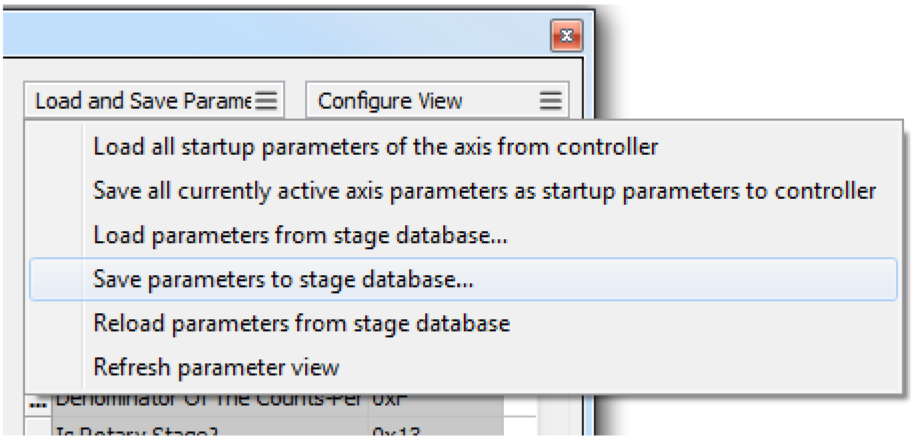
	Name	ID	Active Value	CCL
...	P-Term	0x1	5	0
...	I-Term	0x2	1000	0
...	D-Term	0x3	0	0
...	I-Limit	0x4	2000	0
...	Kvff	0x5	0	0
...	Maximum Position Error (Phys. Unit)	0x8	16,145833330	0
...	Maximum Motor Output	0x9	2000	0
...	Numerator Of The Counts-Per-Physical-Unit Factor	0xE	11520000	0
...	Denominator Of The Counts-Per-Physical-Unit Factor	0xF	1	0
...	Is Rotary Stage?	0x13	1	0

a) 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 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.

7. Click ***Load and Save Parameters > Save parameters to stage database...***.


Load and Save Parame
Configure View
Load all startup parameters of the axis from controller
Save all currently active axis parameters as startup parameters to controller
Load parameters from stage database...
Save parameters to stage database...
Reload parameters from stage database
Refresh parameter view

8. The ***Save Parameters as User Stage Type*** dialog opens.
9. Save the changed parameter values as new positioner type in ***Dialog Save Parameters as User Stage Type***:
   a) Leave the entry in the ***Parameters of axis*** field unchanged.
   b) Enter the name for the new positioner type into the ***Save as*** field.
   c) Click ***OK***.

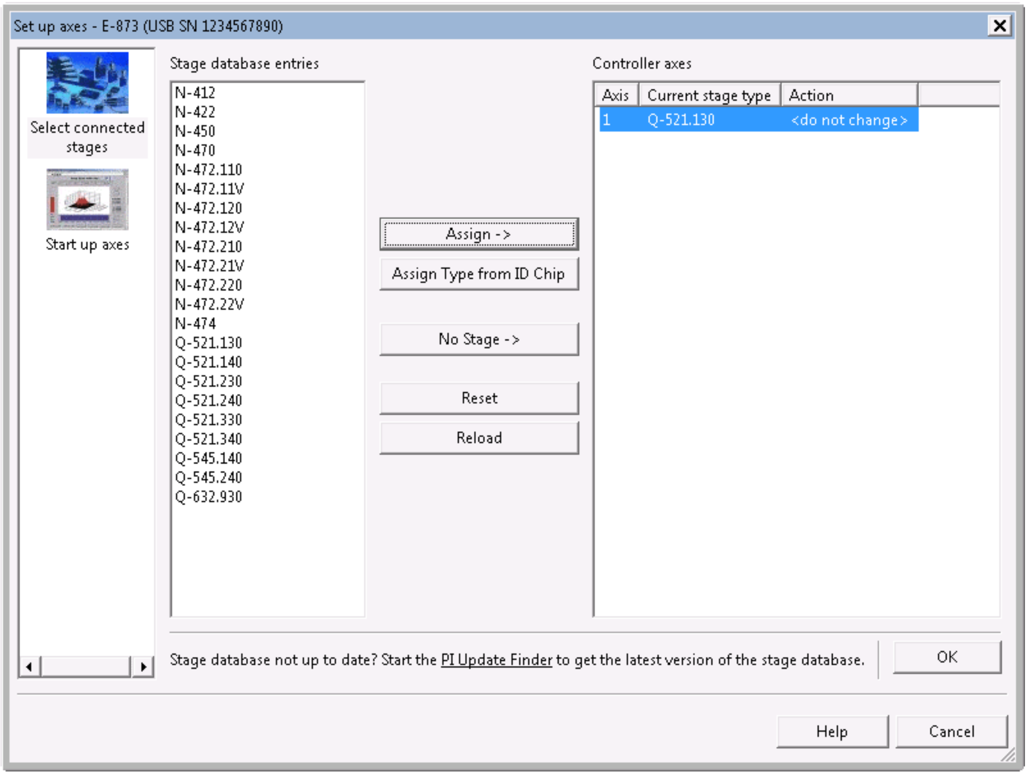
➔ *The new positioner type was saved in the positioner database. The displayed positioner type was updated in PIMikroMove. The new positioner type is also available immediately for selection in the* ***Select connected stages*** *step.*

Changing a Positioner's Parameter Set

1. Select the ***E-873.1AT > Select connected stages...*** menu item in the main window of PIMikroMove.

➔ *The* ***Start up stages/axes for E-873.1AT*** *window opens and the* ***Select connected stages*** *step is active.*

2. Select a positioner type that you created during the ***Select connected stages*** as described above:
   a) Click ***Assign Type from ID Chip***.
   or
   a) Highlight the positioner in the ***Stage database entries*** list.
   b) Click ***Assign***.



Stage database entries

N-412
N-422
N-450
N-470
N-472.110
N-472.11V
N-472.120
N-472.12V
N-472.210
N-472.21V
N-472.220
N-472.22V
N-474
Q-521.130
Q-521.140
Q-521.230
Q-521.240
Q-521.330
Q-521.340
Q-545.140
Q-545.240
Q-632.930

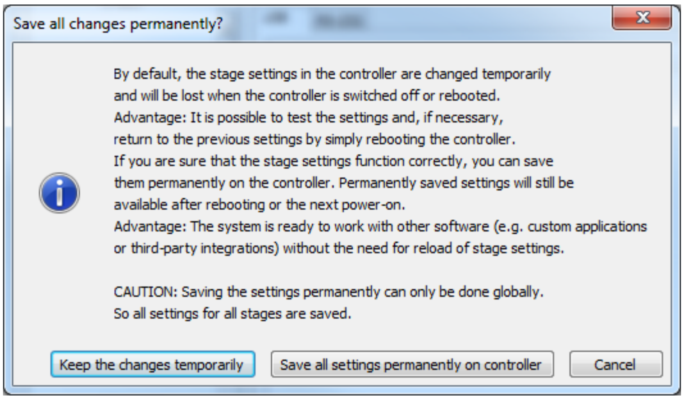
Controller axes

Axis	Current stage type	Action
1	Q-521.130	<do not change>

Stage database not up to date? Start the PI Update Finder to get the latest version of the stage database.

c) Confirm the selection with ***OK***.

➔ *The* ***Save all changes permanently?*** *dialog is opened.*


Save all changes permanently?
Keep the changes temporarily
Save all settings permanently on controller
Cancel

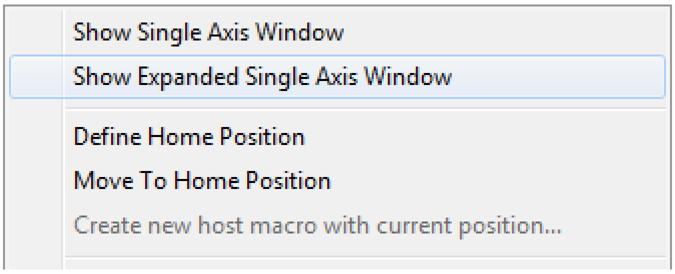
By default, the stage settings in the controller are changed temporarily
and will be lost when the controller is switched off or rebooted.
Advantage: It is possible to test the settings and, if necessary,
return to the previous settings by simply rebooting the controller.
If you are sure that the stage settings function correctly, you can save
them permanently on the controller. Permanently saved settings will still be
available after rebooting or the next power-on.
Advantage: The system is ready to work with other software (e.g. custom applications
or third-party integrations) without the need for reload of stage settings.

CAUTION: Saving the settings permanently can only be done globally.
So all settings for all stages are saved.

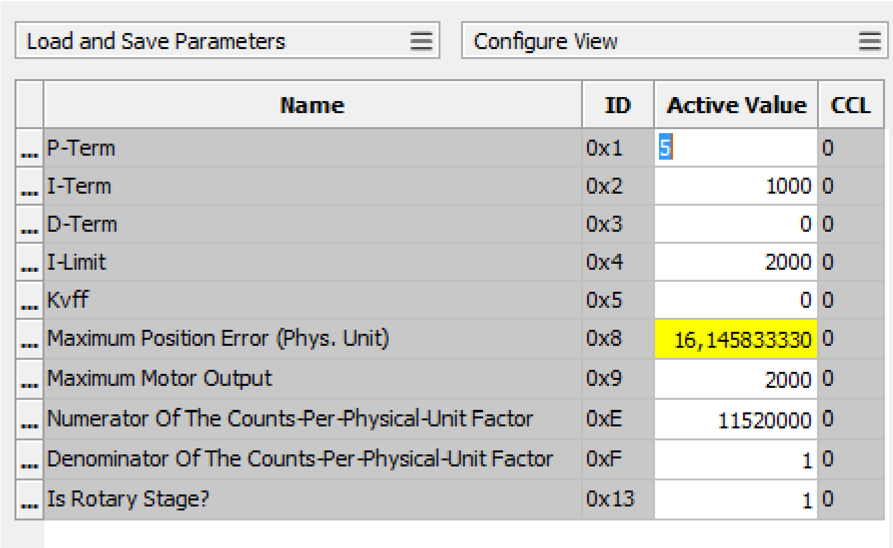
3. Click ***Keep the changes temporarily*** in the ***Save all changes permanently*** dialog to load the parameter settings into the volatile memory of the E-873.1AT.

➔ *The* ***Start up stages/axes*** *window changes to the* ***Start up axes*** *step.*

4. Click ***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.


Show Single Axis Window
Show Expanded Single Axis Window
Define Home Position
Move To Home Position
Create new host macro with current position...

6. Enter new values for the parameters to be changed:



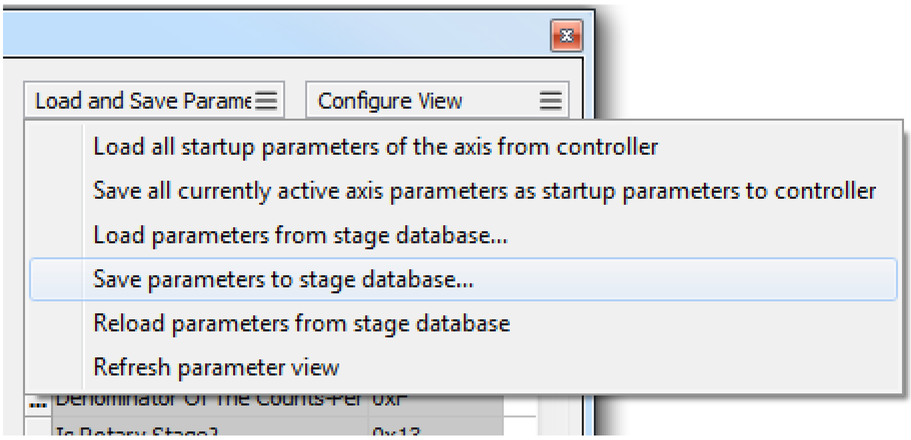
	Name	ID	Active Value	CCL
...	P-Term	0x1	5	0
...	I-Term	0x2	1000	0
...	D-Term	0x3	0	0
...	I-Limit	0x4	2000	0
...	Kvff	0x5	0	0
...	Maximum Position Error (Phys. Unit)	0x8	16,145833330	0
...	Maximum Motor Output	0x9	2000	0
...	Numerator Of The Counts-Per-Physical-Unit Factor	0xE	11520000	0
...	Denominator Of The Counts-Per-Physical-Unit Factor	0xF	1	0
...	Is Rotary Stage?	0x13	1	0

a) 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 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.

7. Click ***Load and Save Parameters > Save parameters to stage database...***.


Load and Save Parame
Configure View
Load all startup parameters of the axis from controller
Save all currently active axis parameters as startup parameters to controller
Load parameters from stage database...
Save parameters to stage database...
Reload parameters from stage database
Refresh parameter view
Denominator Of The Counts-Per 0xF
Is Rotary Stage? 0x13

8. The ***Save Parameters as User Stage Type*** dialog opens.

9. 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 were updated in the positioner database and in the main window of PIMikroMove.*

9 Command Reference

#		Page
#24	Stop All Axes	123
#4	Request Status Register	121
#5	Request Motion Status	122
#7	Request Controller Ready Status	122
#8	Query If Macro Is Running	123
*		Page
*IDN?	Get Device Information	123
A		Page
ACC	Set Closed-Loop Acceleration	124
ACC?	Get Closed-Loop Acceleration	124
ADD	Add And Save To Variable	124
C		Page
CCL	Set Command Level	125
CCL?	Get Command Level	125
CPY	Copy Into Variable	125
CST?	Get Assignment Of Stages To Axes	126
CSV?	Get Current Syntax Version	126
CTO	Set Configuration Of Trigger Output	127
CTO?	Get Configuration Of Trigger Output	129
D		Page
DEC	Set Closed-Loop Deceleration	129
DEC?	Get Closed-Loop Deceleration	130
DEL	Delay The Command Interpreter	130
DFH	Define Home Position	131
DFH?	Get Home Position Definition	131
DIO	Set Digital Output Lines	132
DIO?	Get Digital Input Lines	132
DRC	Set Data Recorder Configuration	133
DRC?	Get Data Recorder Configuration	134
DRL?	Get Number Of Recorded Points	134
DRR?	Get Recorded Data Values	134
DRT	Set Data Recorder Trigger Source	135
DRT?	Get Data Recorder Trigger Source	136
E		Page
ERR?	Get Error Number	136

F		Page
FED	Find Edge	137
FNL	Fast Reference Move To Negative Limit	138
FPL	Fast Reference Move To Positive Limit	138
FRF	Fast Reference Move To Reference Switch	139
FRF?	Get Referencing Result	139
G		Page
GOH	Go To Home Position	140
H		Page
HAR?	Indicate Hard Stops	140
HDR?	Get All Data Recorder Options	141
HDT	Set HID Default Lookup Table	141
HDT?	Get HID Default Lookup Table	142
HIA	Configure Control Done By HID Axis	143
HIA?	Get Configuration Of Control Done By HID Axis	144
HIB?	Get State Of HID Button	144
HIE?	Get Deflection Of HID Axis	145
HIN	Set Activation State For HID Control	145
HIN?	Get Activation State Of HID Control	145
HIS?	Get Configuration Of HI Device	146
HIT	Fill HID Lookup Table	147
HIT?	Get HID Lookup Table Values	147
HLP?	Get List Of Available Commands	147
HLT	Halt Motion Smoothly	148
HPA?	Get List Of Available Parameters	148
HPV?	Get Parameter Value Description	149
I		Page
IFC	Set Interface Parameters Temporarily	150
IFC?	Get Current Interface Parameters	151
IFS	Set Interface Parameters As Default Values	152
IFS?	Get Interface Parameters As Default Values	153
J		Page
JRC	Jump Relatively Depending On Condition	154
L		Page
LIM?	Indicate Limit Switches	154
M		Page
MAC	Call Macro Function	155
MAC BEG	Call Macro Function: BEG	155

MAC DEF	Call Macro Function: DEF	155
MAC DEF?	Call Macro Function: DEF?	155
MAC DEL	Call Macro Function: DEL	156
MAC END	Call Macro Function: END	156
MAC ERR?	Call Macro Function: ERR?	156
MAC NSTART	Call Macro Function: NSTART	156
MAC START	Call Macro Function: START	157
MAC?	List Macros	157
MAN?	Get Help String For Command	157
MAT	Calculate And Save To Variable	158
MEX	Stop Macro Execution Due To Condition	159
MOV	Set Target Position	160
MOV?	Get Target Position	160
MVR	Set Target Relative To Current Position	161
O		Page
ONT?	Get On-Target State	161
OSM	Open-Loop Step Moving	162
OSN?	Read Number Steps	162
P		Page
POS	Set Real Position	163
POS?	Get Real Position	163
R		Page
RBT	Reboot System	163
RMC?	List Running Macros	164
RON	Set Reference Mode	164
RON?	Get Reference Mode	165
RPA	Reset Volatile Memory Parameters	165
RTR	Set Record Table Rate	166
RTR?	Get Record Table Rate	166
S		Page
SAI	Set Current Axis Identifiers	166
SAI?	Get List Of Current Axis Identifiers	167
SEP	Set Nonvolatile Memory Parameters	167
SEP?	Get Nonvolatile Memory Parameters	168
SPA	Set Volatile Memory Parameters	168
SPA?	Get Volatile Memory Parameters	169
SRG?	Query Status Register Value	170
SST	Set Step Size	171

SST?	Get Step Size	171
STE	Start Step And Response Measurement	172
STP	Stop All Axes	172
SVO	Set Servo Mode	172
SVO?	Get Servo Mode	173
T		Page
TAC?	Tell Analog Channels	173
TAV?	Get Analog Input Voltage	173
TIO?	Tell Digital I/O Lines	173
TMN?	Get Minimum Commandable Position	174
TMX?	Get Maximum Commandable Position	174
TNR?	Get Number Of Record Tables	174
TRO	Set Trigger Output State	175
TRO?	Get Trigger Output State	175
TRS?	Indicate Reference Switch	175
TVI?	Tell Valid Character Set For Axis Identifiers	176
V		Page
VAR	Set Variable Value	176
VAR?	Get Variable Values	176
VEL	Set Closed-Loop Velocity	176
VEL?	Get Closed-Loop Velocity	177
VER?	Get Versions Of Firmware And Drivers	177
W		Page
WAC	Wait For Condition	178
WPA	Save Parameters To Nonvolatile Memory	179

#4 Request Status Register

Used in: (29), On-Target State (39)

Description: Queries system status information.

This command is identical in function to SRG? (p. 170) except that only one character is sent via the interface. Therefore, #4 can also be used when the controller is doing time-consuming tasks.

Format: **#4**

#4 corresponds to the EOT control character in ISO/IEC 6429.

Response: <StatusRegister>

<StatusRegister> System status information (HEX)

<StatusRegister> is bit-mapped. Includes per axis:

Bit	Description
15	On-target state
14	Referencing is being done
13	In motion
12	Servo mode activated
11	
10	Sensor signal valid
9	Reference edge found
8	Error
7	Digital input 4 active
6	Digital input 3 active
5	Digital input 2 active
4	Digital input 1 active
3	Sensor is referenced
2	Positive limit switch active
1	Reference switch active
0	Negative limit switch active

Deactivated axes are not included in the response.

#5 Request Motion Status

Used in: Triggering Motion (37)

Description: Queries the motion status of the axes.

Format: **#5**

#5 corresponds to the ENQ control character in ISO/IEC 6429.

Response: <MotionStatus>

<MotionStatus> motion status (HEX)

<MotionStatus> is bit-mapped. Each axis corresponds to one bit and the bit value corresponds to the axis number, e.g.,

Response	Description
1	Axis 1 is moving
2	Axis 2 is moving
5	Axis 3 and 1 are moving

#7 Request Controller Ready Status

Used in: Commandable Items (23)

Description: Queries the controller's ready state.

The controller is ready as soon as a new command can be executed.

Format: **#7**

#7 corresponds to the ␇ control character in ISO/IEC 6429.

Response: <ReadyStatus>

<ReadyStatus> ready state (HEX)

Possible answers:

Response from controller	Character in ISO/IEC 8859-1	Description
0xB1	±	Controller is ready
0xB0	°	Controller is not ready

Troubleshooting: The response characters may be displayed differently in other character sets.

#8 Query If Macro Is Running

Used in: (99)

Description: Tests if a macro is running on the controller.

Format: **#8**

#8 corresponds to the BS control character in ISO/IEC 6429.

Response: <MacroRunning>

<MacroRunning>	Macro is running (UINT)

<MacroRunning>	Description
1	A macro is running.
0	No macro is running.

#24 Stop All Axes

Used in: (100), Triggering Motion (37)

Description: Stops all axes abruptly.

Stops all motion started by motion commands (e.g., MOV (p. 160), MVR (p. 161), MVE, STE (p. 172), SMO), referencing commands (FNL (p. 138), FPL (p. 138), FRF (p. 139)), and macros.

Also stops the macro.

Sets the error code to 10.

After the axes are stopped, their target positions are set to their current positions.

This command is identical in function to STP (p. 172) except only one character is sent via the interface. Therefore, #24 can also be used when the controller is doing time-consuming tasks.

Format: **#24**

#24 corresponds to the CAN control character in ISO/IEC 6429.

*IDN? Get Device Identification

Used in: Commandable Items (23)

Description: Queries the device identification string.

Format: `*IDN?`

Response: <DeviceInformation> LF

<DeviceInformation>	Device identification string with controller name, serial number, and firmware version. (STRING)

ACC Set Closed-Loop Acceleration

Used in: Variables for Generating a Dynamics Profile (31)

Description: Sets acceleration of specified axes.

Changes the value of parameter 0xB (p. 192) in the volatile memory.

The maximum value that can be set with ACC and is specified by 0x4A (p. 202).

The acceleration set with ACC is only taken into account when the respective axis is in closed-loop operation (servo mode on). For fast step mode in PiezoMike mode, the value that is set is not evaluated.

ACC? Get Closed-Loop Acceleration

Used in: Variables for Generating a Dynamics Profile (31)

Description: Queries the acceleration value set with ACC (p. 124).

Format: `ACC?[{␣<AxisID>}]`

Arguments: <AxisID> Axis ID

The value for all axes will be queried if no arguments are specified.

Response: {<AxisID>=<Acceleration> LF}

<AxisID> Axis ID

<Acceleration> Acceleration in physical units / s^2 (FLOAT)

Troubleshooting: Illegal axis identifier

ADD Add and Save To Variable

Used in: Running the Macros (99)

Description: Adds two summands (as variable or specified directly) and saves the sum as variable.

Format: `ADD␣<Variable>␣<Summand1>␣<Summand2>`

Arguments: <Variable> Name of the variable where the sum is to be saved (STRING)

<Summand1> First summand (FLOAT)

<Summand2> Second summand (FLOAT)

CCL Set Command Level

Used in: (22), Parameter Commands (108)

Description: Changes the active command level.

The command level determines the availability of commands and write access to system parameters.

Command level: 0 - standard; 1 - expert mode; > 1 - PI service mode

Command levels > 1 cannot be activated; they are intended for PI service personnel. Contact the customer service department if there are problems with the parameters for command level 2 or higher (p. 10).

The active command level is always 0 when the electronics are switched on or rebooted.

The HPA? (p. 148) command lists the parameters and includes information on which command level allows write access to them.

Format: `CCL␣<Level>[␣<PSWD>]`

Arguments:

<Level>	Controller command level (UINT)
<PSWD>	Password for changing to the corresponding command level Password for changing to command level 1: advanced

Troubleshooting: Wrong password

CCL? Get Command Level

Used in: (22), Parameter Commands (108)

Description: Queries the active command level.

Format: `CCL?`

Response: <Level> LF

<Level>	Command level (UINT)

CPY Copy Into Variable

Used in: Analog Input Signals (89), Digital Input Signals (86), Running the Macros (99)

Description: Copies the response to a command into a variable.

Format: `CPY␣<Variable>␣<CMD?>`

Arguments:

<Variable>	Name of the variable to be copied to
<CMD?>	Query command that responds with a single value

CST? Get Assignment Of Stages To Axes

Used in: Commandable Items (23)

Description: Queries the name of the positioner type that is configured for the specified axis.

The positioner name is read from parameter 0x3C (p. 200). If the parameter has the value "NOSTAGE", the axis is deactivated.

Format: `CST? [{<AxisID>}]`

Arguments: <AxisID> Axis ID

Response: {<AxisID>=<String>LF}

<AxisID> Axis ID

<String> Name of the positioner type (STRING)

CSV? Get Current Syntax Version

Used in: (22)

Description: Get current GCS syntax version used in the firmware.

Format: `CSV?`

Response: <SyntaxVersion>

<SyntaxVersion> GCS syntax version (STRING)

<SyntaxVersion>	Description
1.0	GCS syntax version 1.0
2.0	GCS syntax version 2.0

CTO Set Configuration Of Trigger Output

Used in: Digital Output Signals (83), Setting "In Motion" Trigger Mode (85), Setting "Position Distance" Trigger Mode (83), Setting "Position+Offset" Trigger Mode (85), Setting "Single Position" Trigger Mode (86), Setting Up "Motion Error" Trigger Mode (84), Setting Up "On Target" Trigger Mode (84)

Description:

Configures the trigger output conditions for the specified digital output line.

Available CTO parameters:

<CTO - Pam>	Parameters	Possible values for <Value>	Description
1	TriggerStep	0	Distance
2	Axis	(Axis identifier)	Selects the axis that the trigger output is configured for.
3	TriggerMode	0	PositionDistance A trigger pulse is output each time the axis has traveled the distance set with **TriggerStep** (<CTOPam> 1).
		2	OnTarget The on-target status of the selected axis is transferred to the selected digital output line.
		5	MotionError The selected digital output line becomes active when a motion error occurs. The line remains active until the error code is reset to 0 (by a query with ERR? (p. 136)).
		6	InMotion The selected digital output line is active as long as the selected axis is in motion.
		7	Position+Offset The first trigger pulse is output when the axis has reached the position specified with **TriggerPosition** (<CTOPam> 10). The next trigger pulses are output respectively when the axis position equals the sum of the last valid trigger position and the distance specified by **TriggerStep** (<CTOPam> 1). Trigger output ends when the axis position exceeds the value specified by **StopThreshold** (<CTOPam> 9).

<CTO - Pam>	Parameters	Possible values for <Value>	Description
			The sign of the **TriggerStep** value determines which motion direction trigger pulses are to be output for.
		8	SinglePosition The selected digital output line is active when the axis position has reached or exceeded the position specified by **TriggerPosition** (<CTOPam> 10).
		9	HardwareTrigger Basically corresponds to the Position+Offset trigger mode but is done by the electronics (shorter processing time). Assignment of the axes to the digital output lines can be fixed.
7	Polarity	0	Sets the signal polarity to "active low"
		1	Sets the signal polarity to "active high" (default value)
8	StartThreshold	(Position - value)	Trigger output start position for the following trigger modes: ■ **PositionDistance** (TriggerMode 0)
9	StopThreshold	(Position - value)	Trigger output stop position for the following trigger modes: ■ **PositionDistance** (TriggerMode 0) ■ **Position+Offset** (TriggerMode 7) ■ **HardwareTrigger** (TriggerMode 9)
10	TriggerPosition	(Position - value)	Position of the (first) trigger output for the following trigger modes: ■ **Position+Offset** (TriggerMode 7) ■ **SinglePosition** (TriggerMode 8) ■ **HardwareTrigger** (TriggerMode 9)

<CTO - Pam>	Parameters	Possible values for <Value>	Description
11	PulseWidth	(Integer factor)	Factor that determines the pulse width for the Hardware-Trigger mode (TriggerMode 9). Pulse width = 33.3 ns × PulseWidth

Format: `CTO{␣<TrigOutID>␣<CTOPam>␣<Value>}`

Arguments:

<TrigOutID>	Digital output of the electronics
<CTOPam>	CTO parameter ID
<Value>	Value that the CTO parameter is set to

CTO? Get Configuration Of Trigger Output

Used in: Digital Output Signals (83)

Description: Queries the value that is configured for the specified trigger output line and the specified CTO parameter.

Format: `CTO?[{␣<TrigOutID>␣<CTOPam>}]`

Arguments:

<TrigOutID>	Digital output of the electronics
<CTOPam>	CTO parameter ID

If no arguments are specified, the response contains the values for all parameters and all output lines.

Response: {<TrigOutID>␣<CTOPam>=<Value>LF}

<TrigOutID>	Digital output of the electronics
<CTOPam>	CTO parameter ID
<Value>	Value of <CTOPam> that was configured for <TrigOutID>

DEC Set Closed-Loop Deceleration

Used in: Variables for Generating a Dynamics Profile (31)

Description: Sets the deceleration for the specified axis.

Changes the value of parameter 0xC (p. 192) in the volatile memory.

The maximum value that can be set with DEC, is specified by parameter 0x4B (p. 202).

The deceleration set with DEC is only taken into account when the respective axis is in closed-loop operation (servo mode on). For fast step mode in PiezoMike mode, the value that is set is not evaluated.

DEC? Get Closed-Loop Acceleration

Used in: Variables for Generating a Dynamics Profile (31)

Description:	Queries the deceleration value set with DEC (p. 129).	
Format:	`DEC? [{<AxisID>}]`	
Arguments:	<AxisID>	Axis ID
	The value for all axes will be queried if no arguments are specified.	
Response:	{<AxisID>=<Acceleration>LF}	
	<AxisID>	Axis ID
	<Acceleration>	Acceleration in physical units / s (FLOAT)
Troubleshooting:	Illegal axis identifier	

DEL Delay The Command Interpreter

Used in: Running the Macros (99)

Description:	Delays running the macro for a fixed period of time. DEL can only be used in macros. Note: Do not mix up DEL (delayed) with MAC DEL (p. 156) (deletes macros).	
Format:	`DEL␣<uint>`	
Arguments:	<uint>	Delay time span in milliseconds [UINT]

DFH Define Home Position

Description:

Redefines the zero position of the specified axis by setting the position value to zero at the current position.

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. 163)
- TMN? (p. 174)
- TMX? (p. 174)

Does **not** change the parameter values for the definition of travel range and soft limits.

The offset is reset to zero in the following cases:

- When switching on and rebooting the E-873.1AT: For all axes
- During referencing: For the affected axis

Format: `DFH[{␣<AxisID>}]`

Arguments:	<AxisID>	Axis ID

The zero positions for all axes will be newly defined if no arguments are specified.

Troubleshooting: Illegal axis identifier

DFH? Get Home Position Definition

Description:

Queries the position value that is configured for the specified axis as offset for shifting the zero position.

The offset for shifting the zero position is in the volatile memory. The offset is reset to zero in the following cases:

- When switching on or rebooting the Electronics: For all axes
- During referencing: For the affected axis

Format: `DFH?[{␣<AxisID>}]`

Arguments:	<AxisID>	Axis ID

The value for all axes will be queried if no arguments are specified.

Response: {<AxisID>=<PositionOffset> LF}

<AxisID>	Axis ID
<PositionOffset>	Axis position that was valid when executing DFH (p. 131)

Troubleshooting: Illegal axis identifier

DIO Set Digital Output Lines

Used in: Digital Output Signals (83)

Description: Switches a digital output line to the specified state.

All digital output lines with a bit-mapped hexadecimal number for <OutputOn> can be switched with <DIOID> = 0.

Use TIO? (p. 173) to query the number of available digital I/O lines.

Format: **DIO{␣<DIOID>␣<OutputOn>}**

Arguments:		
	<DIOID>	Digital output of the electronics
	<OutputOn>	Status of the digital output line (BOOL)

DIO? Get Digital Input Lines

Used in: Digital Input Signals (86)

Description: Queries the status of a digital input line.

Use TIO? (p. 173) to query the number of available digital I/O lines.

Format: **DIO?[{␣<DIOID>}]**

Arguments:		
	<DIOID>	Digital input line of the electronics

The state of all digital input lines is queried and output as bit-mapped hexadecimal number if no arguments are specified.

Response: {<DIOID>=<InputOn> LF}

	<DIOID>	Digital input line of the electronics
	<InputOn>	Digital input line state (HEX)

DRC Set Data Recorder Configuration

Used in: Analog Input Signals (89), Commandable Items (23), Determining the data to be recorded (80)

Description: Determines the data source to be used and the data type to be recorded for a data recorder table.

The data type to be recorded is specified via a record option. The record options depend on the controller and can be queried with HDR? (p. 141).

Format: `DRC{␣<RecTableID>␣<Source>␣<RecOption>}`

Arguments:

<RecTableID>	Data recorder table
<Source>	Data source ID
<RecOption>	Record option, data type to be recorded

Record options

Data source: all

0	Nothing is Recorded

Data source: axis

1	Commanded Position of Axis
2	Actual Position of Axis
3	Position Error of Axis
44	Timestamp (TIM?)
70	Commanded Velocity of Axis
71	Commanded Acceleration 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
91	Motor Current

Data source: input channel (1 - 8)

81	Analog Input

DRC? Get Data Recorder Configuration

Used in: Determining the data to be recorded (80)

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

Format: `DRC?[{␣<RecTableID>}]`

Arguments:

<RecTableID>	Data recorder table of the controller

If no arguments are specified, the settings for all data recorder tables will be queried.

Response: {<RecTableID>=<Source>␣<RecOption> LF}

<RecTableID>	Data recorder table
<Source>	Data source ID, e.g., controller axis or channel
<RecOption>	Data type to be recorded (record option, see DRC (p. 133))

DRL? Get Number Of Recorded Points

Used in: Reading data out (81)

Description: Queries the number of points stored by the last recording.

Format: `DRL?[{␣<RecTableID>}]`

Arguments:

<RecTableID>	Data recorder table

Response: {<RecTableID>=<uint> LF}

<RecTableID>	Data recorder table
<uint>	Number of saved points (UINT)

DRR? Get Recorded Data Values

Used in: Reading data out (81)

Description: Gets the last recorded data.

Can be used during recording.
Querying can take some time depending on the number of points to be read.

Format: `DRR?[␣<StartPoint> <NumberOfPoints> [{<RecTableID>}]]`

Arguments:

<StartPoint>	First point to be read (UINT >0)
<NumberOfPoints>	Number of points to be read per table
<RecTableID>	Data recorder table If <RecTableID> is not specified, all data from all tables is read whose recording option is different to zero.

If no arguments are specified, the data is read from all recorder tables with recording option not equal to zero.

Response: (Data in GCS array format)

DRT Set Data Recorder Trigger Source

Used in: Setting the trigger for recording (80), Starting the Recording (81)

Description: Defines a trigger option for the specified data recorder table.

Irrespective of the trigger option set, data recording is always triggered when step response measuring is done with STE (p. 172). Trigger options depend on the controller and can be queried with HDR? (p. 141).

Format: `DRT␣<RecTableID>␣<TriggerSource>␣<Value>`

Arguments:

<RecTableID>	Data recorder table ID <RecTableID> = 0 defines the trigger option for all available data recorder tables.
<TriggerSource>	ID of the trigger option
<Value>	Depends on the trigger option

Trigger options

0	***Default Setting*** Data recording is triggered with STE (p. 172). <Value> is a dummy.
1	***Any Command Changing Position*** Data recording is triggered by commands that influence the target position (e.g., MVR (p. 161), MOV (p. 160), MVE). <Value> is a dummy.
2	***Next Command*** Resets the trigger afterwards. <Value> is a dummy.
3	***External Trigger*** Data recording is triggered with a digital output line. <Value> is the ID of the digital output line.
6	***Any Command Changing Position, Reset Trigger After Execution*** Corresponds to <TriggerSource> = 1, resets the trigger afterwards. <Value> is a dummy.

DRT? Get Data Recorder Trigger Source

Used in: Setting the trigger for recording (80)

Description: Queries the trigger option for the data recorder tables.

Format: `DRT?[{␣<RecTableID>}]`

Arguments: <RecTableID> data recorder table

If no arguments are specified, the trigger option for all data recorder tables will be queried.

Response: {<RecTableID>=<TriggerSource>␣<Value>LF}

<RecTableID>	Data recorder table ID <RecTableID> = 0 queries the trigger option for all available data recorder tables.
<TriggerSource>	ID of the trigger option (see DRT (p. 135))
<Value>	Depends on the trigger option

ERR? Get Error Number

Used in: Restoring the E-873.1AT's Operational Readiness (79)

Description: Queries the error code of the last error that occurred and resets the error code to 0.

Only the last error is buffered; therefore, in the case of a problem, ERR? should be called after each command.

Format: `ERR?`

Response: <ErrorNumber>

<ErrorNumber>	Code for the last error (INT)

Troubleshooting: Communication breakdown

FED Find Edge

Used in: Digital Input Signals (86), Triggering motion in closed-loop operation (37)

Description:

Moves an axis to a specified signal edge.

FED does not set a specific position value at the selected edge (in contrast to FNL (p. 138), FPL (p. 138), and FRF (p. 139)); i.e., the axis is not referenced after using FED.

Servo mode must be switched on for the specified axis (axes) (closed-loop operation). HID control may not be activated for the corresponding axis.

If multiple axes are specified in the command, they are moved synchronously.

The following settings influence what type of motion is possible with FED:

- Is there a reference switch (parameter 0x14 (p. 194))?
- Are there limit switches (parameter 0x32 (p. 199))?
- If the reference switch outputs an index pulse: How is the move to the index pulse to be done (parameters 0x70 (p. 208), 0x78 (p. 210), 0x79 (p. 211))?

You can use the digital input lines instead of the switches as source of the switch signals for FED.

FED can be used to measure the physical travel range of a new mechanics and therefore determine the values for the corresponding parameters:

- Distance from the negative to the positive limit switch
- Gap between the negative limit switch and the reference switch (parameter ID 0x17 (p. 196))
- Gap between the reference switch and the positive limit switch (parameter ID 0x2F (p. 197)).

Format:

`FED{␣<AxisID>␣<EdgeID>␣<Param>}`

Arguments:

<AxisID>	Axis ID
<EdgeID>	Edge type, which the axis is to move to
<Param>	Depends on the selected edge and determines it more precisely

If multiple axes are specified in the command, they are moved synchronously.

Troubleshooting:

Illegal axis identifier

Limit and/or reference switches deactivated

Servo mode not active (SVO? (p. 173) responds with the value "0")

FNL Fast Reference Move To Negative Limit

Used in: Digital Input Signals (86), Referencing options (43), Triggering motion in closed-loop operation (37)

Description: Starts a reference move to the negative physical travel range limit respectively to the negative limit switch.

FNL sets the current position to the difference between 0x16 (p. 195) and 0x17 (p. 196). If the reference move was successful, absolute motion will then be possible in closed-loop operation.

Limit switches cannot be used for reference moves if soft limits (parameter 0x15 (p. 195) and 0x30 (p. 198)) are set.

The motion can be stopped by #24 (p. 123), STP (p. 172), and HLT (p. 148).

Servo mode must be switched on for the specified axis (axes) (closed-loop operation).

Format: `FNL[{␣<AxisID>}]`

Arguments: <AxisID> Axis ID

If multiple axes are specified in the command, they are moved synchronously.

If no arguments are specified, all axes are moved synchronously.

Troubleshooting: Illegal axis identifier

Servo mode not active (SVO? (p. 173) responds with the value "0")

FPL Fast Reference Move To Positive Limit

Used in: Digital Input Signals (86), Referencing options (43), Triggering motion in closed-loop operation (37)

Description: Starts a reference move to the positive physical travel range limit respectively to the positive limit switch.

FPL sets the current position to the sum of 0x16 (p. 195) and 0x2F (p. 197). If the reference move was successful, absolute motion will then be possible in closed-loop operation.

Limit switches cannot be used for reference moves if soft limits (parameter 0x15 (p. 195) and 0x30 (p. 198)) are set.

The motion can be stopped by #24 (p. 123), STP (p. 172), and HLT (p. 148).

Servo mode must be switched on for the specified axis (axes) (closed-loop operation).

Format: `FPL[{␣<AxisID>}]`

Arguments: <AxisID> Axis ID

If multiple axes are specified in the command, they are moved synchronously.

If no arguments are specified, all axes are moved synchronously.

Troubleshooting: Illegal axis identifier

Servo mode not active (SVO? (p. 173) responds with the value "0")

FRF Fast Reference Move To Reference Switch

Used in: Digital Input Signals (86), Referencing options (43), Triggering motion in closed-loop operation (37)

Description: Starts a reference move to the reference switch.

FRF sets the current position to the value of the **Value at Reference Position** (0x16 (p. 195)) parameter. If the reference move was successful, absolute motion will then be possible in closed-loop operation.

The motion can be stopped by #24 (p. 123), STP (p. 172), and HLT (p. 148).

Servo mode must be switched on for the specified axis (axes) (closed-loop operation).

Use FNL (p. 138) or FPL (p. 138) to do a reference move for an axis that does not have a reference switch but instead, only limit switches.

Format: `FRF[{␣<AxisID>}]`

Arguments: <AxisID> Axis ID

If multiple axes are specified in the command, they are moved synchronously.

If no arguments are specified, all axes are moved synchronously.

Troubleshooting: Illegal axis identifier

Servo mode not active (SVO? (p. 173) responds with the value "0")

FRF? Get Referencing Result

Used in: Referencing options (43)

Description: Queries whether the specified axis is referenced.

Format: `FRF?[{␣<AxisID>}]`

Arguments: <AxisID> Axis ID

The value for all axes will be queried if no arguments are specified.

Response: {AxisID>=<uint>LF}

<uint> Reference state of the axis (BOOL)

<uint>	Description
1	Axis referenced successfully
0	Axis not referenced

Troubleshooting: Illegal axis identifier

GOH Go To Home Position

Description: Moves the specified axis to the zero position.

`GOH{␣<AxisID>}` identical to `MOV{␣<AxisID>␣0}` (p. 160).

The motion can be stopped by #24 (p. 123), STP (p. 172), and HLT (p. 148).

Servo mode must be switched on for the specified axis (axes) (closed-loop operation). HID control may not be activated for the corresponding axis.

Format: `GOH[{␣<AxisID>}]`

Arguments:

<AxisID>	Axis ID

If multiple axes are specified in the command, they are moved synchronously.

If no arguments are specified, all axes are moved synchronously.

Troubleshooting: Illegal axis identifier

Servo mode not active (SVO? (p. 173) responds with the value "0")

HAR? Indicate Hard Stops

Used in: Referencing options (43)

Description: Queries whether the hard stops of the specified axis can be used for reference moves.

The E-873.1AT's firmware uses the Use Hard Stops for Referencing (p. 211) parameter (0x7A) to determine whether the hard stops of the axis can be used for reference moves. Depending on the value of this parameter, the E-873.1AT activates or deactivates reference moves that use the hard stops.

Format: `HAR?[{␣<AxisID>}]`

Arguments:

<AxisID>	Axis ID

Response: {<AxisID>=<uint> LF}

<AxisID>	Axis ID
<uint>	Can hard stops of the axis be used for reference moves? (BOOL)

Troubleshooting: Illegal axis identifier

HDR? Get All Data Recorder Options

Used in: Reading general information from the data recorder (80)

Description: Shows a help text that contains all available information on data recording.

Shows information on:

- Recording options
- Trigger options
- Additional parameters and commands for data recording

Format: `HDR?`

Response:

```
#RecordOptions␣LF
{<RecOption>=<DescriptionString>[␣of␣<Channel>]␣LF}
#TriggerOptions␣LF
[{<TriggerOption>=<DescriptionString>␣LF}]
#Parameters␣to␣be␣set␣with␣SPA␣LF
[{<ParameterID>=<DescriptionString>␣LF}]
#Additional␣information␣LF
[{<CommandDescription>(<Command>)␣LF}]
#Sources␣for␣Record␣Options␣LF
[{<RecOption>=<Source>␣LF}]
end␣of␣help
```

HDT Set HID Default Lookup Table

Used in: HID Control Configuration (90)

Description: Assigns a lookup table to the specified axis of the specified HID.

Format: `HDT{␣<HIDeviceID>␣<HIDeviceAxis>␣<HIDTableID>}`

Arguments:

<HIDeviceID>	Identifier of an HID connected to the electronics
<HIDeviceAxis>	HID axis
<HIDTableID>	Lookup table identifier

HDT? Get HID Default Lookup Table

Used in: HID Control Configuration (90)

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

Format: `HDT?[{␣<HIDeviceID>␣<HIDeviceAxis>}]`

Arguments:

<HIDeviceID>	Identifier of an HID connected to the electronics
<HIDeviceAxis>	HID axis

Response: {<HIDeviceID>␣<HIDeviceAxis>=<HIDTableID>LF}

<HIDeviceID>	Identifier of an HID connected to the electronics
<HIDeviceAxis>	HID axis
<HIDTableID>	Lookup table identifier

HIA Configure Control Done By HID Axis

Used in: HID Control Configuration (90), Triggering motion in closed-loop operation (37)

Description:

Configures control of electronics' axes via HID axes (HID control).

Assigns an HID axis to the specified motion variable.

The configuration of the HID control is saved only to the volatile memory (RAM) of the electronics.

HID control may not be activated for the corresponding axis.

Motion variables that can be controlled by HIDs (depending on the controller; use the HIA? (p. 144) command to query the current configuration of the HID controlled motion variables):

<MotionParam>	Description
0	Deletes the current configuration of the HID control. Can be sent from <HIDeviceID> and <HIDeviceAxis> without specification.
1	Absolute target position The lookup table value corresponding to the current displacement of the human interface device axis is mapped to the travel range of the E-873.1AT axis to be controlled. The travel range limits are specified by the values of parameters 0x30 (p. 198) and 0x15 (p. 195) can be queried with TMN? (p. 174) and TMX? (p. 174).
2	Relative target position Each pulse received (if applicable: Each mechanical detent) triggers relative motion over the distance set with the SST command (p. 171). No lookup tables are used to control the relative target position.
3	Velocity of the axis Product of the lookup table value corresponding to the current displacement of the HID axis and the currently valid maximum velocity of the controller axis.
4	Maximum velocity of the axis Product of the lookup table value corresponding to the current displacement of the HID axis and the currently valid maximum velocity of the controller axis. Motion variable 4 can only be assigned to an HID axis when motion variable 3 has been assigned to a different HID axis.

Format:

```
HIA{␣<AxisID>␣<MotionParam>␣<HIDeviceID>␣<HIDeviceAxis>}
```

Arguments:

<AxisID>	Axis ID
<MotionParam>	Axis motion variable
<HIDeviceID>	Identifier of an HID connected to the electronics
<HIDeviceAxis>	HID axis

Troubleshooting: <MotionParam> has the value zero, i.e., function to be controlled has not been selected for the axis

<HIDeviceID> has the value zero, i.e., HID has not been selected

<HIDeviceAxis> has the value zero, i.e., HID's axis has not been selected for HID control

HIA? Get Configuration Of Control Done By HID Axis

Used in: HID Control Configuration (90)

Description: Queries an HID's axis assigned to the specified motion variable of the specified motion variable.

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

Arguments:

<AxisID>	Axis ID
<MotionParam>	Axis motion variable

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

<AxisID>	Axis ID
<MotionParam>	Axis motion variable
<HIDeviceID>	Identifier of an HID connected to the electronics
<HIDeviceAxis>	HID axis

HIB? Get State Of HID Button

Used in: HID Control Configuration (90)

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

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

Arguments:

<HIDeviceID>	Identifier of an HID connected to the electronics
<HIDeviceButton>	HID button

Response: {<HIDeviceID>␣<HIDeviceButton>=<HIDButtonState>LF}

<HIDeviceID>	Identifier of an HID connected to the electronics
<HIDeviceButton>	HID button
<HIDButtonState>	Button status (INT)

Any possible values of <HIDButtonState> depend on the button type. The value range can be queried with HIS? (p. 146).

HIE? Get Deflection Of HID Axis

Used in: HID Control Configuration (90)

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

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

Arguments:

<HIDeviceID>	Identifier of an HID connected to the electronics
<HIDeviceAxis>	HID axis

Response: {<HIDeviceID>␣<HIDeviceAxis>=<HIDDeflection> LF}

<HIDeviceID>	Identifier of an HID connected to the electronics
<HIDeviceAxis>	HID axis
<HIDDeflection>	Displacement of the HID's axis (FLOAT)

<HIDDeflection> is a value between -1.0 and 1.0. A value almost equal to 0.0 corresponds to the axis' center position, -1.0 respectively 1.0 of the maximum displacement in a negative respectively positive direction.

HIN Set Activation State For HID Control

Used in: HID Control Configuration (90), Triggering motion in closed-loop operation (37)

Description: Sets the status of HID control for the specified axis.

Servo mode must be switched on for the specified axis (axes) (closed-loop operation).

Format: `HIN{␣<AxisID>␣<HIDControlState>}`

Arguments:

<AxisID>	Axis ID
<HIDControlState>	HID control's activation state (BOOL)

Troubleshooting: Illegal axis identifier
Servo mode not active (SVO? (p. 173) responds with the value "0")
HID control is not suitably configured (p. 143)

HIN? Get Activation State Of HID Control

Used in: HID Control Configuration (90)

Description: Queries status of HID control for the specified axis.

Format: `HIN?[{␣<AxisID>}]`

Arguments:

<AxisID>	Axis ID

Response: {<AxisID>=<HIDControlState> LF}

<AxisID>	Axis ID
<HIDControlState>	HID control's activation state (BOOL)

HIS? Get Configuration Of HI Device

Used in: Commandable Items (23), HID Control Configuration (90)

Description: Queries the specified property for the specified operating element of an HID.

Format: `HIS?[{␣<HIDeviceID>␣<HIDItemID>␣<HIDPropID>}]`

Arguments:

<HIDeviceID>	Identifier of an HID connected to the electronics
<HIDItemID>	HID operating element
<HIDPropID>	Property of the operating element

If no arguments are specified, information is queried on the supported operating elements of all HIDs.

Response: {<HIDeviceID>␣<HIDItemID>␣<HIDPropID>=<HIDPropValue> LF}

<HIDeviceID>	Identifier of an HID connected to the electronics
<HIDItemID>	HID operating element
<HIDPropID>	Property of the operating element
<HIDPropValue>	The property of the operating element is set to this value (STRING)

Possible values:

<HIDPropID>	Description
	Axis_<x>: HID axis, e.g., joystick axis or continuous slider, <x> indicates the identifier
	Button_<x>: HID button, <x> indicates the identifier
	Led_<x>: Output unit, e.g., LED or vibration motor on/off time, <x> indicates the identifier
2	Operating element status, e.g., axis displacement or LED activation status (FLOAT)
3	Name of the operating element (STRING)
4	HID Name (STRING)
5	Smallest possible value for the status of a "Button"- or "Led"-type operating element (INT)
6	Largest possible value for the status of a "Button"- or "Led"-type operating element (INT)

HIT Fill HID Lookup Table

Used in: HID Control Configuration (90)

Description: Fills the specified lookup table with values.

HIT can only be used to fill user-defined tables. Tables with identifier ≤ 100 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. The sign of the values determines the direction of motion of the HID-controlled axis. Parameter 0x61 (p. 207) can be used to reverse the assigned direction that is specified by the values in the lookup table for an HID-controlled axis.

The HDT (p. 141) command assigns the lookup tables to HID axes.

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

Arguments:

<HIDTableID>	Lookup table identifier
<HIDTableAddr>	Index of a point in the lookup table
<HIDTableValue>	Value of the point with the index <HIDTableAddr> (FLOAT, -1.0...1.0)

HIT? Get HID Lookup Table Values

Used in: HID Control Configuration (90)

Description: Queries the values of the specified points in the specified lookup table.

Format: `HIT?[␣<StartPoint>[␣<NumberOfPoints>[{␣<HIDTableID>}]]]`

Arguments:

<StartPoint>	Index of the first point that is queried
<NumberOfPoints>	Number of points to be queried per lookup table
<HIDTableID>	Lookup table identifier

Response: (Data in GCS array format)

HLP? Get List Of Available Commands

Used in: (22)

Description: Lists a help string which contains all commands available.

Format: `HLP?`

Response: (List of available commands)

HLT Halt Motion Smoothly

Used in: Triggering Motion (37)

Description:	Stops motion for the specified axis while considering the maximum set deceleration. Sets the error code to 10. Does not apply to trajectories: HLT also triggers an abrupt stop of motion when a trajectory is being followed.	
Format:	`HLT[{␣<AxisID>}]`	
Arguments:	<AxisID>	Axis ID
Troubleshooting:	Illegal axis identifier	

HPA? Get List Of Available Parameters

Used in: (22), Adapting Settings (108), Parameter Commands (108)

Description:	Shows a help text that contains all available parameters with a short description.	
Format:	`HPA?`	
Response:	List of available parameters in the format: <PamID>=TAB<CmdLevel>TAB<MaxItem>TAB<DataType>TAB<FunctionGroupDescription>TAB<ParameterDescription>TAB[{<PossibleValue>=<ValueDescription>}]	
	<PamID>	Parameter ID
	<CmdLevel>	Command level for write access to the parameter
	<MaxItem>	Maximum number of elements of the same type that are affected by the parameter
	<DataType>	Data type of the parameter value
	<FunctionGroupDescription>	Name of the function group which the parameter belongs to
	<ParameterDescription>	Name of the parameter
	<PossibleValue>	Possible value
	<ValueDescription>	Value description

HPV? Get Parameter Value Description

Description: Displays a help text that contains possible parameter values.

Format: HPV?

Response:

#Possible␣parameter␣values␣are: LF

{<PamID>␣<ItemID>=<ListType>[{→<PossibleValue>=<ValueDescription>}] LF}

#CCL␣levels␣are: LF

{<PamID>␣<ItemID>=<CmdLevel> LF}

end of help

<PamID>	Parameter ID
<ItemID>	Element of the electronics <ItemID> = 0: Description applies to all elements
<ListType>	Value list type <ListType> = 0: Parameter does not apply to this element <ListType> = 1: List of possible values <ListType> = 2: Minimum and maximum value
<PossibleValue>	Possible value
<ValueDescription>	Value description
<CmdLevel>	Command level for write access to the parameter

IFC Set Interface Parameters Temporarily

Used in: E-873.1AT Interfaces (16)

Description: Configures the interface parameters in the volatile memory.

The changed interface parameters are active immediately. The PC's interface configuration may also have to be changed and the connection to the electronics re-established.

The configuration of the interface parameters is saved only to the volatile memory (RAM) of the electronics. Use IFS (p. 152) to change interface parameters in the nonvolatile memory.

Format: `IFC{␣<InterfacePam>␣<PamValue>}`

Arguments:

<InterfacePam>	Interface parameters (STRING)
<PamValue>	Parameter value

Possible values

<InterfacePam>	<PamValue>	Description
RSBAUD	9600, 19200, 38400, 57600 or 115200	Baud rate for the RS-232 interface
IPADR	<UINT>.<UINT>.<UINT>.<UINT>:50000	IP address and port for the TCP/IP interface Port 50000 cannot be changed. IPADR is only used when IPSTART = 0.
IPSTART	0, 1	<PamValue> = 0: The IP address defined by IPADR is used <PamValue> = 1: DHCP is used (default)
IPMASK	<UINT>.<UINT>.<UINT>.<UINT>	TCP/IP interface subnet mask
IPGTWAY	<UINT>.<UINT>.<UINT>.<UINT>	Default gateway for TCP/IP communication

IFC? Get Current Interface Parameters

Used in: E-873.1AT Interfaces (16)

Description: Queries the interface parameter values in the volatile memory.

Format: `IFC?[{␣<InterfacePam>}]`

Arguments: <InterfacePam> Interface parameters (STRING)

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

<InterfacePam> Interface parameters (STRING)

<PamValue> Parameter value

in the volatile memory

Possible values for <InterfacePam>:

<InterfacePam>	Description
RSBAUD	Baud rate for the RS-232 interface
IPADR	IP address and port for the TCP/IP interface
IPSTART	Startup behavior (DHCP status) of the TCP/IP interface
IPMASK	TCP/IP interface subnet mask
MACADR	Mac address (unique network hardware address)
IPGTWAY	Default gateway for TCP/IP communication

IFS Set Interface Parameters As Default Values

Used in: E-873.1AT Interfaces (16)

Description: Configures the interface parameters in the nonvolatile memory.

The changed interface parameters are active after the next reboot. The PC's interface configuration may also have to be changed.

Notice: Note that the number of write cycles in the nonvolatile memory is limited. Therefore, save to the nonvolatile memory only when necessary.

Use IFC (p. 150) to change the interface parameters in the volatile memory (RAM).

Format: `IFS <Pswd>{␣<InterfacePam>␣<PamValue>}`

Arguments:

<Pswd>	Password for writing to the nonvolatile memory

Default value is "100".

<InterfacePam>	Interface parameters (STRING)
<PamValue>	Parameter value

Possible values

<InterfacePam>	<PamValue>	Description
RSBAUD	9600, 19200, 38400, 57600 or 115200	Baud rate for the RS-232 interface
IPADR	<UINT>.<UINT>.<UINT>.<UINT>:50000	IP address and port for the TCP/IP interface Port 50000 cannot be changed. IPADR is only used when IPSTART = 0.
IPSTART	0, 1	<PamValue> = 0: The IP address defined by IPADR is used <PamValue> = 1: DHCP is used (default)
IPMASK	<UINT>.<UINT>.<UINT>.<UINT>	TCP/IP interface subnet mask
IPGTWAY	<UINT>.<UINT>.<UINT>.<UINT>	Default gateway for TCP/IP communication

IFS? Get Interface Parameters As Default Values

Used in: E-873.1AT Interfaces (16)

Description: Queries the values of the interface parameters in the nonvolatile memory.

Format: `IFS?[{␣<InterfacePam>}]`

Arguments: <InterfacePam> Interface parameters (STRING)

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

<InterfacePam> Interface parameters (STRING)

<PamValue> Parameter value

in the nonvolatile memory

Possible values for <InterfacePam>:

<InterfacePam>	Description
RSBAUD	Baud rate for the RS-232 interface
IPADR	IP address and port for the TCP/IP interface
IPSTART	Startup behavior (DHCP status) of the TCP/IP interface
IPMASK	TCP/IP interface subnet mask
MACADR	Mac address (unique network hardware address)
IPGTWAY	Default gateway for TCP/IP communication

JRC Jump Relatively Depending On Condition

Used in: Analog Input Signals (89), Digital Input Signals (86), Running the Macros (99)

Description: Jumps relative to a specified number of program lines within a macro.

Jumps irrespective of a specified condition.

Can only be used in macros.

Possible relational operators:

<OP>	Description
=	Equal
!=	Not equal
<=	Smaller than or equal
<	Smaller than
>=	Larger than or equal
>	Larger than

Format: `JRC␣<Jump>␣<CMD?>␣<OP>␣<Value>`

Arguments:

<Jump>	Size of relative jump
<CMD?>	Query command that responds with a single value
<OP>	Relational operator
<Value>	Relational value with <CMD?>

Troubleshooting: Wrong jump target (<Jump>) specified

Wrong relational operator (<OP>) specified

LIM? Indicate Limit Switches

Used in: Detecting Limit Switches (41)

Description: Queries whether the specified axis has limit switches.

Limit switch is detected by 0x32 (p. 199).

Format: `LIM?[{␣<AxisID>}]`

Arguments:

<AxisID>	Axis ID

The value for all axes will be queried if no arguments are specified.

Response: {<AxisID>=<uint>LF}

<AxisID>	Axis ID
<uint>	Limit switches available? (BOOL)

Troubleshooting: Illegal axis identifier

MAC Call Macro Function

Used in: Triggering motion in closed-loop operation (37)

Description: Calls a macro function.

Possible macro functions are described separately:

- MAC BEG (p. 155)
- MAC DEF (p. 155)
- MAC DEF? (p. 155)
- MAC DEL (p. 156)
- MAC END (p. 156)
- MAC ERR? (p. 156)
- MAC NSTART (p. 156)
- MAC START (p. 157)

Format: `MAC␣<Keyword>{␣<Parameter>}`

Arguments: <Keyword> Macro function called

<Parameter> Function-dependent parameters

Troubleshooting: Macro recording is active

MAC BEG Call Macro Function: BEG

Used in: (98)

Description: Start recording a macro.

Recording is stopped by MAC END (p. 156).

Format: `MAC␣BEG␣<MacroName>`

Arguments: <MacroName> Macro name

MAC DEF Call Macro Function: DEF

Used in: (101)

Description: Specifies a macro as startup macro.

Format: `MAC␣DEF␣<MacroName>`

Arguments: <MacroName> Name of the macro that is specified as startup macro

A startup macro is not used when no arguments are specified.

MAC DEF? Call Macro Function: DEF?

Used in: (101)

Description: Queries the name of the startup macro.

Format: `MAC␣DEF?`

Response: [<MacroName>] LF

<MacroName> Startup macro name

MAC DEL Call Macro Function: DEL

Used in: (101)

Description: Deletes the specified macro.

Format: `MAC␣DEL␣<MacroName>`

Arguments: <MacroName> Macro name to be deleted

MAC END Call Macro Function: END

Used in: (98)

Description: Ends macro recording.

Format: `MAC␣END`

MAC ERR? Call Macro Function: ERR?

Used in: (100)

Description: Reports the last error that occurred while the macro was running.

Format: `MAC␣ERR?`

Response: <MacroName>␣<uint1>=<uint2>"<"<CMD>">"

<MacroName>	Name of the macro
<uint1>	Line in the micro where the error occurred
<uint2>	Error code
<CMD>	Bad command

MAC NSTART Call Macro Function: NSTART

Used in: (99)

Description: Runs the specified macro several times.

The macro is restarted when the last macro recording has completed, until <uint> has been reached.

Format: `MAC␣NSTART␣<MacroName>␣<uint>[␣<String1>[␣<String2>[␣<String3>[␣<String4>]]]]`

Arguments:

<MacroName>	Macro name
<uint>	Number of runs
<String1...4>	Local variables 1 to 4

Troubleshooting: No local variables specified although local variables are used in the macro

MAC START Call Macro Function: START

Used in: (99)

Description: Runs the specified macro.

Format:

```
MAC␣START␣<Macro-
Name>[␣<String1>[␣<String2>[␣<String3>[␣<String4>]]]]
```

Arguments:

<MacroName>	Macro name
<String1...4>	Local variables 1 to 4

Troubleshooting: No local variables specified although local variables are used in the macro

MAC? List Macros

Used in: (98)

Description: Lists macros or content of a specified macro.

Format:

```
MAC?[␣<MacroName>]
```

Arguments:

<MacroName>	Name of a macro

If no arguments are specified, the names of all saved macros are listed.

Response: <String>

<String>	Content of the macro or list of the names of all saved macros

Troubleshooting: Wrong macro name (<MacroName>)

MAN? Get Help String For Command

Description: Shows a help text for a command.

Format:

```
MAN?␣<CMD>
```

Arguments:

<CMD>	Command that the help text is to be shown for

Response: <String>

<String>	Help text

MAT Calculate And Save To Variable

Description:

Performs a mathematical or bit operation and saves the result as a variable.

The variable is available in volatile memory (RAM) only.

Possible operators:

<OP>	Operation
+	Addition (mathematical)
-	Subtraction (mathematical)
*	Multiplication (mathematical)
AND	AND (logical)
OR	OR (logical)
XOR	EXCLUSIVE OR (logical)

Format:

```
MAT␣<Variable>=<Float1>␣<OP>␣<Float2>
```

Arguments:

<Variable>	Name of the variable where the result is to be saved
<Float1>, <Float2>	Values for calculating the result
<OP>	Operator to be used

MEX Stop Macro Execution Due To Condition

Used in: Analog Input Signals (89), Digital Input Signals (86), Stopping the Macros (100)

Description:

Stops the macro due to a specified condition.

If the parser encounters this command, the condition is checked. If the condition is fulfilled at a later time, it is ignored by the parser.

Can only be used in macros.

Possible relational operators:

<OP>	Description
=	Equal
!=	Not equal
<=	Smaller than or equal
<	Smaller than
>=	Larger than or equal
>	Larger than

Format: `MEX␣<CMD?>␣<OP>␣<Value>`

Arguments:

<CMD?>	Query command that responds with a single value
<OP>	Relational operator
<Value>	Relational value with <CMD?>

MOV Set Target Position

Used in: Triggering motion in closed-loop operation (37), Variables for Generating a Dynamics Profile (31)

Description: Sets the absolute target position for the specified axis.

MOV overwrites the last target position received by the electronics. Motion commands from macros and command lines can overwrite each other. The last target position received may then not be reached.

Servo mode must be switched on for the specified axis (axes) (closed-loop operation). HID control may not be activated for the corresponding axis.

The target position must be inside the soft limits.

The motion can be stopped by #24 (p. 123), STP (p. 172), and HLT (p. 148).

Format: `MOV{␣<AxisID>␣<Position>}`

Arguments:

<AxisID>	Axis ID
<Position>	Absolute target position in physical units

Troubleshooting:

Illegal axis identifier

Servo mode not active (SVO? (p. 173) responds with the value "0")

Executing motion commands deactivated by parameter 0x130

Target position outside of the soft limits

HID control is enabled

MOV? Get Target Position

Description: Queries last valid commanded target position.

Format: `MOV?[{␣<AxisID>}]`

Arguments:

<AxisID>	Axis ID

Response: {<AxisID>=<float>LF}

<AxisID>	Axis ID
<float>	Last valid target position in physical units (FLOAT)

Troubleshooting: Illegal axis identifier

MVR Set Target Relative To Current Position

Used in: Triggering motion in closed-loop operation (37), Variables for Generating a Dynamics Profile (31)

Description:

Moves the specified axis relative to the last commanded target position.

The new target position is calculated from the sum of the last commanded target position and <Distance>.

Servo mode must be switched on for the specified axis (axes) (closed-loop operation). HID control may not be activated for the corresponding axis.

The target position must be inside the soft limits.

The motion can be stopped by #24 (p. 123), STP (p. 172), and HLT (p. 148).

Format: `MVR{␣<AxisID>␣<Distance>}`

Arguments:

<AxisID>	Axis ID
<Distance>	Distance that the axis has to move, specification in physical units

Troubleshooting:

Illegal axis identifier

Servo mode not active (SVO? (p. 173) responds with the value "0")

Executing motion commands deactivated by parameter 0x130

Target position outside of the soft limits

HID control is enabled

ONT? Get On-Target State

Used in: On-Target State (39)

Description:

Queries the on-target status of the specified axis.

The on-target state is influenced by the 0x36 (p. 200) and 0x3F (p. 201) settings.

Servo mode must be switched on for the specified axis (axes) (closed-loop operation).

Format: `ONT?[{␣<AxisID>}]`

Arguments:

<AxisID>	Axis ID

The value for all axes will be queried if no arguments are specified.

Response: {<AxisID>=<uint>LF}

<AxisID>	Axis ID
<uint>	On-Target-Status (BOOL)

Troubleshooting:

Illegal axis identifier

Servo mode not active (SVO? (p. 173) responds with the value "0")

OSM Open-Loop Step Moving

Used in: Triggering motion in open-loop operation (38)

Description:	Moves the specified axis by the specified number of steps. The velocity in open-loop step mode is controlled via the step frequency (parameter 0x1F000400). Servo mode must be switched off for the specified axis (axes) (open-loop operation).	
Format:	`OSM{␣<AxisID>␣<Value>}`	
Arguments:	<AxisID>	Axis ID
	<Value>	Number of steps to be made
Troubleshooting:	Illegal axis identifier Servo mode active (SVO? (p. 173) responds with the value "1") HID control is enabled	

OSN? Read Number Steps

Used in: Triggering motion in open-loop operation (38)

Description:	Queries the number of steps that still have to be performed by the specified axis.	
Format:	`OSN?[{␣<AxisID>}]`	
Arguments:	<AxisID>	Axis ID
	The value for all axes will be queried if no arguments are specified.	
Response:	{<AxisID>=<uint> LF}	
	<AxisID>	Axis ID
	<uint>	Number of steps still to be made
Troubleshooting:	Illegal axis identifier	

POS Set Real Position

Used in: Referencing options (43)

Description: Sets the current position of the axis (does not trigger motion).

POS can only be used when the referencing method is set to "0" (p. 164).

The minimum (TMN? (p. 174)) and maximum (TMX? (p. 174)) commandable positions are not influenced by POS. This means that target positions are possible that are permitted by the electronics but cannot be reached by the mechanics, or could be reached by the mechanics but are not permitted by the electronics.

■

■

Format: `POS{␣<AxisID>␣<Position>}`

Arguments: <AxisID> Axis ID

<Position> New current position [phys. unit]

Troubleshooting: Illegal axis identifier

POS? Get Real Position

Description: Queries the current axis position.

Format: `POS?[{␣<AxisID>}]`

Arguments: <AxisID> Axis ID

The value for all axes will be queried if no arguments are specified.

Response: {<AxisID>=<float> LF}

<AxisID> Axis ID

<float> Current axis position in physical units

Troubleshooting: Illegal axis identifier

RBT Reboot System

Used in: Controller Macros (97)

Description: Restarts the electronics.

The electronics behave in the same way after restarting as they do after switching on.

RBT cannot be used in macros.

Format: `RBT`

RMC? List Running Macros

Used in: (99)

Description:	Queries all macros currently running.
Format:	`RMC?`
Response:	{[<MacroName>] LF} <MacroName> Macro name

RON Set Reference Mode

Used in: Referencing options (43)

Description: Sets the referencing method for the specified axis.

Format: `RON{␣<AxisID>␣<ReferenceOn>}`

Arguments:

<AxisID>	Axis ID
<ReferenceOn>	Referencing method

Possible referencing modes:

0 Absolute position value is assigned with POS (p. 163) or a reference move is done with FRF (p. 139) (also FNL (p. 138) or FPL (p. 138) if supported by the controller). Relative motion is possible with MVR (p. 161), even when the axis has not been referenced.

1 A reference move must be done with FRF (p. 139) (also FNL (p. 138) or FPL (p. 138) if supported by the controller) for referencing, POS (p. 163) is not permitted. Motion in closed-loop operation is only possible after the axis has been referenced. (Default)

Troubleshooting: Illegal axis identifier

RON? Get Reference Mode

Used in: Referencing options (43)

Description: Queries referencing method of the specified axis.

Format: `RON?[{␣<AxisID>}]`

Arguments: <AxisID> Axis identifier

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

<AxisID> Axis identifier

<ReferenceOn> Referencing mode

Possible referencing modes:

0 Absolute position value is assigned with POS (p. 163) or a reference move is done with FRF (p. 139) (also FNL (p. 138) or FPL (p. 138) if supported by the controller). Relative motion is possible with MVR (p. 161), even when the axis has not been referenced.

1 A reference move must be done with FRF (p. 139) (also FNL (p. 138) or FPL (p. 138) if supported by the controller) for referencing, POS (p. 163) is not permitted. Motion in closed-loop operation is only possible after the axis has been referenced. (Default)

Troubleshooting: Illegal axis identifier

RPA Reset Volatile Memory Parameters

Used in: Parameter Commands (108)

Description: Resets the specified parameter.

The parameter value in the volatile memory is overwritten by the value in the nonvolatile memory.

RPA resets the parameter for the setting of hardware-specific parameters. Wrong values can lead to faulty operation or damage to the hardware.

Format: `RPA[{␣<ItemID>␣<PamID>}]`

Arguments: <ItemID> Element of the electronics

For example, axis identifier
Parameter <PamID> is reset for this element.

<PamID> Parameter ID

Troubleshooting: Impermissible element identifier

Wrong parameter ID

RTR Set Record Table Rate

Used in: Setting the record table rate (80)

Description: Sets the record table rate of the data recorder.

The recording rate multiplied by the cycle time of the electronics results in the recording interval of the data recorder. Settings greater than 1 allow recording over longer periods of time.

Format: `RTR <RecordTableRate>`

Arguments: <RecordTableRate> Data recording rate of the data recorder in number of cycles (UINT >0)

RTR? Get Record Table Rate

Used in: Setting the record table rate (80)

Description: Queries the current recording rate of the date recorder.

The recording rate multiplied by the cycle time of the electronics results in the recording interval of the data recorder. Settings greater than 1 allow recording over longer periods of time.

Format: `RTR?`

Response: <RecordTableRate> LF

<RecordTableRate> Data recording rate of the data recorder in number of cycles (UINT >0)

SAI Set Current Axis Identifiers

Used in: Commandable Items (23)

Description: Sets the axis identifier for the specified axis.

The new axis identifier is stored in the nonvolatile memory of the electronics.

Use TVI? (p. 176) to query the valid characters for the axis identifier.

Format: `SAI{␣<AxisID>␣<NewIdentifier>}`

Arguments: <AxisID> Axis ID

<NewIdentifier> New axis ID

SAI? Get List Of Current Axis Identifiers

Used in: Commandable Items (23)

Description: Queries the axis identifiers.

ALL ensures that the response also includes axes that are deactivated for electronics that permit axes to be deactivated.

Format: `SAI?[␣ALL]`

Arguments:	[␣ALL]	Ensures that electronics that allow axes to be deactivated (parameter 0x3C (p. 200) = "NOSTAGE") also allow deactivated axes to be queried.
Response:	{<AxisID> LF}	
	<AxisID>	Axis ID

SEP Set Non-Volatile Memory Parameters

Used in: Parameter Commands (108), Variables for Generating a Dynamics Profile (31)

Description: Sets a parameter in the nonvolatile memory to a specific value.

Up to four parameters can be set per command.

Wrong values can lead to faulty operation or damage to the hardware.

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

Arguments:	<Pswd>	Password for writing to the nonvolatile memory
	<ItemID>	Element of the electronics
	<PamID>	Parameter ID
	<PamValue>	Parameter value

Troubleshooting: Impermissible element identifier
Wrong parameter ID
Wrong password

SEP? Get Non-Volatile Memory Parameters

Used in: Parameter Commands (108), Saving Parameter Values in a Text File (109)

Description: Queries the value of a parameter in the nonvolatile memory.

Up to four parameters can be queried per command.

Format: `SEP?[{␣<ItemID>␣<PamID>}]`

Arguments:	<ItemID>	Element of the electronics
	<PamID>	Parameter ID
Response:	{<ItemID>␣<PamID>=<PamValue>LF}	
	<ItemID>	Element of the electronics
	<PamID>	Parameter ID
	<PamValue>	Parameter value

Troubleshooting: Impermissible element identifier
Wrong parameter ID

SPA Set Volatile Memory Parameters

Used in: Parameter Commands (108), Variables for Generating a Dynamics Profile (31)

Description: Sets a parameter in the volatile memory to a specific value.

Up to four parameters can be set per command.
Wrong values can lead to faulty operation or damage to the hardware.

Format: `SPA{␣<ItemID>␣<PamID>␣<PamValue>}`

Arguments:	<ItemID>	Element of the electronics
	<PamID>	Parameter ID
	<PamValue>	Parameter value

Troubleshooting: Impermissible element identifier
Wrong parameter ID

SPA? Get Volatile Memory Parameters

Used in: Parameter Commands (108), Saving Parameter Values in a Text File (109)

Description:	Queries the value of a parameter in the volatile memory. Up to four parameters can be queried per command.	
Format:	`SPA?[{␣<ItemID>␣<PamID>}]`	
Arguments:	<ItemID>	Element of the electronics
	<PamID>	Parameter ID
Response:	{<ItemID>␣<PamID>=<PamValue>LF}	
	<ItemID>	Element of the electronics
	<PamID>	Parameter ID
	<PamValue>	Parameter value
Troubleshooting:	Impermissible element identifier Wrong parameter ID	

SRG? Query Status Register Value

Used in: (29), On-Target State (39)

Description: Queries system status.

Format: `SRG?{␣<ItemID>␣<RegisterID>}`

Arguments:

<ItemID>	Element of the electronics
<RegisterID>	Register ID

Response: {<ItemID>␣<RegisterID>=<Value>LF}

<ItemID>	Element of the electronics
<RegisterID>	Register ID
<Value>	Register value (HEX)

<StatusRegister> is bit-mapped. It comprises:

Bit	Description
15	On-target state
14	Referencing is being done
13	In motion
12	Servo mode activated
11	
10	Sensor signal valid
9	Reference edge found
8	Error
7	Digital input 4 active
6	Digital input 3 active
5	Digital input 2 active
4	Digital input 1 active
3	Sensor is referenced
2	Positive limit switch active
1	Reference switch active
0	Negative limit switch active

SST Set Step Size

Description:

Sets the distance to be travelled for relative motion that is triggered by an HID.

Sets the distance (step size) for motion of the specified axis triggered by a manual control unit.

Format: `SST␣{<AxisID>␣<StepSize>}`

Arguments:

<AxisID>	Axis ID
<StepSize>	Distance. Specified in the physical unit of the axis position (format: FLOAT)

Troubleshooting:

Illegal axis identifier

Illegal value

SST? Get Step Size

Description:

Queries the distance to be travelled for relative motion triggered by an HID.

Queries the distance (step size) for motion of the specified axis triggered by a manual control unit.

Format: `SST?␣[{<AxisID>}]`

Arguments:

<AxisID>	Axis ID
<StepSize>	Distance. Specified in the physical unit of the axis position (format: FLOAT)

Response: {<AxisID>=<StepSize> LF}

<AxisID>	Axis ID
StepSize<>	Distance in physical units (format: FLOAT)

Troubleshooting:

Illegal axis identifier

Illegal value

STE Start Step And Response Measurement

Used in: Starting the Recording (81), Triggering motion in closed-loop operation (37), Triggering motion in open-loop operation (38)

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

Servo mode must be switched on for the specified axis (axes) (closed-loop operation). HID control may not be activated for the corresponding axis.

The target position must be inside the soft limits.

Format: `STE␣<AxisID>␣<Amplitude>`

Arguments:

<AxisID>	Axis ID
<Amplitude>	Jump size in physical units

Troubleshooting:

Illegal axis identifier

Servo mode not active (SVO? (p. 173) responds with the value "0")

Executing motion commands deactivated by parameter 0x130

Target position outside of the soft limits

HID control is enabled

STP Stop All Axes

Used in: (100), Triggering Motion (37)

Description: Stops all axes abruptly.

Stops all motion started by motion commands (e.g., MOV (p. 160), MVR (p. 161), MVE, STE (p. 172), SMO), referencing commands (FNL (p. 138), FPL (p. 138), FRF (p. 139)), and macros.

Also stops the macro.

Sets the error code to 10.

After the axes are stopped, their target positions are set to their current positions.

Format: `STP`

SVO Set Servo Mode

Used in: (29), Restoring the E-873.1AT's Operational Readiness (79)

Description: Sets the servo mode for the specified axis.

The target position is set to the current position when switching to closed-loop operation. SVO deactivates the brake when it is activated.

Format: `SVO{␣<AxisID>␣<ServoState>}`

Arguments:

<AxisID>	Axis ID
<ServoState>	Servo mode (UINT)

SVO? Get Servo Mode

Used in: (29)

Description: Queries the servo mode for the specified axis.

Format: `SVO?[{␣<AxisID>}]`

Arguments: <AxisID> Axis ID

The value for all axes will be queried if no arguments are specified.

Response: {<AxisID>=<ServoState> LF}

<AxisID> Axis ID

<ServoState> Axis servo mode (UINT)

Troubleshooting: Illegal axis identifier

TAC? Tell Analog Channels

Used in: Analog Input Signals (89), Commandable Items (23)

Description: Queries the number of available analog lines.

Format: `TAC?`

Response: <uint>

<uint> Number of analog lines (inputs and outputs)

TAV? Get Analog Input Voltage

Used in: Analog Input Signals (89), Commandable Items (23)

Description: Queries the voltage at the analog input.

Format: `TAV?[{␣<AnalogInputID>}]`

Arguments: <AnalogInputID> is the identifier of an analog input channel

Response: {<AnalogInputID>=<float> LF}

<AnalogInputID> is the identifier of an analog input channel
<float> Current voltage at the analog input (FLOAT)
All analog inputs are queried if no arguments are specified.

TIO? Tell Digital I/O Lines

Used in: Digital Input Signals (86), Digital Inputs and Outputs (82), Digital Output Signals (83)

Description: Queries the number of available digital I/O lines.

Format: `TIO?`

Response: I=<uint1>␣LF
O=<uint2> LF

<uint1> Number of digital input lines (UINT)
<uint2> Number of digital output lines (UINT)

TMN? Get Minimum Commandable Position

Description: Queries the minimum commandable position.

The minimum commandable position is determined by parameter 0x30 (p. 198).

Format: `TMN?[{␣<AxisID>}]`

Arguments: <AxisID> Axis ID

Response: {<AxisID>=<float>LF}

<AxisID> Axis ID

<float> Minimum commandable position in physical units (FLOAT)

TMX? Get Maximum Commandable Position

Description: Queries the maximum commandable position.

The maximum commandable position is defined by parameter 0x15 (p. 195).

Format: `TMX?[{␣<AxisID>}]`

Arguments: <AxisID> Axis ID

Response: {<AxisID>=<float>LF}

<AxisID> Axis ID

<float> Maximum commandable position in physical units (FLOAT)

TNR? Get Number Of Record Tables

Used in: Commandable Items (23)

Description: Queries the number of available data recorder tables.

Format: `TNR?`

Response: <uint>

<uint> Number of available data recorder tables (UINT)

TRO Set Trigger Output State

Used in: Digital Output Signals (83), Setting "In Motion" Trigger Mode (85), Setting "Position Distance" Trigger Mode (83), Setting "Position+Offset" Trigger Mode (85), Setting "Single Position" Trigger Mode (86), Setting Up "On Target" Trigger Mode (84)

Description: Activates or deactivates the trigger output for a digital output line.

Format: `TRO{␣<TrigOutID>␣<TrigMode>}`

Arguments:

<TrigOutID>	Digital output of the electronics
<TrigMode>	Trigger output state (UINT)

Troubleshooting: Impermissible element identifier

TRO? Get Trigger Output State

Used in: Digital Output Signals (83)

Description: Queries the trigger output state for a a digital output line.

Format: `TRO?[{␣<TrigOutID>}]`

Arguments:

<TrigOutID>	Digital output of the electronics

If no arguments are specified, the state of all digital output lines is queried.

Response: {<TrigOutID>=<TrigMode> LF}

<TrigMode>	Trigger output state (UINT)

Troubleshooting: Impermissible element identifier

TRS? Indicate Reference Switch

Used in: Reference Switch Detection (41)

Description: Queries whether the specified axis has a reference switch with direction sensing.

TRS? queries the value of parameter 0x14 (p. 194).

Format: `TRS?[{␣<AxisID>}]`

Arguments:

<AxisID>	Axis ID

Response: {<AxisID>=<uint> LF}

<AxisID>	Axis ID
<uint>	Direction sensing reference switch available? (BOOL)

Troubleshooting: Illegal axis identifier

TVI? Tell Valid Character Set For Axis Identifiers

Description: Queries permissible characters for axis identifiers.

Format: `TVI?`

Response: <String>

<String> Characters that are permitted for use in axis identifiers

VAR Set Variable Value

Used in: Running the Macros (99)

Description: Sets a variable to a specific value.

Local variables can only be set in macros.
The variable is only in volatile memory (RAM).

Format: `VAR␣<Variable>␣<String>`

Arguments: <Variable> Name of the variable whose value is set

<String> Value, that the variable is set to
Can be specified directly or via the value of a variable.

VAR? Get Variable Values

Used in: Running the Macros (99)

Description: Queries the value of a variable.

Local variables can only be queries when a macro is running that contains local variables.

Format: `VAR?[{␣<Variable>}]`

Arguments: <Variable> Name of the variable being queried

All variables are queried if no arguments are specified.

Response: {<Variable>=<String> LF}

<Variable> Name of the variable

<String> Value of the variable

VEL Set Closed-Loop Velocity

Used in: Variables for Generating a Dynamics Profile (31)

Description: Sets the velocity for the specified axis.

Changes the value of the parameter 0x49 (p. 202) in the volatile memory. The maximum value that can be set with VEL, is specified by parameter 0xA (p. 192).

The setting only takes effect when the specified axis is in closed-loop operation (servo mode ON). For fast step mode in PiezoMike mode, the value that is set is not evaluated.

VEL? Get Closed-Loop Velocity

Used in: Variables for Generating a Dynamics Profile (31)

Description: Gets the commanded velocity.

Format: `VEL?[{␣<AxisID>}]`

Arguments: <AxisID> Axis ID

The value for all axes will be queried if no arguments are specified.

Response: {<AxisID>=<float>LF}

<AxisID> Axis ID

<float> Velocity in physical units / s (FLOAT)

Troubleshooting: Illegal axis identifier

VER? Get Versions Of Firmware And Drivers

Used in: Important Firmware Components (21)

Description: Queries the version numbers of the firmware.

VER? also queries the version numbers of further components such as drivers and libraries.

Format: `VER?`

Response: {<string1>:␣<string2>LF}

<string1> Name of the component

<string2> Version information and optional specifications

WAC Wait For Condition

Used in: Analog Input Signals (89), Digital Input Signals (86), Running the Macros (99)

Description: Waits until a condition is met.

WAC compares a specified value with a queried value according to a specified rule.

Can only be used in macros.

Possible relational operators:

<OP>	Description
=	Equal
!=	Not equal
<=	Smaller than or equal
<	Smaller than
>=	Larger than or equal
>	Larger than

Format: `WAC␣<CMD?>␣<OP>␣<Value>`

Arguments:

<CMD?>	Query command that responds with a single value
<OP>	Relational operator
<Value>	Relational value with <CMD?>

WPA Save Parameters To Non-Volatile Memory

Used in: Parameter Commands (108)

Description:

Writes the value of a parameter from the volatile memory (RAM) to the nonvolatile memory.

WPA can also save parameter-independent settings. The used password determines what is saved with WPA:

<Pswd>	Description
100	Saves current valid values of all parameters and the current valid settings for HDT (p. 141), HIA (p. 143), and HIT (p. 147).
101	Saves the currently valid values of all parameters.
HID	Saves the current valid settings for HDT (p. 141), HIA (p. 143), and HIT (p. 147).

Wrong values can lead to faulty operation or damage to the hardware.

Notice: Note that the number of write cycles in the nonvolatile memory is limited. Therefore, save to the nonvolatile memory only when necessary.

Format:

```
WPA␣<Pswd>[{␣<ItemID>␣<PamID>}]
```

Arguments:

<Pswd>	Password for writing to the nonvolatile memory
<ItemID>	Element of the electronics
<PamID>	Parameter ID

The <ItemID> and <PamID> specifications are not supported by all electronics.

Troubleshooting:

Impermissible element identifier

Wrong parameter ID

Wrong password

10 Parameter Reference

0x1	P term	Proportional constant of the PID servo algorithm. Is used for fast correction of the position error. 0 to 65535
0x2	I term	Integration constant of the PID servo algorithm. Used for reducing static position error. 0 to 65535
0x3	D term	Differential constant of the PID servo algorithm. Used for damping rapid control oscillation. The D term can be calculated as a floating average over several servo cycles. The parameter 0x71 (D-Term Delay) specifies how many values (i.e., servo cycles) are to be used for calculating the average. 0 to 65535 The preset value of this parameter should not be changed.
0x4	I limit	Limit of the integration constant. 0 to 1048575
0x5	Kvff	Feed-forward control of the commanded velocity. Used for minimizing positioning error. 0 to 65535
0x8	Maximum Position Error (Phys. Unit)	Maximum position error. A motion error occurs when the difference between the current position and the commanded position exceeds this value and the movement of the axis is stopped.
0x9	Maximum Motor Output	Maximum control value for driving an axis respectively a channel. See the response to HPA? for possible values
0xA	Maximum Closed-Loop Velocity (Phys. Unit/s)	Maximum velocity with dynamics profile in closed-loop operation. Specifies the maximum value for parameter 0x49.
0xB	Closed-Loop Acceleration (Phys. Unit/s^2)	Acceleration with dynamics profile in closed-loop operation. 0 to value of 0x4A [phys. units. / s2]
0xC	Closed-Loop Deceleration (Phys. Unit/s^2)	Deceleration with dynamics profile in closed-loop operation. 0 to value of 0x4B [phys. units. / s2]

0xE	Numerator Of The Counts-Per-Physical-Unit Factor	Numerator of the factor for converting impulses to physical units. 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. 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. Decimal number
0xF	Denominator Of The Counts-Per-Physical-Unit Factor	Denominator of the factor for converting impulses to physical units. 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. 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. Decimal number
0x13	Is Rotary Stage?	Is this a rotation stage? Is not evaluated by the electronics but instead by the PC software. 0 Not a rotation stage 1 Rotation stage
0x14	Has Reference?	Do the mechanics have a reference switch? Activates respectively deactivates reference moves to the installed reference switch. 0 No reference switch 1 Reference switch available
0x15	Maximum Travel In Positive Direction (Phys. Unit)	Soft limit in positive direction, in relation to the zero position. The positive travel range limit cannot be used for reference moves if this value is smaller than the position value for the positive travel range limit (that results from the sum of parameters 0x16 and 0x2F). The value can be negative.
0x16	Value At Reference Position (Phys. Unit)	Position value at the reference switch. The current position is set to this value if the axis has performed a reference move to thereference switch. The parameter value is also used for calculating the position values set after reference moves to the limit switches; this also applies when the mechanics do not have a reference switch.

0x17	Distance From Negative Limit To Reference Position (Phys. Unit)	Gap between reference switch and negative travel range limit. The current position is set to the difference between the values of parameters 0x16 and 0x17 if the axis has done a reference move to the negative travel range limit.
0x18	Limit Mode	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
0x1B	Profile Mode	Type of dynamics profile
0x2F	Distance From Reference Position To Positive Limit (Phys. Unit)	Gap between the reference switch and the positive travel range limit. If the axis has done a reference move to the positive travel range limit, the current position is set to the sum of the values of parameters 0x16 and 0x2F.
0x30	Maximum Travel In Negative Direction (Phys. Unit)	Soft limit in a negative direction, in relation to the zero position. The negative travel range limit cannot be used for reference moves if this value is greater than the position value for the negative travel range limit (that results from the difference between parameters 0x16 and 0x17). The value can be negative.
0x31	Invert Reference?	Should the reference signal be inverted? Inverts the signal of the reference switch or a digital input that is used instead of the reference switch. 0 Reference signal not inverted 1 Reference signal inverted
0x32	Has No Limit Switches?	Do the mechanics not have limit switches? Activates motion stop at the installed limit switches. 0 Mechanics have limit switches 1 Mechanics do not have limit switches
0x33	Motor Offset Positive	Drive offset for the positive direction of motion. 0 to 32767
0x34	Motor Offset Negative	Drive offset for the negative direction of motion. 0 to 32767

0x36	Settling Window (Encoder Counts)	Settling window around the target position. Presets the window limits (half of the window width). If the current position enters the settling window, the target position is considered as reached. If the current position exits the settling window, the target position is no longer considered as reached. Can be changed only if servo mode is switched off. 0 to maximum value. The maximum value depends on the controller but may not exceed 231 encoder counts.
0x3C	Stage Name	Positioner name. Default value: NOSTAGE String up to 20 characters
0x3F	Settling Time (s)	Delay time for setting the on-target state. Recommended values: 0.000 to 1.000 seconds
0x47	Reference Travel Direction	Default direction for the reference move. 0 Automatic detection 1 Negative direction 2 Positive direction
0x48	Motor Drive Offset	Velocity-dependent drive offset. Used if the commanded velocity does not equal zero (i.e., if the end of the dynamics profile has not been reached). 0 to 32767
0x49	Closed-Loop Velocity (Phys. Unit/s)	Velocity in closed-loop operation with dynamics profile. 0 to value of 0xA [phys. units / s]
0x4A	Maximum Closed-Loop Acceleration (Phys. Unit/s2)	Maximum acceleration in closed-loop operation with dynamics profile. Specifies the maximum value for parameter 0xB.
0x4B	Maximum Closed-Loop Deceleration (Phys. Unit/s2)	Maximum deceleration in closed-loop operation with dynamics profile. Specifies the maximum value for parameter 0xC.
0x50	Velocity For Reference Moves (Phys. Unit/s)	Maximum velocity for reference moves. 0 to value of value 0x49 [phys. units / s]
0x56	Sensor Power Supply	Supply voltage for encoder activated? This parameter can only be set to 0 in open-loop operation. The servo cannot be switched on as long as the value is 0. The axis is considered to be not referenced if the value of the parameter is 0.
0x5A	Numerator Of The Servo-Loop Input Factor	Input factor numerator for the servo loop. Decimal number
0x5B	Denominator Of The Servo-Loop Input Factor	Input factor denominator for the servo loop. Decimal number

0x5C	Source Of Reference Signal	Reference signal source for axis motion to the reference switch.
0x5D	Source Of Negative Limit Signal	Reference signal source for axis motion to the negative travel range limit.
0x5E	Source Of Positive Limit Signal	Reference signal source for axis motion to the positive travel range limit.
0x5F	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.
0x60	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.
0x61	Invert Direction Of Motion For Joystick-Controlled Axis?	Inverts the direction of motion for HID-controlled axes. 0 Direction of motion not inverted (default) 1 Direction of motion inverted
0x63	Distance Between Limit And Hard Stop (Phys. Unit)	Gap between internal limit switch and hard stop. Determines the maximum stopping distance during reference moves. The actual velocities during a reference move are calculated on the basis of this value, the set deceleration (0xC) and set velocities (0x49 and 0x50).
0x70	Reference Signal Type	Reference signal type.
0x71	D-Term Delay (No. Of Servo Cycles)	D term delay. Determines how many values (i.e., servo cycles) are used for calculating the mean value of the D term. 0 to 15. The preset value of this parameter should not be changed.
0x72	Ignore Macro Error?	Ignore macro error? 0 Stop macro on error (default) 1 Ignore error
0x77	Use Limit Switches Only For Reference Moves?	Should the limit switches only be used for reference moves? Is intended for use with rotation stages. Only evaluated when parameter 0x32 has the value 0.
0x78	Distance From Limit To Start Of Ref Search (Phys. Unit)	Distance between the limit switch and the starting position for the reference move to the index pulse. Used for FRF when parameter 0x70 has the value 2.
0x79	Distance For Reference Search (Phys. Unit)	Maximum distance for motion to the index pulse.
0x7A	Use Hard Stops For Referencing?	Should the hard stops be used for reference moves?

0x94	Notch Filter Frequency 1 (Hz)	Frequency of the first notch filter. The appropriate frequency component is reduced in the control value to compensate for undesired resonances in the mechanics. 40 to 20,000 [Hz]
0x95	Notch Filter Edge 1	Edge steepness of the first notch filter. The greater the value of this parameter, the narrower the notch filter bandwidth. 0.1 to 10 (dimensionless)
0x3003300	Sensor Interpolation	Interpolation rate for the signals of the incremental sensor.
0x3003301	Sensor Hysteresis (Deg)	Correction of the incremental sensor hysteresis.
0x3003302	Sensor Digital Gain	Gain value for correcting the digitized signals of the incremental sensor.
0x3003303	Sensor Digital Offset 0 (V)	Offset 0 for correcting the digitized signals of the incremental sensor.
0x3003304	Sensor Digital Offset 1 (V)	Offset 1 for correcting the digitized signals of the incremental sensor.
0x3003305	Sensor Digital Phase (Deg)	Phase correction for the signals of the incremental sensor.
0x3003306	Sensor Analog Gain (dB)	Gain value for correcting the analog signals of the incremental sensor.
0x3003307	Sensor Analog Offset 0 (V)	Offset 0 for correcting the analog signals of the incremental sensor.
0x3003308	Sensor Analog Offset 1 (V)	Offset 1 for correcting the analog signals of the incremental sensor.
0x3003320	Sensor Signal Type	Signal type output by the position sensor.
0x7000601	Axis Unit	Unit symbol. 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". String up to 20 characters
0xD000000	Controller Device S/N	Serial number of the electronics.
0xD000200	Hardware Name	Product name
0xE000102	Number of Decimal Places	Number of decimal places for floating point numbers.
0xE000200	Servo Update Time	Servo cycle time.
0xE000B01	Number of output channels	Number of drive channels of the controller
0xF000100	Stage Type	Mechanics type. x-xxx default positioners x-xxxKxxx customized positioners
0xF000200	Stage Serial Number	Serial number of the mechanics. 9-digit number

0xF000300	Stage Assembly Date	Manufacturing date of the mechanics. Date in DDMMYY format
0xF000400	Stage HW Version	Version number of the mechanics hardware.
0x10000500	Fast IF Axis Input Usage	Use of input data that arrives at the E-873.1AT via data segment 1 of the SPI interface.
0x10000501	Fast IF Data Type	Selection of the data type for the data that is sent via data segment 1 of the SPI interface. If the FLOAT data type is selected, the data can be written and read in the unit of the axis directly. Scaling is then not necessary. If an UINT data type is selected, scaling is necessary. In this case, the parameters must also be set for the lower and upper limit of the value range (Fast IF Data Low Limit 0x10000502, Fast IF Data High Limit 0x010000503).
0x10000502	Fast IF Data Low Limit	Lower limit value for the data that is sent via data segment 1 of the SPI interface. Required for internal data scaling if an UINT data type is selected (Fast IF Data Type 0x10000501). The limit value is not evaluated for the FLOAT data type.
0x10000503	Fast IF Data High Limit	Upper limit value for the data that is sent via data segment 1 of the SPI interface. Required for internal data scaling if an UINT data type is selected (Fast IF Data Type 0x10000501). The limit value is not evaluated for the FLOAT data type.
0x16000001	Recorded Points Per Trigger	Number of data points recorded per trigger impulse. 0 Unlimited (default) n>0 n data points are recorded Maximum value for n at E-873.1AT: 7616
0x1F000000	PIShift Upper Supply Voltage (V)	Maximum output voltage for piezo inertia drives. The level of the voltage depends on the connected positioner and is automatically set when the corresponding set of parameters is loaded from the positioner database. A wrong setting of this value can present a danger for the user and damage the connected positioner. Therefore, do not change the automatically set value! The value depends on the type of the drive.
0x1F000100	PIShift Lower Supply Voltage (V)	Minimum output voltage for piezo inertia drives. The value depends on the type of the drive.
0x1F000200	PIShift Forward Current (A)	Maximum output current for piezo inertia drives during forward motion. The value depends on the type of the drive.
0x1F000300	PIShift Backward Current (A)	Maximum output current for piezo inertia drives during backward motion. The value depends on the type of the drive.

0x1F000400	PIShift Frequency (Hz)	Frequency of the piezo voltage for open-loop operation of piezo inertia drives. Determines the velocity of a drive in open-loop operation. This parameter must not exceed the value of the parameter 0x9 (Maximum Motor Output).
0x1F000700	PIShift Step Size (Phys. Unit)	Step size in closed-loop operation
0x1F000701	PIShift Delay (ms)	Delay time when switching between fast step (multi-step) mode and single-step mode of motion in PiezoMike mode. 0.05 to 100 [ms]
0x1F000702	PIShift Open-Loop Driving Mode	Drive mode for piezo inertia drives in open-loop operation. This parameter is evaluated when motion is started in open-loop operation. The value of the parameter has no influence on motion started with the STE command in open-loop operation.
0x1F000703	PIShift Offset Current (A)	Offset current in open-loop operation - 0.05 to 0.05 The preset value should not be changed.
0x1F000711	PIShift P Term, Silent Mode	Proportional constant of the PI servo algorithm for the silent mode of Q-Motion® drives. The proportional constant is used for fast correction of the position error. This parameter will only be evaluated if 0x1F0007A3 is set to 2 (silent mode). 0 to 65535
0x1F000712	PIShift I Term, Silent Mode	Integration constant of the PI servo algorithm for the silent mode of Q-Motion® drives. The integration constant is used for reducing static position errors. This parameter will only be evaluated if 0x1F0007A3 is set to 2 (silent mode). 0 to 65535
0x1F0007A0	PIShift Plateau And Slip Time (µs)	Delay time between the stick and the slip phases of the piezo inertia drive. Depending on the velocity, minimal overshooting of the piezo actuator occurs at the end of the stick and the slip phases. The delay time gives the piezo actuator time to compensate for overshooting before the next phase of motion begins. This results in smoother motion.
0x1F0007A3	PIShift Closed-Loop Driving Mode	Drive mode in closed-loop operation In order to adapt motion in closed-loop operation to different requirements, different modes can be set via this parameter: 0 - Q-Motion® mode 1 - PiezoMike mode 2 - Silent mode for Q-Motion®

0x1F0007A4	PiezoMike Enter Window (Phys. Unit)	Entrance window around the target position for motion in closed-loop operation in PiezoMike mode. Commanded movement will be continued until the entrance window around the target position has been reached. The target position is then considered as reached and the on-target signal is output, if necessary, after a delay time (parameter 0x3F). This parameter will only be evaluated if 0x1F0007A3 is set to 1 (PiezoMike mode).
0x1F0007A5	PiezoMike Exit Window (Phys. Unit)	Exit window around the target position for motion in closed-loop operation in PiezoMike mode. Defines a tolerance range around the entrance window surrounding the target position (parameter 0x1F0007A4) to maintain the on-target signal after a movement. If the current position leaves the exit window, the on-target signal will no longer be output and the system will try to reach the target position once again. This parameter will only be evaluated if 0x1F0007A3 is set to 1 (PiezoMike mode).
0x1F0007A7	PiezoMike Single Step Enter Factor	Entrance window for single-step mode in PiezoMike mode for motion in closed-loop operation. Commanded movement is done in fast step mode (multi-step) until the entrance window for single-step mode has been reached. Motion continues in single-step mode until the entrance window around the target position (parameter 0x1F0007A4) has been reached. This parameter will only be evaluated if 0x1F0007A3 is set to 1 (PiezoMike mode). Specified as entrance window factor around the target position
0x1F0007A8	PiezoMike Single Step Exit Factor	Exit window for single-step mode in PiezoMike mode for motion in closed-loop operation. Defines a tolerance range around the entrance window for single-step mode (parameter 0x1F0007A7) to maintain single-step mode. When the current position leaves the exit window, the system switches to fast step mode (multi step) to reach the target position. This parameter will only be evaluated if 0x1F0007A3 is set to 1 (PiezoMike mode). Specified as exit window factor around the target position (parameter 0x1F0007A5)

0x1F0007C2	PIShift Threshold DeltaPos For Hardstop (Phys. Unit)	Maximum position error for certain period of time. If the difference between the commanded and actual position is greater than this value for a certain period of time, a motion error is triggered and stops axis motion. Difference to parameter 0x8 (Maximum Position Error): The maximum position error triggers a motion error immediately and stops the movement of the axis.

0x1 P Term

Used in: Settings for the servo algorithm (35)

Description:	Proportional constant of the PID servo algorithm. Is used for fast correction of the position error.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 to 65535

0x2 I Term

Used in: Settings for the servo algorithm (35)

Description:	Integration constant of the PID servo algorithm. Used for reducing static position error.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 to 65535

0x3 D Term

Used in: Settings for the servo algorithm (35)

Description:	Differential constant of the PID servo algorithm. Used for damping rapid control oscillation. The D term can be calculated as a floating average over several servo cycles. The parameter 0x71 (p. 209) (**D-Term Delay**) specifies how many values (i.e., servo cycles) are to be used for calculating the average.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 to 65535 The preset value of this parameter should not be changed.

0x4 I Limit

Used in: Settings for the servo algorithm (35)

Description:	Limit of the integration constant.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 to 1048575

0x5 Kvff

Description:	Feed-forward control of the commanded velocity. Used for minimizing positioning error.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 to 65535

0x8 Maximum Position Error (Phys. Unit)

Used in: Behavior with Motion Error (78)

Description:	Maximum position error. A motion error (p. 78) occurs when the difference between the current position and the commanded position exceeds this value and the movement of the axis is stopped.
Data type	FLOAT
Command level	0
Item type	Axis
Source of data	Positioner database

0x9 Maximum Motor Output

Description:	Maximum control value for driving an axis respectively a channel.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	See the response to HPA? for possible values

0xA Maximum Closed-Loop Velocity (Phys. Unit/s)

Used in: Variables for Generating a Dynamics Profile (31)

Description:	Maximum velocity with dynamics profile in closed-loop operation. Specifies the maximum value for parameter 0x49 (p. 202).
Data type	FLOAT
Command level	0
Item type	Axis
Source of data	Positioner database

0xB Closed-Loop Acceleration (Phys. Unit/s²)

Used in: Variables for Generating a Dynamics Profile (31)

Description: Acceleration with dynamics profile in closed-loop operation.

Limited by parameter 0x4A (p. 202). For fast step mode in PiezoMike mode, the value that is set is not evaluated.

0xC Closed-Loop Deceleration (Phys. Unit/s²)

Used in: Reference Move Procedure (44), Variables for Generating a Dynamics Profile (31)

Description: Deceleration with dynamics profile in closed-loop operation.

Limited by parameter 0x4B (p. 202). For fast step mode in PiezoMike mode, the value that is set is not evaluated.

0xE Numerator Of The Counts-Per-Physical-Unit Factor

Used in: Physical Units (26)

Description:	Numerator of the factor for converting impulses to physical units. 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. 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.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	Decimal number

0xF Denominator Of The Counts-Per-Physical-Unit Factor

Used in: Physical Units (26)

Description:	Denominator of the factor for converting impulses to physical units. 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. 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.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	Decimal number

0x13 Is Rotary Stage?

Description:	Is this a rotation stage? Is not evaluated by the electronics but instead by the PC software.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 Not a rotation stage 1 Rotation stage

0x14 Has Reference?

Used in: Reference Switch Detection (41), Using Digital Input as Reference Signal (87)

Description:	Do the mechanics have a reference switch? Activates respectively deactivates reference moves to the installed reference switch.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 No reference switch 1 Reference switch available

0x15 Maximum Travel In Positive Direction (Phys. Unit)

Used in: Settings for Soft Limits (42)

Description:	Soft limit in positive direction, in relation to the zero position. The positive travel range limit cannot be used for reference moves if this value is smaller than the position value for the positive travel range limit (that results from the sum of parameters 0x16 (p. 195) and 0x2F (p. 197)).
Data type	FLOAT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	The value can be negative.

0x16 Value At Reference Position (Phys. Unit)

Used in: Settings for the travel range (42)

Description:	Position value at the reference switch. The current position is set to this value if the axis has performed a reference move to thereference switch. The parameter value is also used for calculating the position values set after reference moves to the limit switches; this also applies when the mechanics do not have a reference switch.
Data type	FLOAT
Command level	0
Item type	Axis
Source of data	Positioner database

0x17 Distance From Negative Limit To Reference Position (Phys. Unit)

Used in: Settings for the travel range (42)

Description:	Gap between reference switch and negative travel range limit. The current position is set to the difference between the values of parameters 0x16 (p. 195) and 0x17 (p. 196) if the axis has done a reference move to the negative travel range limit.
Data type	FLOAT
Command level	0
Item type	Axis
Source of data	Positioner database

0x18 Limit Mode

Used in: Detecting Limit Switches (41)

Description:	Signal logic of the limit switches.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 pos-HI, neg-HI 1 pos-LO, neg-HI 2 pos-HI, neg-LO 3 pos-LO, neg-LO

0x1B Profile Mode

Description:	Type of dynamics profile
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 Trapezoidal dynamics profile

0x2F Distance From Reference Position To Positive Limit (Phys. Unit)

Used in: Settings for the travel range (42)

Description:	Gap between the reference switch and the positive travel range limit. If the axis has done a reference move to the positive travel range limit, the current position is set to the sum of the values of parameters 0x16 (p. 195) and 0x2F (p. 197).
Data type	FLOAT
Command level	0
Item type	Axis
Source of data	Positioner database

0x30 Maximum Travel In Negative Direction (Phys. Unit)

Used in: Settings for Soft Limits (42)

Description: Soft limit in a negative direction, in relation to the zero position.

The negative travel range limit cannot be used for reference moves if this value is greater than the position value for the negative travel range limit (that results from the difference between parameters 0x16 (p. 195) and 0x17 (p. 196)).

Data type: FLOAT

Command level: 0

Item type: Axis

Source of data: Positioner database

Possible values: The value can be negative.

0x31 Invert Reference?

Used in: Reference Switch Detection (41), Using Digital Input as Reference Signal (87)

Description: Should the reference signal be inverted?

Inverts the signal of the reference switch or a digital input that is used instead of the reference switch.

Data type: INT

Command level: 0

Item type: Axis

Source of data: Positioner database

Possible values:

0 Reference signal not inverted

1 Reference signal inverted

0x32 Has No Limit Switches?

Used in: Detecting Limit Switches (41), Using Digital Inputs as Source of the Limit Switch Signals (87)

Description:	Do the mechanics not have limit switches? Activates motion stop at the installed limit switches.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 Mechanics have limit switches 1 Mechanics do not have limit switches

0x33 Motor Offset Positive

Description:	Drive offset for the positive direction of motion.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 to 32767

0x34 Motor Offset Negative

Description:	Drive offset for the negative direction of motion.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 to 32767

0x36 Settling Window (Encoder Counts)

Used in: On-Target State (39)

Description:	Settling window around the target position. Presets the window limits (half of the window width). If the current position enters the settling window, the target position is considered as reached. If the current position exits the settling window, the target position is no longer considered as reached. Can be changed only if servo mode is switched off.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 to maximum value. The maximum value depends on the controller but may not exceed 2^{31} encoder counts.

0x3C Stage Name

Used in: Commandable Items (23)

Description:	Positioner name. Default value: NOSTAGE
Data type	CHAR
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	String up to 20 characters

0x3F Settling Time (s)

Used in: On-Target State (39)

Description:	Delay time for setting the on-target state.
Data type	FLOAT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	Recommended values: 0.000 to 1.000 seconds

0x47 Reference Travel Direction

Used in: Reference Move Procedure (44)

Description:	Default direction for the reference move.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 Automatic detection 1 Negative direction 2 Positive direction

0x48 Motor Drive Offset

Description:	Velocity-dependent drive offset. Used if the commanded velocity does not equal zero (i.e., if the end of the dynamics profile has not been reached).
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 to 32767

0x49 Closed-Loop Velocity (Phys. Unit/s)

Used in: Reference Move Procedure (44), Variables for Generating a Dynamics Profile (31)

Description: Velocity in closed-loop operation with dynamics profile.

Limited by parameter 0xA (p. 192). For fast step mode in PiezoMike mode, the value that is set is not evaluated.

0x4A Maximum Closed-Loop Acceleration (Phys. Unit/s^2)

Used in: Variables for Generating a Dynamics Profile (31)

Description:	Maximum acceleration in closed-loop operation with dynamics profile. Specifies the maximum value for parameter 0xB (p. 192).
Data type	FLOAT
Command level	0
Item type	Axis

Source of data

Positioner database

0x4B Maximum Closed-Loop Deceleration (Phys. Unit/s^2)

Used in: Variables for Generating a Dynamics Profile (31)

Description:	Maximum deceleration in closed-loop operation with dynamics profile. Specifies the maximum value for parameter 0xC (p. 192).
Data type	FLOAT
Command level	0
Item type	Axis
Source of data	Positioner database

0x50 Velocity For Reference Moves (Phys. Unit/s)

Used in: Reference Move Procedure (44)

Description:	Maximum velocity for reference moves.
Data type	FLOAT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 to value of value 0x49 (p. 202) [phys. units / s]

0x56 Sensor Power Supply

Description:	Supply voltage for encoder activated? This parameter can only be set to 0 in open-loop operation. The servo cannot be switched on as long as the value is 0. The axis is considered to be not referenced if the value of the parameter is 0.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	

0	Supply voltage deactivated
1	Supply voltage activated (default setting)

0x5A Numerator Of The Servo-Loop Input Factor

Used in: Settings for the servo algorithm (35)

Description:	Input factor numerator for the servo loop.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	Decimal number

0x5B Denominator Of The Servo-Loop Input Factor

Used in: Settings for the servo algorithm (35)

Description:	Input factor denominator for the servo loop.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	Decimal number

0x5C Source Of Reference Signal

Used in: Using Digital Input as Reference Signal (87)

Description:	Reference signal source for axis motion to the reference switch.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	

0	Reference switch
> 0: <n>	<n> = ID of the digital input line, bit-mapped

0x5D Source Of Negative Limit Signal

Used in: Using Digital Inputs as Source of the Limit Switch Signals (87)

Description:	Reference signal source for axis motion to the negative travel range limit.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	

0	Negative limit switch
> 0: <n>	<n> = ID of the digital input line, bit-mapped

0x5E Source Of Positive Limit Signal

Used in: Using Digital Inputs as Source of the Limit Switch Signals (87)

Description:	Reference signal source for axis motion to the positive travel range limit.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	

0	Positive limit switch
> 0: <n>	<n> = ID of the digital input line, bit-mapped

0x5F Invert Digital Input Used For Negative Limit

Used in: Using Digital Inputs as Source of the Limit Switch Signals (87)

Description: Inverts the polarity of the digital inputs that are used as the source of the negative limit switch signal.

Data type INT

Command level 0

Item type Axis

Source of data Positioner database

Possible values

0	Digital input not inverted
> 0: <n>	<n> = ID of the digital input line, bit-mapped

0x60 Invert Digital Input Used For Positive Limit

Used in: Using Digital Inputs as Source of the Limit Switch Signals (87)

Description: Inverts the polarity of the digital inputs that are used as the source of the positive limit switch signal.

Data type INT

Command level 0

Item type Axis

Source of data Positioner database

Possible values

0	Digital input not inverted
> 0: <n>	<n> = ID of the digital input line, bit-mapped

0x61 Invert Direction Of Motion For Joystick-Controlled Axis?

Used in: HID Control Configuration (90)

Description:	Inverts the direction of motion for HID-controlled axes.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 Direction of motion not inverted (default) 1 Direction of motion inverted

0x63 Distance Between Limit And Hard Stop (Phys. Unit)

Used in: Reference Move Procedure (44)

Description:	Gap between internal limit switch and hard stop. Determines the maximum stopping distance during reference moves. The actual velocities during a reference move are calculated on the basis of this value, the set deceleration (0xC (p. 192)) and set velocities (0x49 (p. 202) and 0x50 (p. 203)).
Data type	FLOAT
Command level	0
Item type	Axis
Source of data	Positioner database

0x70 Reference Signal Type

Used in: Reference Move Procedure (44), Reference Switch Detection (41), Referencing options (43)

Description:	Reference signal type.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	

0	Direction-sensing reference switch The signal level changes when passing the reference switch.
1	Pulse signal (pulse width of several nanoseconds) Parameter 0x47 (p. 201) must be correctly set.
2	Index pulse The reference switch is approached via the negative travel range limit.
3	Index pulse The reference switch is approached via the positive travel range limit.
4	No reference signal Is only evaluated by the electronics that do not support the FNL and FPL commands.
5	The reference move is made to the negative limit switch or hard stop; this is set as reference position. Is only evaluated by the electronics that do not support the FNL and FPL commands.
6	The reference move is made to the positive limit switch or hard stop; this is set as reference position. Is only evaluated by the electronics that do not support the FNL and FPL commands.

0x71 D-Term Delay (No. Of Servo Cycles)

Used in: Settings for the servo algorithm (35)

Description:	D term delay. Determines how many values (i.e., servo cycles) are used for calculating the mean value of the D term.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 to 15. The preset value of this parameter should not be changed.

0x72 Ignore Macro Error?

Used in: (99)

Description:	Ignore macro error?
Data type	INT
Command level	0
Item type	System
Source of data	PC software commands (SPA (p. 168), SEP (p. 167)) or operating elements
Possible values	0 Stop macro on error (default) 1 Ignore error

0x77 Use Limit Switches Only For Reference Moves?

Used in: Detecting Limit Switches (41)

Description:	Should the limit switches only be used for reference moves? Is intended for use with rotation stages. Only evaluated when parameter 0x32 (p. 199) has the value 0.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	

0	Use limit switches for stopping at the end of the travel range and for reference moves (default)
1	Use limit switches for reference moves only

0x78 Distance From Limit To Start Of Ref Search (Phys. Unit)

Used in: Reference Move Procedure (44)

Description:	Distance between the limit switch and the starting position for the reference move to the index pulse. Used for FRF (p. 139) when parameter 0x70 (p. 208) has the value 2.
Data type	FLOAT
Command level	0
Item type	Axis
Source of data	Positioner database

0x79 Distance For Reference Search (Phys. Unit)

Used in: Reference Move Procedure (44)

Description:	Maximum distance for motion to the index pulse.
Data type	FLOAT
Command level	0
Item type	Axis
Source of data	Positioner database

0x7A Use Hard Stops For Referencing?

Used in: Referencing options (43)

Description:	Should the hard stops be used for reference moves?
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	

0	Do not use hard stops for reference moves
1	Use hard stops for reference moves

0x94 Notch Filter Frequency 1 (Hz)

Used in: Settings for the Notch Filter (36)

Description:	Frequency of the first notch filter. The appropriate frequency component is reduced in the control value to compensate for undesired resonances in the mechanics.
Data type	FLOAT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	40 to 20,000 [Hz]

0x95 Notch Filter Edge 1

Used in: Settings for the Notch Filter (36)

Description:	Edge steepness of the first notch filter. The greater the value of this parameter, the narrower the notch filter bandwidth.
Data type	FLOAT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0.1 to 10 (dimensionless)

0x3003300 Sensor Interpolation

Description:	Interpolation rate for the signals of the incremental sensor.
Data type	FLOAT
Command level	2
Item type	Axis
Source of data	ID chip of the mechanics

0x3003301 Sensor Hysteresis (Deg)

Description:	Correction of the incremental sensor hysteresis.
Data type	FLOAT
Command level	depending on the controller
Item type	Axis
Source of data	ID chip of the mechanics, if supported by the controller

0x3003302 Sensor Digital Gain

Description:	Gain value for correcting the digitized signals of the incremental sensor.
Data type	FLOAT
Command level	2
Item type	Axis
Source of data	ID chip of the mechanics

0x3003303 Sensor Digital Offset 0 (V)

Description:	Offset 0 for correcting the digitized signals of the incremental sensor.
Data type	FLOAT
Command level	2
Item type	Axis
Source of data	ID chip of the mechanics

0x3003304 Sensor Digital Offset 1 (V)

Description:	Offset 1 for correcting the digitized signals of the incremental sensor.
Data type	FLOAT
Command level	2
Item type	Axis
Source of data	ID chip of the mechanics

0x3003305 Sensor Digital Phase (Deg)

Description:	Phase correction for the signals of the incremental sensor.
Data type	FLOAT
Command level	2
Item type	Axis
Source of data	ID chip of the mechanics

0x3003306 Sensor Analog Gain (dB)

Description:	Gain value for correcting the analog signals of the incremental sensor.
Data type	FLOAT
Command level	2
Item type	Axis
Source of data	ID chip of the mechanics

0x3003307 Sensor Analog Offset 0 (V)

Description:	Offset 0 for correcting the analog signals of the incremental sensor.
Data type	FLOAT
Command level	2
Item type	Axis
Source of data	ID chip of the mechanics

0x3003308 Sensor Analog Offset 1 (V)

Description:	Offset 1 for correcting the analog signals of the incremental sensor.
Data type	FLOAT
Command level	2
Item type	Axis
Source of data	ID chip of the mechanics

0x3003320 Sensor Signal Type

Description:	Signal type output by the position sensor.
Data type	INT
Command level	1
Item type	Axis
Source of data	ID chip of the mechanics

Possible values

0	No sensor
1	Incremental sensor, AB quadrature signals
2	Incremental sensor, sin/cos signals
3	Incremental sensor, BiSS (32 bit)
4	Absolute-measuring sensor, BiSS (32 bit)
5	Incremental sensor, BiSS (24 bit) and absolute-measuring sensor, BiSS (32 bit)
6	Absolute-measuring sensor, BISS (32 bit) and incremental sensor, BiSS (24 bit)
7	Incremental sensor, BiSS (24 bit)
8	Absolute-measuring sensor, BiSS (24 bit)

0x7000601 Axis Unit

Used in: Physical Units (26)

Description:	Unit symbol. For example, the unit symbol is "mm", if the factor for the counts per physical unit of length is set with parameters 0xE (p. 193) and 0xF (p. 193) so that the encoder counts are converted into millimeters. The unit for rotation stages is normally "deg".
Data type	CHAR
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	String up to 20 characters

0xD000000 Controller Device S/N

Description:	Serial number of the electronics.
Data type	CHAR
Command level	2
Item type	System

0xD000200 Hardware Name

Description:	Product name
Data type	CHAR
Command level	3
Item type	System

0xE000102 Number of Decimal Places

Used in: Physical Units (26)

Description:	Number of decimal places for floating point numbers.
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database

0xE000200 Servo Update Time

Description:	Servo cycle time.
Data type	FLOAT
Command level	2
Item type	System
Source of data	PC software commands (SPA (p. 168), SEP (p. 167)) or operating elements

0xE000B01 Number of Output Channels

Description:	Number of drive channels of the controller
Data type	INT
Command level	3
Item type	System
Source of data	Factory default setting

0xF000100 Stage Type

Description:	Mechanics type.
Data type	CHAR
Command level	2
Item type	Axis
Source of data	ID chip of the mechanics
Possible values	x-xxx default positioners x-xxxKxxx customized positioners

0xF000200 Stage Serial Number

Description:	Serial number of the mechanics.
Data type	CHAR
Command level	2
Item type	Axis
Source of data	ID chip of the mechanics
Possible values	9-digit number

0xF000300 Stage Assembly Date

Description:	Manufacturing date of the mechanics.
Data type	CHAR
Command level	2
Item type	Axis
Source of data	ID chip of the mechanics
Possible values	Date in DDMMYY format

0xF000400 Stage HW Version

Description:	Version number of the mechanics hardware.
Data type	INT
Command level	2
Item type	Axis
Source of data	ID chip of the mechanics

0x10000500 Fast IF Axis Input Usage

Description:	Use of input data that arrives at the E-873.1AT via data segment 1 of the SPI interface.
Data type	INT
Command level	0
Item type	System
Source of data	Positioner database

Possible values

0	Input data is not used
1	Input data is used as target value (closed-loop operation) or control value (open-loop operation) for the axis

0x10000501 Fast IF Data Type

Description:

Selection of the data type for the data that is sent via data segment 1 of the SPI interface.

If the FLOAT data type is selected, the data can be written and read in the unit of the axis directly. Scaling is then not necessary.

If an UINT data type is selected, scaling is necessary. In this case, the parameters must also be set for the lower and upper limit of the value range (**Fast IF Data Low Limit** 0x10000502 (p. 220), **Fast IF Data High Limit** 0x010000503 (p. 221)).

Data type

INT

Command level

0

Item type

System

Source of data

Positioner database

Possible values

0	FLOAT 32 bit
1	UINT 16 bit
2	UINT 24 bit
3	UINT 32 bit

0x10000502 Fast IF Data Low Limit

Description:

Lower limit value for the data that is sent via data segment 1 of the SPI interface.

Required for internal data scaling if an UINT data type is selected (**Fast IF Data Type** 0x10000501 (p. 220)). The limit value is not evaluated for the FLOAT data type.

Data type

FLOAT

Command level

0

Item type

System

Source of data

Positioner database

0x10000503 Fast IF Data High Limit

Description:	Upper limit value for the data that is sent via data segment 1 of the SPI interface. Required for internal data scaling if an UINT data type is selected (**Fast IF Data Type** 0x10000501 (p. 220)). The limit value is not evaluated for the FLOAT data type.
Data type	FLOAT
Command level	0
Item type	System
Source of data	Positioner database

0x16000001 Recorded Points Per Trigger

Used in: Configuring data processing (81)

Description:	Number of data points recorded per trigger impulse.
Data type	INT
Command level	0
Item type	System
Source of data	PC software commands (SPA (p. 168), SEP (p. 167)) or operating elements
Possible values	0 Unlimited (default) n>0 n data points are recorded Maximum value for n at E-873.1AT: 7616

0x1F000000 PIShift Upper Supply Voltage (V)

Used in: (28)

Description: Maximum output voltage for piezo inertia drives.

The level of the voltage depends on the connected positioner and is automatically set when the corresponding set of parameters is loaded from the positioner database. A wrong setting of this value can present a danger for the user and damage the connected positioner. Therefore, do **not** change the automatically set value!

Data type FLOAT

Command level 1

Item type Axis

Source of data Positioner database

Possible values The value depends on the type of the drive.

0x1F000100 PIShift Lower Supply Voltage (V)

Description: Minimum output voltage for piezo inertia drives.

Data type FLOAT

Command level 1

Item type Axis

Source of data Positioner database

Possible values The value depends on the type of the drive.

0x1F000200 PIShift Forward Current (A)

Used in: (28)

Description: Maximum output current for piezo inertia drives during forward motion.

Data type FLOAT

Command level 1

Item type Axis

Source of data Positioner database

Possible values The value depends on the type of the drive.

0x1F000300 PIShift Backward Current (A)

Used in: (28)

Description:	Maximum output current for piezo inertia drives during backward motion.
Data type	FLOAT
Command level	1
Item type	Axis
Source of data	Positioner database
Possible values	The value depends on the type of the drive.

0x1F000400 PIShift Frequency (Hz)

Used in: OSM Open-Loop Step Moving (162)

Description:	Frequency of the piezo voltage for open-loop operation of piezo inertia drives. Determines the velocity of a drive in open-loop operation.
Data type	FLOAT
Command level	1
Item type	Axis
Source of data	Positioner database
Possible values	This parameter must not exceed the value of the parameter 0x9 (p. 191) (**Maximum Motor Output**).

0x1F000700 PIShift Step Size (Phys. Unit)

Description:	Step size in closed-loop operation
Data type	FLOAT
Command level	1
Item type	Axis
Source of data	Positioner database

0x1F000701 PIShift Delay (ms)

Used in: Motion in closed-loop operation (29)

Description:	Delay time when switching between fast step (multi-step) mode and single-step mode of motion in PiezoMike mode.
Data type	FLOAT
Command level	1
Item type	Axis
Source of data	Positioner database
Possible values	0.05 to 100 [ms]

0x1F000702 PIShift Open-Loop Driving Mode

Used in: Servo mode (28)

Description: Drive mode for piezo inertia drives in open-loop operation.

This parameter is evaluated when motion is started in open-loop operation. The value of the parameter has no influence on motion started with the STE (p. 172) command in open-loop operation.

Data type: INT

Command level: 2

Item type: Axis

Source of data: PC software commands (SPA (p. 168), SEP (p. 167)) or operating elements

Possible values:

0	Current Generated: The level of the input voltage determines the frequency for the movement. The drive electronics of the E-873.1AT 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 runner and the piezo actuator and therefore continuous feed of the runner. The output of one period of the modified sawtooth signal generates one "step" for the runner.
1	Constant Frequency: Constant motion at the frequency set with parameter 0x9 (p. 191) (**Maximum Motor Output**). The drive electronics of the E-873.1AT convert the control value to an analog signal. The output piezo voltage corresponds to 10 times this analog signal. The feed to the runner is generated by the expansion of the piezo actuator caused by the piezo voltage. The piezo actuator achieves its maximum expansion when the E-873.1AT outputs the maximum permitted piezo voltage.

0x1F000703 PIShift Offset Current (A)

Used in: (28)

Description:	Offset current in open-loop operation
Data type	FLOAT
Command level	1
Item type	Axis
Source of data	Positioner database
Possible values	- 0.05 to 0.05 The preset value should not be changed.

0x1F000711 PIShift P Term, Silent Mode

Used in: Settings for the servo algorithm (35)

Description:	Proportional constant of the PI servo algorithm for the silent mode of Q-Motion® drives. The proportional constant is used for fast correction of the position error. This parameter will only be evaluated if 0x1F0007A3 (p. 228) is set to 2 (silent mode).
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 to 65535

0x1F000712 PIShift I Term, Silent Mode

Used in: Settings for the servo algorithm (35)

Description:	Integration constant of the PI servo algorithm for the silent mode of Q-Motion® drives. The integration constant is used for reducing static position errors. This parameter will only be evaluated if 0x1F0007A3 (p. 228) is set to 2 (silent mode).
Data type	INT
Command level	0
Item type	Axis
Source of data	Positioner database
Possible values	0 to 65535

0x1F0007A0 PIShift Plateau And Slip Time (µs)

Used in: Working Principle of a Piezo Inertia Drive (27)

Description:	Delay time between the stick and the slip phases of the piezo inertia drive. Depending on the velocity, minimal overshooting of the piezo actuator occurs at the end of the stick and the slip phases. The delay time gives the piezo actuator time to compensate for overshooting before the next phase of motion begins. This results in smoother motion.
Data type	INT
Command level	1
Item type	Axis
Source of data	PC software commands (SPA (p. 168), SEP (p. 167)) or operating elements

0x1F0007A3 PIShift Closed-Loop Driving Mode

Used in: Motion in closed-loop operation (29)

Description:

Drive mode in closed-loop operation

In order to adapt motion in closed-loop operation to different requirements, different modes can be set via this parameter:

0 - Q-Motion® mode

1 - PiezoMike mode

2 - Silent mode for Q-Motion®

Data type

INT

Command level

1

Item type

Axis

Source of data

PC software commands (SPA (p. 168), SEP (p. 167)) or operating elements

Possible values

0	Q-Motion® mode: A commanded movement is done in fast step mode. At the end of the movement, analog mode will be continued until the entrance window around the target position has been reached. On-target status is then set or if necessary, after a predefined delay time.
1	PiezoMike mode: A commanded movement is done in fast step mode until the entrance window for single-step mode mode has been reached. After that, single-step mode will be continued until the entrance window around the target position has been reached. On-target status is then set or if necessary, after a predefined delay time.
2	Silent mode for Q-Motion® Motion analog to Q-Motion® mode, but separate P and I term parameters are used for control.

0x1F0007A4 PiezoMike Enter Window (Phys. Unit)

Used in: On-Target State (39)

Description:	Entrance window around the target position for motion in closed-loop operation in PiezoMike mode. Commanded movement will be continued until the entrance window around the target position has been reached. The target position is then considered as reached and the on-target signal is output, if necessary, after a delay time (parameter 0x3F (p. 201)). This parameter will only be evaluated if 0x1F0007A3 (p. 228) is set to 1 (PiezoMike mode).
Data type	FLOAT
Command level	1
Item type	Axis
Source of data	Positioner database

0x1F0007A5 PiezoMike Exit Window (Phys. Unit)

Used in: On-Target State (39)

Description:	Exit window around the target position for motion in closed-loop operation in PiezoMike mode. Defines a tolerance range around the entrance window surrounding the target position (parameter 0x1F0007A4 (p. 229)) to maintain the on-target signal after a movement. If the current position leaves the exit window, the on-target signal will no longer be output and the system will try to reach the target position once again. This parameter will only be evaluated if 0x1F0007A3 (p. 228) is set to 1 (PiezoMike mode).
Data type	FLOAT
Command level	1
Item type	Axis
Source of data	Positioner database

0x1F0007A7 PiezoMike Single Step Enter Factor

Used in: Motion in closed-loop operation (29)

Description:	Entrance window for single-step mode in PiezoMike mode for motion in closed-loop operation. Commanded movement is done in fast step mode (multi-step) until the entrance window for single-step mode has been reached. Motion continues in single-step mode until the entrance window around the target position (parameter 0x1F0007A4 (p. 229)) has been reached. This parameter will only be evaluated if 0x1F0007A3 (p. 228) is set to 1 (PiezoMike mode).
Data type	FLOAT
Command level	1
Item type	Axis
Source of data	Positioner database
Possible values	Specified as entrance window factor around the target position

0x1F0007A8 PiezoMike Single Step Exit Factor

Used in: Motion in closed-loop operation (29)

Description:	Exit window for single-step mode in PiezoMike mode for motion in closed-loop operation. Defines a tolerance range around the entrance window for single-step mode (parameter 0x1F0007A7 (p. 230)) to maintain single-step mode. When the current position leaves the exit window, the system switches to fast step mode (multi step) to reach the target position. This parameter will only be evaluated if 0x1F0007A3 (p. 228) is set to 1 (PiezoMike mode).
Data type	FLOAT
Command level	1
Item type	Axis
Source of data	Positioner database
Possible values	Specified as exit window factor around the target position (parameter 0x1F0007A5 (p. 229))

0x1F0007C2 PIShift Threshold DeltaPos For Hardstop (Phys. Unit)

Used in: Behavior with Motion Error (78)

Description:	Maximum position error for certain period of time. If the difference between the commanded and actual position is greater than this value for a certain period of time, a motion error (p. 78) is triggered and stops axis motion. Difference to parameter 0x8 (p. 191) (**Maximum Position Error**): The maximum position error triggers a motion error **immediately** and stops the movement of the axis.
Data type	FLOAT
Command level	1
Item type	Axis
Source of data	Positioner database

11 Maintenance

11.1 Cleaning

Requirements

✓ You have disconnected the E-873.1AT from the power supply.

Auxiliary Materials Required

- Soft, lint-free cloth
- Mild cleaning agent or disinfectant

If you have any questions on the auxiliary materials recommended for the E-873.1AT, contact our customer service department (p. 10).

NOTICE

Short circuits or flashovers!

The E-873.1AT contains electrostatically sensitive devices that can be damaged by short circuits or flashovers when cleaning fluids penetrate the housing.

► Before cleaning, disconnect the E-873.1AT from the power supply.

► Prevent cleaning fluid from penetrating the case.

Cleaning the E-873.1AT

1. Dampen the cloth with the cleaning agent or disinfectant.
2. Carefully wipe the surfaces of the E-873.1AT.

11.2 Updating the Firmware

The following describes the procedure for updating the E-873.1AT's firmware.

Requirements

✓ You have connected the E-873.1AT to the PC via the RS-232 interface (p. 57).

✓ "PIFirmwareManager" is installed on the PC (p. 51).

✓ 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 that 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-873.1AT changes.

✓ You have saved the parameter values of the E-873.1AT in a text file on the PC (p. 76).

✓ You have saved the controller macros of the E-873.1AT in files on the PC (p. 77).

✓ You have established communication between the E-873.1AT and the PC with PIMikroMove® or PITerminal (p. 59).

NOTICE

Malfunction due to faulty firmware update!

A faulty or incomplete update of the E-873.1AT's firmware may mean that the E-873.1AT can only be made operational again by the PI customer service department.

- ▶ Update the E-873.1AT's firmware only after consulting our customer service department. If possible, ask our customer service department to do the firmware update for you.
- ▶ Before updating the firmware, make sure that you have received a suitable firmware from our customer service department and have stored it at a location that is accessible to the update program.

Information

The status LED of the E-873.1AT flashes as long as the E-873.1AT is in firmware update mode. The E-873.1AT exits the firmware update mode only when it **reboots** after the firmware was **successfully** updated. If the firmware update was unsuccessful or aborted, the E-873.1AT remains in the firmware update mode after a reboot.

If the status LED lights up continuously although the E-873.1AT was restarted after updating the firmware:

- ▶ Repeat the firmware update.
- ▶ If the update of the firmware fails, contact our customer service department (p. 10).

Updating the Firmware of the E-873.1AT

1. Start the ***PIFirmwareManager*** program on the PC and update the controller firmware. Proceed as described in the user manual SM164E (p. 8).

Restarting the E-873.1AT

1. Switch off the E-873.1AT.
2. Switch the E-873.1AT on again.

➔ *If the firmware update was successful, the E-873.1AT exits the firmware update mode and indicates that it is in operating mode again.*

➔ *If new parameters were added during updating of the firmware or the memory management of the E-873.1AT was changed: Initialize the E-873.1AT.*

Initializing the E-873.1AT after a Firmware Update

1. Make sure that the current parameter values and controller macros of the E-873.1AT have been saved on the PC.
   The initialization of the E-873.1AT 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.
2. On the PC, start PITerminal or PIMikroMove, connect to the E-873.1AT, and, if necessary, open the window to send commands.
   Initialize the E-873.1AT by sending the following commands successively:
   ```
   ZZZ 100 parameter
   ZZZ 100 macros
   ```

➔ *After successful initialization, the controller issues a corresponding message.*

3. Adapt the parameter values of the E-873.1AT:
   - 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.

4. If you have saved controller macros on the PC: Load the controller macros into the E-873.1AT, see "Loading Controller Macros from the PC into the E-873.1AT (p. 77)".

12 Troubleshooting

The positioner does not move	
Cable not connected correctly	▶ Check the cable connections.
Unsuitable positioner cable used. Interference with the signal transmission between the positioner and E-873.1AT can occur when an unsuitable cable is used.	▶ Only use genuine PI parts when connecting the positioner to the E-873.1AT. ▶ If you need extension cables, contact our customer service department (p. 10).
Positioner or drive cable is defective	▶ If available, replace the defective positioner with another one and test the new combination. ▶ If available, replace the defective drive cable with another one and test the new combination.
Positioner was connected to the switched-on E-873.1AT The sensor electronics in the positioner was not initialized and the sensor's ID chip was not read out.	▶ Switch the E-873.1AT off and on again, or reboot the E-873.1AT with the RBT command or with the corresponding functions of the PC software.
Limit switch signal logic wrongly set In order for the positioner to be able to move, the settings of the E-873.1AT must correspond to the limit switch logic level of the positioner.	▶ Adjust the **Limit Mode** (p. 196) parameter (0x18) accordingly.
Limit switch signals not compatible with the E-873.1AT It is possible that positioners from third-party suppliers use unsuitable limit switch signals.	▶ Contact the customer service department or the manufacturer of the positioner.
Motion platform has triggered the limit switch	▶ Switch the servo mode on for the affected axis again. ▶ Command the axis to move away from the limit switch.
Incorrect axis or channel commanded	▶ Make sure that the correct axis or channel identifier is used and that the positioner is connected correctly.
Incorrect configuration	▶ Check the parameter settings of the E-873.1AT with the SPA? (p. 169) (volatile memory) and the SEP? (p. 168) commands (nonvolatile memory), and make the necessary corrections, refer to adapting settings (p. 108).
Incorrect command or incorrect syntax	▶ Send the ERR? command and check the error code that is returned.
HID control active Motion commands are not permitted when HID control is activated for the axis or the channel.	▶ Deactivate HID control (p. 90).
Positioner performs unintentional motion	
Control device is not connected, but HID control is activated in the E-873.1AT	▶ Activate HID control (p. 90) only when a control device is actually connected to the E-873.1AT.
HID axis is not calibrated	▶ Calibrate the axis of the control device (p. 93).

Startup macro is run	▶ Check whether a macro is specified as the startup macro (p. 100) and cancel selection of the startup macro if necessary.
Stage is oscillating or positions inaccurately	
The load was changed	▶ Reset the notch filter (p. 68) and the servo control parameters (p. 72) according to the change of load.
Stage is oscillating already during the reference move	
Very high load on the positioner	▶ Stop the reference move immediately. ▶ Switch servo mode off. ▶ Enter suitable values for setting the notch filter (p. 68). ▶ Restart the reference move. ▶ If the positioner continues to oscillate, repeat the specified steps until the reference move has completed successfully without oscillation.
Communication between the E-873.1AT and the PC not functioning	
Wrong communication cable used	▶ Use a straight-through cable for TCP/IP connections to a network. ▶ Use a crossover network cable for TCP/IP connections direct to the PC. ▶ Use a null-modem cable for RS-232 connections. ▶ Use a standard-compliant USB cable (type A to type B or type A to type Mini-B) for USB connections
Communication cable defective	▶ Replace the communication cable.
RS-232 interface not configured correctly	▶ Check the port settings, the baud rate and the handshake setting of the PC.
TCP/IP interface not configured correctly	▶ Connect the controller to the network **before** you switch it on. Restart the E-873.1AT if necessary. ▶ Check the network settings. ▶ Check the firewall settings to ensure that UDP broadcast is permitted. If necessary, contact your network administrator. Alternatively, enter the IP address of the controller manually. ▶ Make sure that the network is not blocked for unknown devices. ▶ Make sure that you have selected the correct E-873.1AT when establishing communication. ▶ If you cannot solve the problems, consult your network administrator if necessary.
Another program is accessing the interface	▶ Close the other program.
The start procedure of the E-873.1AT's firmware has not finished yet	▶ Wait until the corresponding LED indicates operational readiness after switching on or rebooting the E-873.1AT. ▶ Try to establish communication.

Problems with special software	▶ Check whether the system works with other software, e.g., a terminal program, or a development environment; for this purpose, enter the `*IDN?` or `HLP?` command using that software. Make sure that you end commands with an LF (line feed); a command is only executed when an LF is received.
E-873.1AT does not send an error code in the case of incorrect system behavior	
The error code was already queried by a different instance In the case of simultaneous access to the E-873.1AT by several instances, the error code is only returned to the first instance that sent the `ERR?` command. The error code is reset to 0 during the query.	▶ If possible, access the E-873.1AT with one instance only. ▶ Check whether the error code is queried regularly in the background by a macro or script or PC software (e.g., PIMikroMove).
LEDs do not light up even though the E-873.1AT is switched on	
E-873.1AT not connected to the power supply or the the power cord is defective.	▶ Switch off the E-873.1AT. ▶ Make sure that the E-873.1AT is connected to the power supply and the power cord is not defective. ▶ Switch on the E-873.1AT.

If the problem is not listed in the table or cannot be solved as described, contact our customer service department (p. 10).

13 Technical Data

Subject to change. You can find the latest product specifications on the product web page at www.pi.ws.

13.1 Specifications

	E-873.1AT
Function	Q-Motion® controller for positioning systems with piezo inertia drives, also suitable for closed-loop PiezoMikes Benchtop device with option for cabinet mounting
Axes	1
Supported functions	Startup macro. Data recorder for recording operating data such as motor voltage, velocity, position or position error. Internal safety circuitry: Watchdog timer. ID chip detection.

Motion and control	E-873.1AT
Controller type	PID controller, parameters modifiable during operation
Servo cycle time	50 µs
Dynamics profile	Trapezoid velocity profile. Point-to-point motion.
Encoder input	Analog encoder inputs sine-cosine, interpolation to 20000. Interpolation electronics preset for differential transmission, 1 Vpp and 2.5 V encoder signal offset. BiSS interface for absolute encoders
Stall detection	Automatic motor stop
Input limit switch	2 × TTL (pull-up / pull-down, programmable)
Input reference switch	1 × TTL for integrated reference in the encoder

Electrical properties	E-873.1AT
Max. output power	30 W
Output voltage	0 to 100 V, set automatically when a positioner is connected, depending on its drive

Interfaces and operation	E-873.1AT
Communication interfaces	TCP/IP: RJ45/Ethernet; USB: Mini-B; RS-232: D-sub 9 (m); SPI
Motor / sensor connector	D-sub 15 (f)
I/O lines	4 analog / digital inputs, 4 digital outputs
Command set	PI General Command Set (GCS)
User software	PIMikroMove
Application programming interfaces	C, C++, C#, MATLAB, NI LabVIEW, Python
Manual control (optional)	Analog joystick

Miscellaneous	E-873.1AT
Operating Voltage	24 V DC from external power adapter (included in the scope of delivery)
Max. current consumption	1.5 A
Operating temperature range	0 to 50 °C
Mass	0.335 kg
Dimensions	147 mm × 105 mm × 44 mm (incl. mounting rails)

13.2 Maximum Ratings

The E-873.1AT is designed for the following operating data:

⚠	Maximum operating voltage	Maximum operating frequency	Maximum current consumption
	24 V	-	2.5 A

13.3 Ambient Conditions and Classifications

The following ambient conditions and classifications for the E-873.1AT must be observed:

Area of application	For indoor use only
Maximum altitude	2000 m above msl
Air pressure	1100 hPa to 0.1 hPa
Relative humidity	Max. 80 % for temperatures to 31 °C, decreasing linearly to 50 % 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
Degree of protection according to IEC 60529	IP20

13.4 Dimensions

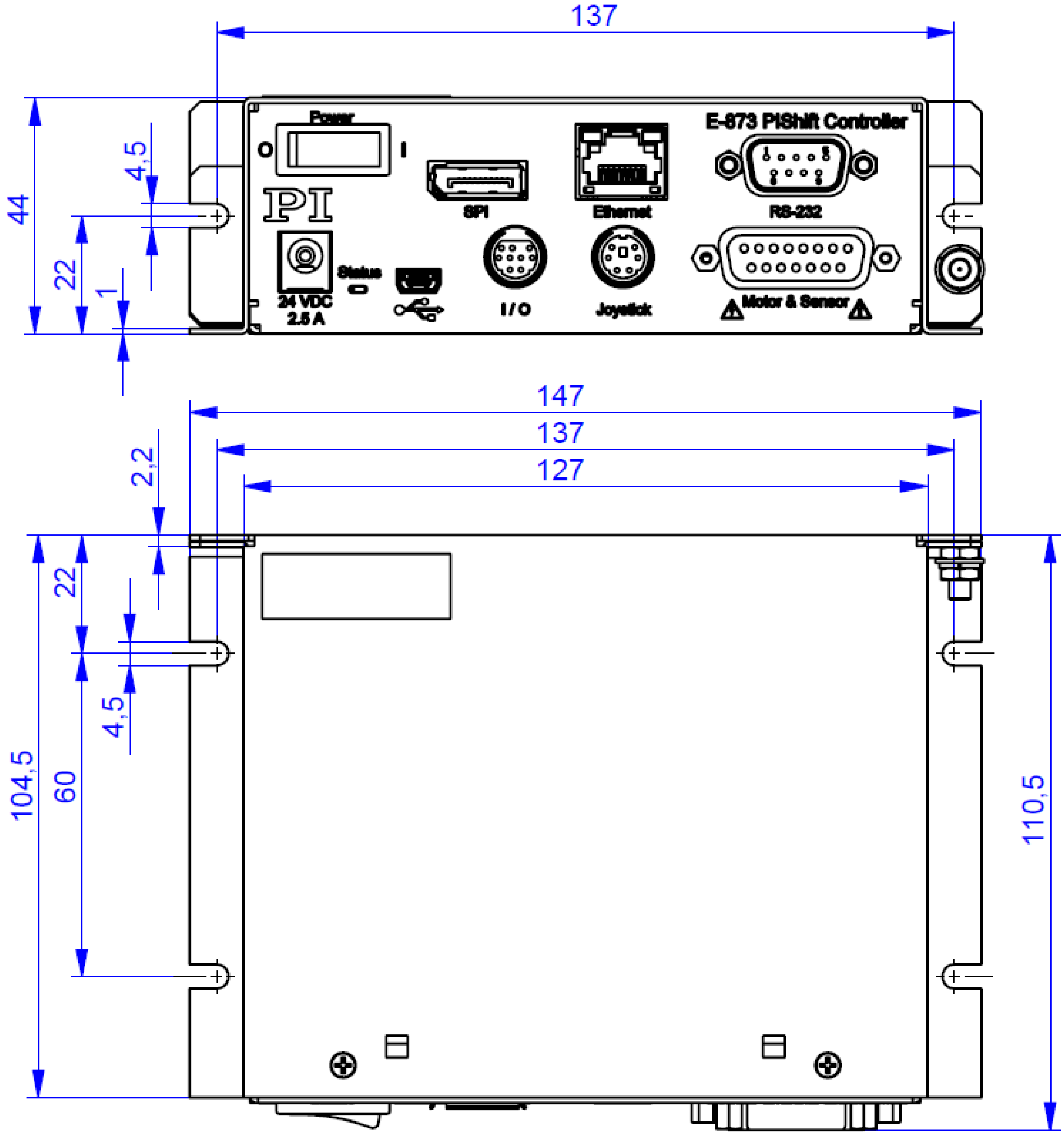


Figure 15: Dimensions of the E-873.1AT

Dimensions in mm. Note that the decimal places are separated by a comma in the drawings.

14 Appendix

14.1 Pin Assignment

14.1.1 Motor and Sensor Connector

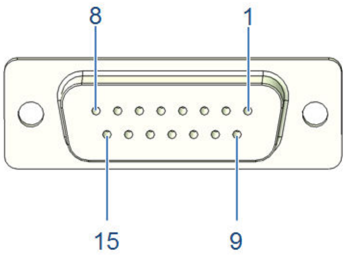


Figure 16: D-sub 15 (f)

Pin	Signal	Function
1	REF-	Reference switch, differential (-)
2	PIEZO-	Motor signal (-)
3	PIEZO+	Motor signal (+)
4	5 V	Supply voltage, +5 V
5	PLIM	Positive limit switch
6	ID_CHIP	Bidirectional: Data line for ID chip
7	ENCA-	Encoder channel A, differential (-)
8	ENCB-	Encoder channel B, differential (-)
9	PIEZO-	Motor signal (-)
10	GND	Ground
11	PIEZO+	Motor signal (+)
12	NLIM	Negative limit switch
13	REF+	Reference switch, differential (+)
14	ENCA+	Encoder channel A, differential (+)
15	ENCB+	Encoder channel B, differential (+)

14.1.2 I/O Connector

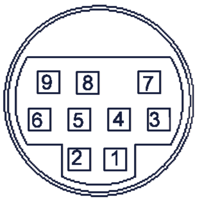


Figure 17: I/O connector: Mini DIN 9 (f)

Pin	Function
1	Input 1 (analog: 0 to +5 V / digital: TTL)
2	Input 2 (analog: 0 to +5 V / digital: TTL)
3	Input 3 (analog: 0 to +5 V / digital: TTL)
4	Input 4 (analog: 0 to +5 V / 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	Ground

14.1.3 C-170.IO Cable for Connecting to the I/O Socket

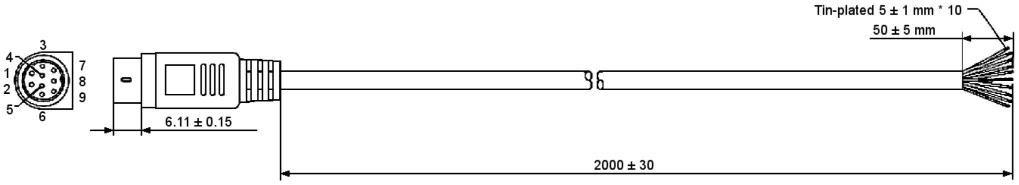


Figure 18: C-170.IO cable

Pin	Wire Color	Function on the I/O socket of the E-873.1AT
1	Black	Input 1 (analog: 0 to +5V / digital: TTL)
2	white	Input 2 (analog: 0 to +5V / digital: TTL)
3	Red	Input 3 (analog: 0 to +5V / digital: TTL)
4	Yellow	Input 4 (analog: 0 to +5V / digital: TTL)
5	Purple	Output 1 (digital, TTL)
6	Blue	Output 2 (digital, TTL)
7	Green	Output 3 (digital, TTL)
8	Brown	Output 4 (digital, TTL)
9	Gray	Vcc (+5V)
Sheath	Shield, coated black (thicker than the wire connected to pin 1)	GND

14.1.4 Joystick Connector

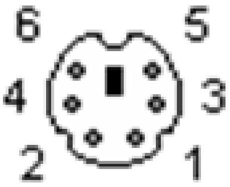


Figure 19: Joystick connector: Mini DIN 6 (f)

Pin	Function
1	GND
2	Input: Axis 2 of HID 1 (0 to 3.3 V)
3	Output: Vcc (3.3 V)
4	Input: HID 1 axis 1 (0 to 3.3 V)
5	Input: HID 1 button 1 (0 or 3.3 V)
6	Input: HID 1 button 2 (0 or 3.3 V)

14.1.5 RS-232 Connector

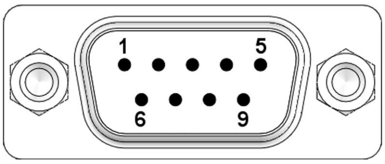


Figure 20: D-sub 9 (m)

Pin	Function
1	Not connected
2	RxD (PC to controller)
3	TxD (controller to PC)
4	Not connected
5	GND
6	Not connected
7	Not connected
8	Not connected
9	Not connected

14.1.6 Power Adapter Connector

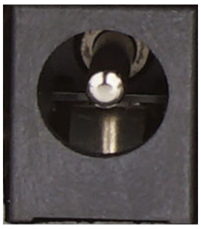


Figure 21: Power adapter connector: barrel connector socket

Pin	Signal	Function
Center pin	V+	Input: Power supply, +24 V
Line connector	GND	Ground

14.2 GCS 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

Controller errors		
20	PI_CNTR_MACRO_NOT_FOUND	Macro not found
21	PI_CNTR_AXIS_HAS_NO_BRAKE	Axis has no brake
22	PI_CNTR_DOUBLE_AXIS	Axis identifier specified more than once
23	PI_CNTR_ILLEGAL_AXIS	Illegal axis
24	PI_CNTR_PARAM_NR	Incorrect number of parameters
25	PI_CNTR_INVALID_REAL_NR	Invalid floating point number
26	PI_CNTR_MISSING_PARAM	Parameter missing
27	PI_CNTR_SOFT_LIMIT_OUT_OF_RANGE	Soft limit out of range
28	PI_CNTR_NO_MANUAL_PAD	No manual pad found
29	PI_CNTR_NO_JUMP	No more step-response values
30	PI_CNTR_INVALID_JUMP	No step-response values recorded
31	PI_CNTR_AXIS_HAS_NO_REFERENCE	Axis has no reference sensor
32	PI_CNTR_STAGE_HAS_NO_LIM_SWITCH	Axis has no limit switch
33	PI_CNTR_NO_RELAY_CARD	No relay card installed
34	PI_CNTR_CMD_NOT_ALLOWED_FOR_STAGE	Command not allowed for selected stage(s)
35	PI_CNTR_NO_DIGITAL_INPUT	No digital input installed
36	PI_CNTR_NO_DIGITAL_OUTPUT	No digital output configured
37	PI_CNTR_NO_MCM	No more MCM responses
38	PI_CNTR_INVALID_MCM	No MCM values recorded
39	PI_CNTR_INVALID_CNTR_NUMBER	Controller number invalid

Controller errors		
40	PI_CNTR_NO_JOYSTICK_CONNECTED	No joystick configured
41	PI_CNTR_INVALID_EGE_AXIS	Invalid axis for electronic gearing, axis can not be slave
42	PI_CNTR_SLAVE_POSITION_OUT_OF_RANGE	Position of slave axis is out of range
43	PI_CNTR_COMMAND_EGE_SLAVE	Slave axis cannot be commanded directly when electronic gearing is enabled
44	PI_CNTR_JOYSTICK_CALIBRATION_FAILED	Calibration of joystick failed
45	PI_CNTR_REFERENCING_FAILED	Referencing failed
46	PI_CNTR_OPM_MISSING	OPM (Optical Power Meter) missing
47	PI_CNTR_OPM_NOT_INITIALIZED	OPM (Optical Power Meter) not initialized or cannot be initialized
48	PI_CNTR_OPM_COM_ERROR	OPM (Optical Power Meter) Communication Error
49	PI_CNTR_MOVE_TO_LIMIT_SWITCH_FAILED	Move to limit switch failed
50	PI_CNTR_REF_WITH_REF_DISABLED	Attempt to reference axis with referencing disabled
51	PI_CNTR_AXIS_UNDER_JOYSTICK_CONTROL	Selected axis is controlled by joystick
52	PI_CNTR_COMMUNICATION_ERROR	Controller detected communication error
53	PI_CNTR_DYNAMIC_MOVE_IN_PROCESS	MOV! motion still in progress
54	PI_CNTR_UNKNOWN_PARAMETER	Unknown parameter
55	PI_CNTR_NO_REP_RECORDED	No commands were recorded with REP
56	PI_CNTR_INVALID_PASSWORD	Password invalid
57	PI_CNTR_INVALID_RECORDER_CHAN	Data Record Table does not exist

Controller errors		
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 non-volatile 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 can not be started
69	PI_CNTR_DDL_TIME_DELAY_TOO_LARGE	Time delay larger than DDL table; DDL can not be started
70	PI_CNTR_DIFFERENT_ARRAY_LENGTH	The requested arrays have different lengths; query them separately
71	PI_CNTR_GEN_SINGLE_MODE_RESTART	Attempt to restart the generator while it is running in single step mode
72	PI_CNTR_ANALOG_TARGET_ACTIVE	Motion commands and wave generator activation are not allowed when analog target is active
73	PI_CNTR_WAVE_GENERATOR_ACTIVE	Motion commands are not allowed when wave generator is active
74	PI_CNTR_AUTOZERO_DISABLED	No sensor channel or no piezo channel connected to selected axis (sensor and piezo matrix)
75	PI_CNTR_NO_WAVE_SELECTED	Generator started (WGO) without having selected a wave table (WSL).

Controller errors		
76	PI_CNTR_IF_BUFFER_OVERRUN	Interface buffer did 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 allowed when servo is on
80	PI_CNTR_RAM_ERROR	Hardware error affecting RAM
81	PI_CNTR_MACRO_UNKNOWN_COMMAND	Not macro command
82	PI_CNTR_MACRO_PC_ERROR	Macro counter out of range
83	PI_CNTR_JOYSTICK_ACTIVE	Joystick is active
84	PI_CNTR_MOTOR_IS_OFF	Motor is off
85	PI_CNTR_ONLY_IN_MACRO	Macro-only command
86	PI_CNTR_JOYSTICK_UNKNOWN_AXIS	Invalid joystick axis
87	PI_CNTR_JOYSTICK_UNKNOWN_ID	Joystick unknown
88	PI_CNTR_REF_MODE_IS_ON	Move without referenced stage
89	PI_CNTR_NOT_ALLOWED_IN_CURRENT_MOTION_MODE	Command not allowed in current motion mode
90	PI_CNTR_DIO_AND_TRACING_NOT_POSSIBLE	No tracing possible while digital IOs are used on this HW revision. Reconnect to switch operation mode.
91	PI_CNTR_COLLISION	Move not possible, would cause collision
92	PI_CNTR_SLAVE_NOT_FAST_ENOUGH	Stage is not capable of following the master. Check the gear ratio.
93	PI_CNTR_CMD_NOT_ALLOWED_WHILE_AXIS_IN_MOTION	This command is not allowed while the affected axis or its master is in motion.

Controller errors		
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

Controller errors		
212	PI_CNTR_FILE_WRITE_ERROR	Error writing file on controller
213	PI_CNTR_DTR_HINDERS_VELOCITY_CHANGE	VEL command not allowed in DTR Command Mode
214	PI_CNTR_POSITION_UNKNOWN	Position calculations failed
215	PI_CNTR_CONN_POSSIBLY_BROKEN	The connection between controller and stage may be broken
216	PI_CNTR_ON_LIMIT_SWITCH	The connected stage has driven into a limit switch, some controllers need CLR to resume operation
217	PI_CNTR_UNEXPECTED_STRUT_STOP	Strut test command failed because of an unexpected strut stop
218	PI_CNTR_POSITION_BASED_ON_ESTIMATION	While MOV! is running position can only be estimated!
219	PI_CNTR_POSITION_BASED_ON_INTERPOLATION	Position was calculated during MOV motion
220	PI_CNTR_INTERPOLATION_FIFO_UNDERRUN	FIFO buffer underrun during interpolation
221	PI_CNTR_INTERPOLATION_FIFO_OVERFLOW	FIFO buffer overflow 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

Controller errors		
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

Controller errors		
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

Controller errors		
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

Controller errors		
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?
659	PI_CNTR_PARAM_CONFLICT	Parameter could not be set. Conflict with another parameter.

Controller errors		
700	PI_CNTR_COMMAND_NOT_ALLOWED_IN_EXTERNAL_MODE	Command not allowed 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 cycletime invalid
720	PI_CNTR_BRAKE_ACTIVATED	Brake is activated
725	PI_CNTR_DRIVE_STATE_TRANSITION_ERROR	Drive state transition error
731	PI_CNTR_SURFACEDETECTION_RUNNING	Command not allowed 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

Controller errors		
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
1012	PI_CNTR_ERROR_IN_MACRO	Error occured 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_CMD	User Profile Mode: Command is not allowed, check for required preparatory commands
1064	PI_CNTR_EXT_PROFILE_EXPECTING_MOTION_ERROR	User Profile Mode: First target position in User Profile is too far from current position
1065	PI_CNTR_PROFILE_ACTIVE	Controller is (already) in User Profile Mode
1066	PI_CNTR_PROFILE_INDEX_OUT_OF_RANGE	User Profile Mode: Block or Data Set index out of allowed range
1071	PI_CNTR_PROFILE_OUT_OF_MEMORY	User Profile Mode: Out of memory
1072	PI_CNTR_PROFILE_WRONG_CLUSTER	User Profile Mode: Cluster is not assigned to this axis
1073	PI_CNTR_PROFILE_UNKNOWN_CLUSTER_IDENTIFIER	Unknown cluster identifier
1090	PI_CNTR_TOO_MANY_TCP_CONNECTIONS_OPEN	There are too many open tcpip connections

Controller errors		
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 can not be executed during scanning/recording
5002	PI_CNTR_INVALID_PCC_AXIS	Given axis cannot be defined as PPC axis
5003	PI_CNTR_PCC_SCAN_OUT_OF_RANGE	Defined scan area is larger than the travel range
5004	PI_CNTR_PCC_TYPE_NOT_EXISTING	Given PicoCompensation type is not defined
5005	PI_CNTR_PCC_PAM_ERROR	PicoCompensation parameter error
5006	PI_CNTR_PCC_TABLE_ARRAY_TOO_LARGE	PicoCompensation table is larger than maximum table length
5100	PI_CNTR_NEXLINE_ERROR	Common error in NEXLINE® firmware module

Controller errors		
5101	PI_CNTR_CHANNEL_ALREADY_USED	Output channel for NEXLINE® can not 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 can not 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

Interface errors		
-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

Interface errors		
-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

Interface errors		
-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

DLL errors		
-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

DLL errors		
-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

DLL errors		
-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

DLL errors		
-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

DLL errors		
-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.
-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.

DLL errors		
-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_CONTANS_UNKNOWN_PAM_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.

DLL errors		
-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.

Glossary

Daisy chain

Wiring diagram by which one controller is connected to the next in sequence (series connection principle). The first controller is connected directly to the PC. The additional controllers are always connected to the ones that precede them so that a chain is formed. The signal to and from a controller goes to the PC via the previous controllers.

Dynamics profile

The dynamic profile includes the target position, speed, and acceleration of the axis calculated by the profile generator of the electronics for each point in time of motion. The calculated values are called "commanded values".

GCS

Abbreviation for "General Command Set", the command set for electronics from PI. Piezo and servo controllers can be operated together with minimal programming effort thanks to GCS.

HID

Abbreviation for "Human Interface Device". HID refers to an input or output device that is connected to the electronics and intended for manual operation. Depending on the electronics, the connection can be made via USB, analog or digital interfaces. Joysticks and gamepads are typical HIDs.

Nonvolatile memory

Electronics read-only memory. The default values for the parameters are loaded from the nonvolatile memory into the volatile memory when the electronics are started. The parameter values in the nonvolatile memory are also referred to as "Startup Values" in the PC software from PI.

Specifications

The performance specifications are checked before dispatch. The performance specifications apply to room temperature (22 ±3 °C), systems in closed-loop operation are calibrated at this temperature. It may be necessary to reset the operating parameters when operating at considerably lower or higher temperatures.

Volatile memory

Electronics main memory. Parameters are stored in the volatile memory when the device is switched on. 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.